blog-nsfocus-net.translate.goog /darkcasino-apt-evilnum/

Operation DarkCasino: In-depth analysis of recent attacks by APT group Evilnum

Fuying Laboratory

I. Overview

Recently, NSFOCUS’s Fuying Lab captured a series of phishing campaigns targeting European countries. These activities are mainly aimed at online gambling platforms, and the goal is to steal the transaction credentials of service providers and consumers by attacking the active online transaction behavior behind such services, and then obtain illegal profits.

Through in-depth analysis, Fuying Lab determined that this series of activities is a continuation of the recent attack activities of the APT organization Evilnum (http://blog.nsfocus.net/agentvxapt-evilnum/). Compared with previous activities, the Evilnum attackers inherited their representative attack methods in this operation, but used more diverse attack processes and complex attack components, and enabled two new Trojan programs, DarkMe and PikoloRAT, Demonstrated its high tool development ability, process design ability and rich experience in offensive and defensive confrontation. At the same time, due to the obvious differences in the design ideas and specific implementations of different attack processes, Fuying Lab believes that multiple attackers participated in this operation at the same time.

By extracting and combining the keywords of the attack target and the main Trojan program, Fuying Lab named the Evilnum operation DarkCasino. The operation shows that Evilnum is still primarily targeting online trading platforms and is able to quickly spot cybercrime opportunities and execute attacks.

As of the time of the report, the DarkCasino operation was still ongoing.

2. Organizational Information

Evilnum is an APT group discovered in 2018, active in the UK and EU countries, mainly targeting financial technology companies, with the purpose of stealing corporate or personal account funds by stealing transaction credentials. The group name Evilnum comes from the Trojan of the same name, also known as DeathStalker by Kaspersky.

Evilnum's representative attack method is to disguise malicious programs as customer identification documents, deceive the staff of financial companies to run these programs, and then obtain high-value information on the victim's host by implanting spyware Trojans.

Evilnum has strong development capabilities and can design complex attack processes and attack components. NSFOCUS Fuying Lab has captured and disclosed the organization's highly complete attack process and a stub-type Trojan, AgentVX.

3. Impact analysis

The analysis found that the victims of Evilnum’s operation were mainly distributed in European countries in the Mediterranean region and related countries such as Canada, Singapore, and the Philippines. Its direct attack targets included online casino platforms, consumers in various countries using such platforms, and online casinos. other persons involved in the above transactions.

In the discovered attack flow, Evilnum uses the following string as the decoy filename:

Decoy filename
offer deal visa 2022.lnk
offer crypto casino.scr
Scatters Casino offers Daily Promotions.pif
new casino crypto.com

Promo CPL CPA Traffic.com
PayRedeemUpdateIntegration19052022.scr
DOCUMENTATION AGREEMENTS S CONSULTING
INTEGRATION.pif

The Scatters Casino in the above content is an online casino operated by the Maltese company Gammix Limited. These decoy documents try to disguise themselves as online transaction proofs or advertising service promotion documents to attack the operators of scatters, while also trying to disguise themselves. Promotional advertisements for scatters to attack users of scatters, thereby enabling Evilnum attackers to obtain transaction credentials or related information held on these targets' hosts.

According to statistics on the sources of various decoy documents, Fuying Lab found that the victims of this DarkCasino activity are widely distributed in Malta, Poland, Cyprus, Armenia, Spain, Switzerland, France, Ireland and other European countries, as well as Canada, Israel and even Singapore , the Philippines and other non-European countries:

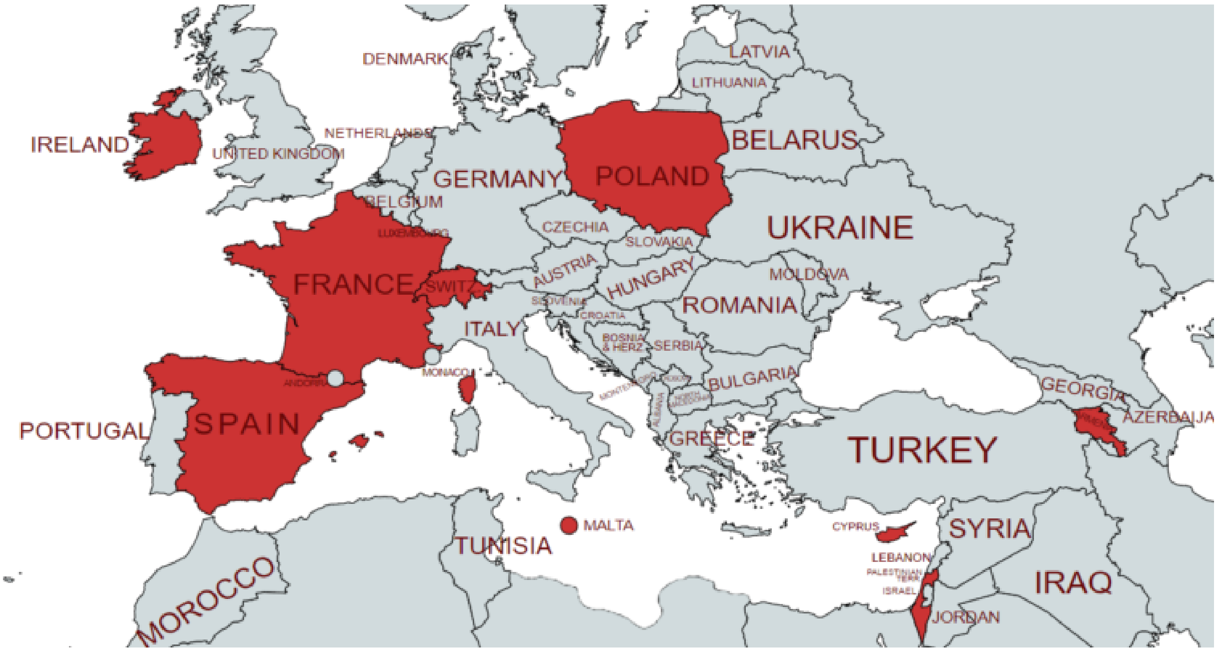


Figure 3.1 Distribution of victims of Operation DarkCasino

It can be found that the geographical location of the victims is centered on Malta and radiates to many countries that may use the services of the scatters website.

The online casino platform of scatters was established in 2019 and has expanded rapidly. Currently, scatters claims that its online casino service has a prize pool worth 230 million euros, which may be the main reason why Evilnum is targeting it in this operation.

Additionally, some information suggests that Operation DarkCasino may be part of a larger and more persistent cyber attack campaign. IoC correlation clues show that some of Evilnum's assets can be linked to a cyber-attack campaign targeting cryptocurrency-related trading platforms and users starting in the second half of 2021 and continuing into early 2022. In this activity, the attackers mainly delivered a large number of ParallaxRAT and NetWire Trojans with signatures to steal the information of the target host, and its main impact targets are mainly concentrated in European countries. Although the decoy forms and network resources used by the attackers in this activity are related to the DarkCasino operation to a certain extent, Fuying Lab has not obtained direct evidence to prove that this activity is also carried out by Evilnum, the attacker may be in the form of cooperation Borrow Evilnum assets and join them in action.

4. Analysis of Attack Process

In this operation, the attackers of the Evilnum group mainly created three different attack processes. The three types of processes start with different types of decoy files, obtain steganographic images by accessing public resources or compromised sites, extract the contents of the DarkMe Trojan payload, and then load and execute them in different ways.

4.1 Attack Process A

This is the earliest type of attack process implemented by Evilnum attackers, and it is also the process with the highest component complexity. The earliest date of discovery of the critical component is May 2.

When researching different components, we found that the attack flow contains two variants, namely flow A1, which is created on April 28, and obtains content from a network location; and one created on May 1, which does not require Internet access to download content. process A2. The actual implementation of both variants is similar.

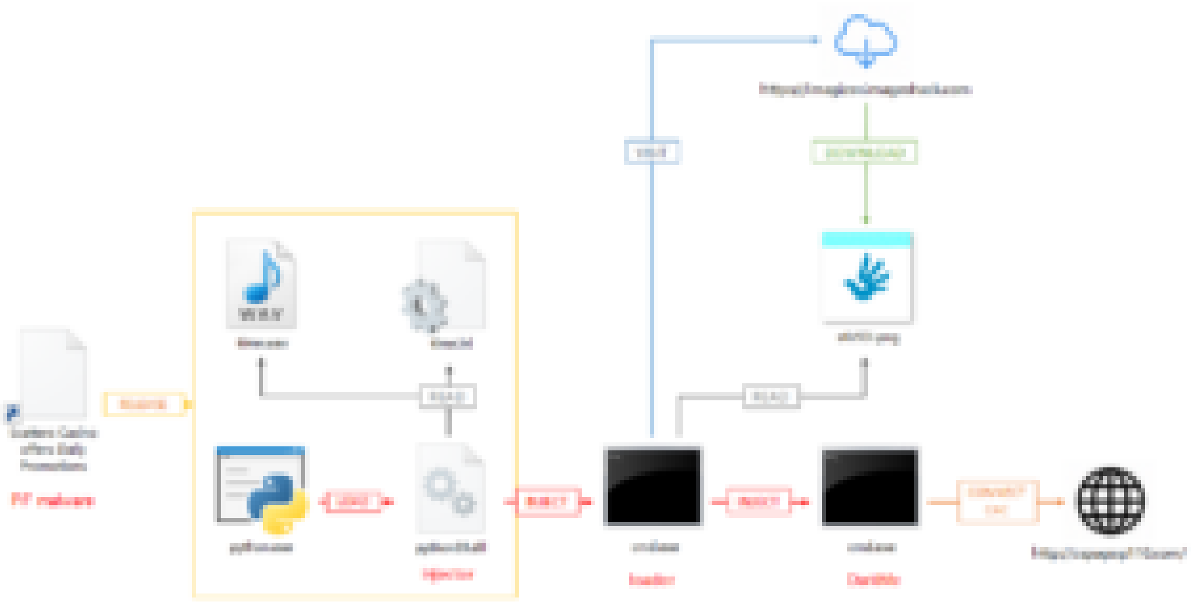


Figure 4.1 DarkCasino attack process A

The figure is a diagram of the process A1. This process is very similar to the AgentVX attack activity of the Evilnum organization (http://blog.nsfocus.net/agentvxapt-evilnum/) disclosed by Fuying Lab earlier. And the composition of the DarkMe Trojan.

After the InstallShield program disguised as a PIF file is started, it will execute the general process of the installation program, release the built-in files in the system %TEMP% directory, and run the legitimate program python.exe in it.

After python.exe is started, it will run the python39.dll program carrying malicious code in the form of side loading, thereby starting a piece of shellcode.

The function of the malicious shellcode in python39.dl is to read the time.wav file in the same directory, decrypt and extract the next-stage shellcode code, then start the cmd.exe puppet process and inject the next-stage shellcode into it, and inject the next-stage shellcode into it. A url address string read from time.ini is written into the cmd.exe puppet process as the startup parameter of the shellcode.

The shellcode in the cmd.exe puppet process will obtain a steganographic image from the above url address, and extract the third-stage shellcode from the image through the built-in image processing module and run it.

The third stage shellcode will try to inject the built-in DarkMe Trojan into another cmd.exe puppet process to run. The DarkMe Trojan communication CnC is cspapop110.com.

4.2 Attack Process B

This is a type of attack process that was first observed on May 9, and related documents show that the process was constructed on May 3.

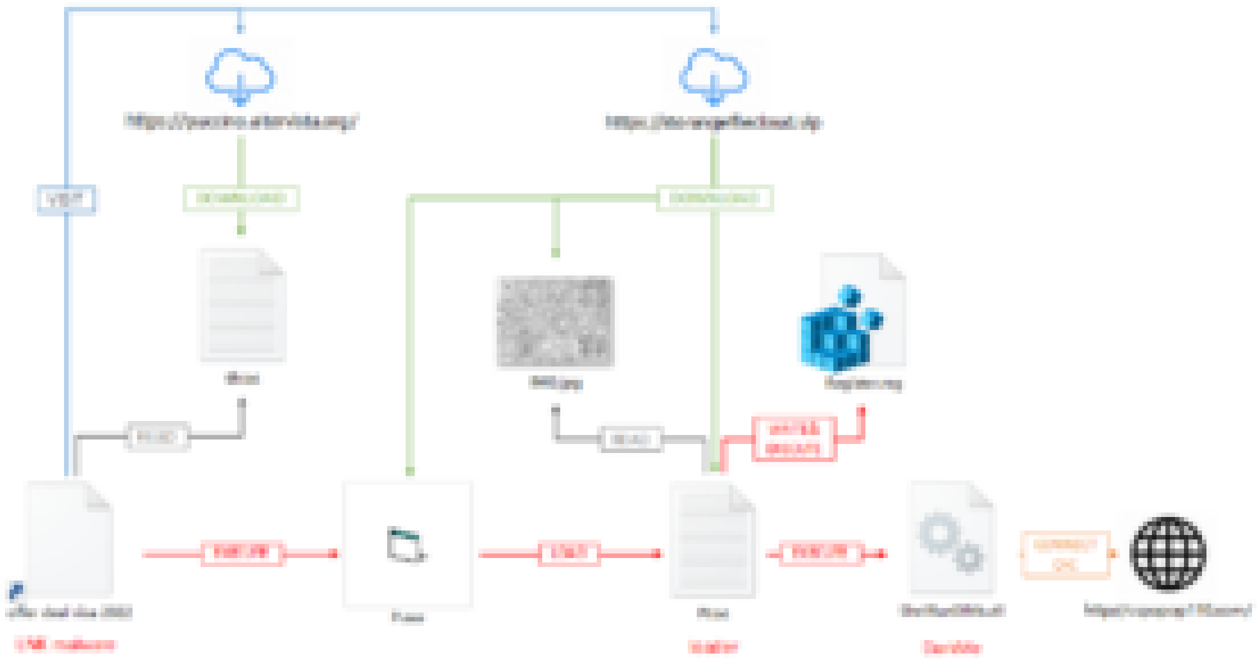


Figure 4.2 DarkCasino attack process B

The figure above is an illustration of Process B. During this process, the attackers followed the organization's consistent thinking, by delivering a shortcut decoy file with malicious mshta instructions, accessing the controlled wordpress site to obtain subsequent instruction codes and running them.

The key stage of this process lies in the three files P.exe, PI.txt and IMG.jpg obtained by visiting the second stage site. After being loaded by P.exe, the main loader Trojan PI.txt will extract the hidden executable file ShellRunDllVb.dll in IMG.jpg, and register the dll file as a system component by creating a registry file named Register.reg {A762B0C7 -5244-4B3E-ADED-D549E9CEA39E}. The loader Trojan finally executes the component through the rundll32 /sta command.

The final execution of the above operation is a spy Trojan program named DarkMe, and the communication CnC is cspapop110.com.

4.3 Attack Process C

The Evilnum attackers added a more streamlined process on May 19.

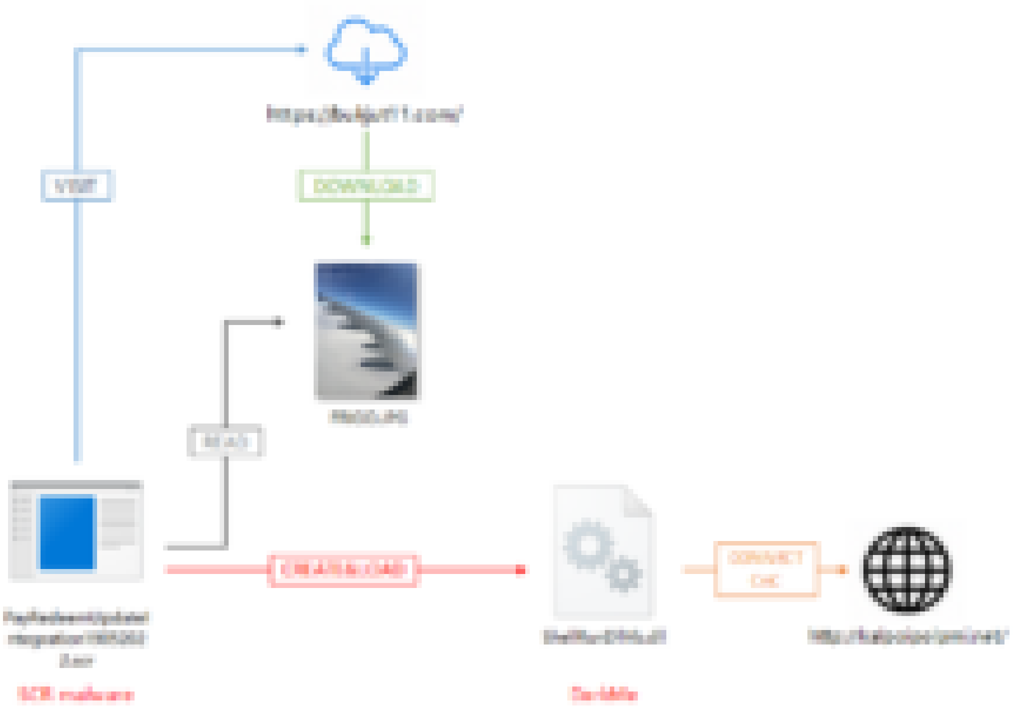


Figure 4.3 DarkCasino attack process C

The above figure is an illustration of process C. The process is initiated by a loader Trojan disguised as an scr file. The Trojan obtains a steganographic image by directly accessing the built-in url link, then extracts the ShellRunDllVb.dll file and loads it for execution. The dll file is also the DarkMe Trojan, and the communication CnC is kalpoipolpmi.net.

5. Component Analysis

In this operation, Evilnum mainly used a new type of self-made Trojan program. Fuying Lab named it DarkMe through a special string in the Trojan program.

In addition, Fuying Lab discovered another new type of Trojan program closely related to Evilnum's operation during association analysis, and named it PikoloRAT through a special string in it.

5.1 DarkMe

DarkMe is a VisualBasic spy trojan developed by Evilnum attackers and used in various attack flows. The initial version of DarkMe appeared on September 25, 2021, and 5 versions have been iterated so far.

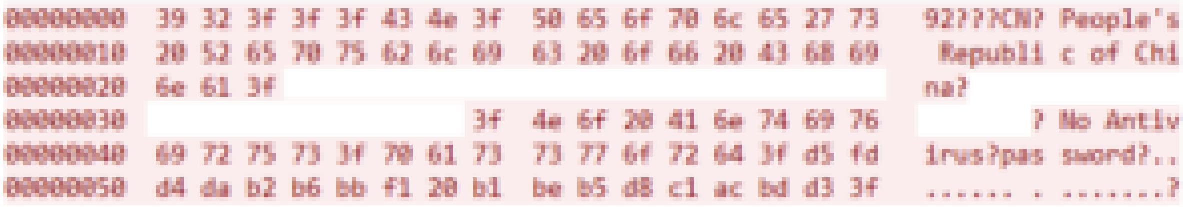
DarkMe's communication capabilities are implemented through a public WinSock32 module (http://leandroascierto.com/blog/winsock32/). This module performs socket communication with the server in the form of window information by creating a window named SOCKET_WINDOW.

On the basis of this module, the DarkMe Trojan has successively added a large number of functional codes, making it gradually evolve from a downloader Trojan to a stub-type Trojan with spyware capabilities.

5.1.1 Function

Since the function codes of different versions of DarkMe are different, the V5 version of the Trojan program (ShellRunDllVb.dll) that appeared in this operation is described here.

After the Trojan runs, it first collects host information and sends it to CnC. The information collected by the V4 version includes the geographical location abbreviation of the host computer, the full name of the country, the computer name, the user name, the anti-virus software list, the Trojan horse mark and the title of the foreground window. These information are separated by the fixed separator 0x3F, and a fixed The string "92" forms the online information and sends it to the CnC terminal.



```
00000000  39 32 3f 3f 3f 43 4e 3f  50 65 6f 70 6c 65 27 73   92???CN? People's
00000010  20 52 65 70 75 62 6c 69  63 20 6f 66 20 43 68 69    Republi c of Chi
00000020  6e 61 3f [redacted]                                  na?
00000030  [redacted]            3f  4e 6f 20 41 6e 74 69 76   [redacted] ? No Antiv
00000040  69 72 75 73 3f 70 61 73  73 77 6f 72 64 3f d5 fd   irus?pas sword?..
00000050  d4 da b2 b6 bb f1 20 b1  be b5 d8 c1 ac bd d3 3f   ....... . .......?
```

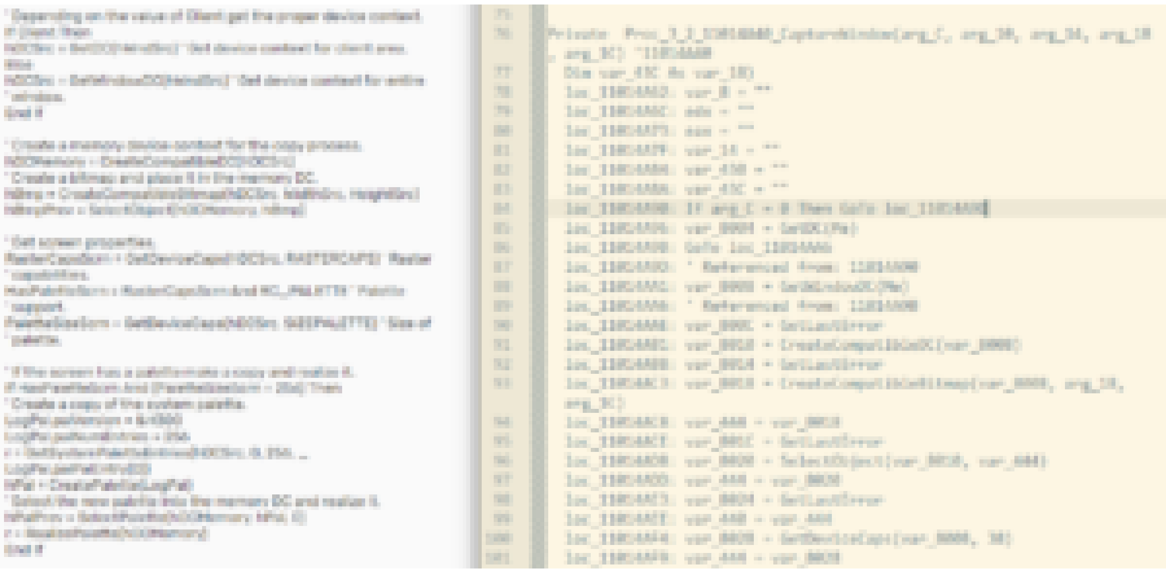
Figure 5.1 DarkMe online traffic

DarkMe implements several modules to support various espionage functions. One of the main modules is named clsfile and is used to implement file operations under CnC control. The CnC control command is given by the first 6 bytes of the communication content. The corresponding operations of each command are as follows:

instruction	**Function**
300100	Get disk volume information
STRFLS	Traverse the specified directory to get the directory structure
STRFL2	Traverse the specified directory to obtain the directory structure, support large directories
SHLEXE	execute cmd command
RNMFIL	Rename the specified file
DELDEL	delete the specified file
DIRMAP	Create the specified directory
DELMAP	delete the specified directory
SEITUS	write to the specified file
SEITUD	read the specified file
ZIPALO	write compressed file
FRIKAT	Write registry startup key
COPALO	Copy the specified file
PASALO	Paste the specified file

Table 5.1 DarkMe command comparison table

In addition, DarkMe also integrates a set of public code (https://forums.codeguru.com/showthread.php?15579-Save-Screen-Capture-output-to-a-file) to realize the screenshot function.



```
' Depending on the value of bClient get the proper device context.
If bClient Then
hDCSrc = GetDC(hWndSrc) ' Get device context for client area.
Else
hDCSrc = GetWindowDC(hWndSrc) ' Get device context for entire
' window.
End If

' Create a memory device context for the copy process.
hDCMemory = CreateCompatibleDC(hDCSrc)
' Create a bitmap and place it in the memory DC.
hBmp = CreateCompatibleBitmap(hDCSrc, WidthSrc, HeightSrc)
hBmpPrev = SelectObject(hDCMemory, hBmp)

' Get screen properties.
RasterCapsScrn = GetDeviceCaps(hDCSrc, RASTERCAPS) ' Raster
' capabilities.
HasPaletteScrn = RasterCapsScrn And RC_PALETTE ' Palette
' support.
PaletteSizeScrn = GetDeviceCaps(hDCSrc, SIZEPALETTE) ' Size of
' palette.

' If the screen has a palette make a copy and realize it.
If HasPaletteScrn And (PaletteSizeScrn = 256) Then
' Create a copy of the system palette.
LogPal.palVersion = &H300
LogPal.palNumEntries = 256
r = GetSystemPaletteEntries(hDCSrc, 0, 256, _
LogPal.palPalEntry(0))
hPal = CreatePalette(LogPal)
' Select the new palette into the memory DC and realize it.
hPalPrev = SelectPalette(hDCMemory, hPal, 0)
r = RealizePalette(hDCMemory)
End If
```

```
[illegible]
```

Figure 5.2 DarkMe screenshot function (right) and public implementation (left)

Other features of DarkMe include persistence, self-updating, and keylogging in some versions.

Version 5.1.2

Through mining samples in the wild, Fuying Lab found that the DarkMe Trojan has a development history of more than half a year and has produced multiple versions. The version iteration timeline of the Trojan is as follows:

Figure 5.3 Iterative process of DarkMe version

It can be seen that during its life cycle, the positioning of the DarkMe Trojan has changed, from a directly dropped loader-type Trojan to a spy Trojan, and then to a stub payload integrated into the complex attack process. The V4 and V5 versions of the DarkMe Trojan with complete code functions can be used both as a basic stealing tool and as a loader for other tools, so they are widely used by Evilnum attackers in recent attacks.

5.2 PikoloRAT

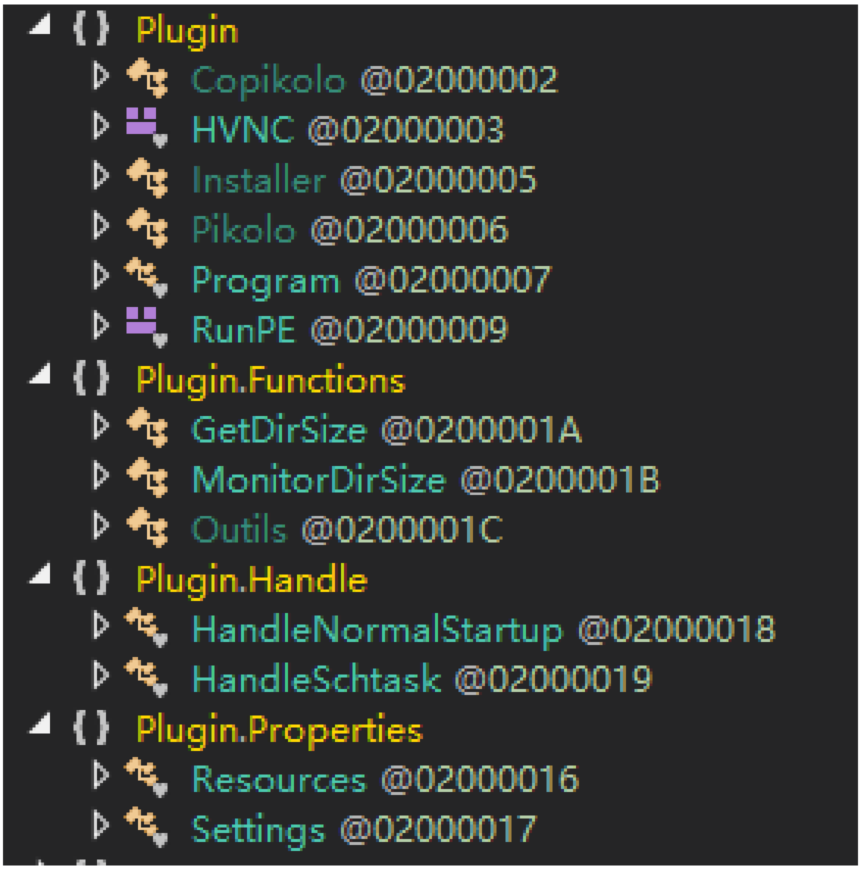
Through in-depth mining of the relevant information of this event, Fuying Lab discovered another new type of remote control Trojan, PikoloRAT. The Trojan comes with typical remote control functions and can use built-in components for more complex control operations.

Since the discovered built-in CnC address of the PikoloRAT Trojan coincides with the address used in this Evilnum operation, and its functions can complement the above-mentioned DarkMe Trojan, Fuying Lab judged that PikoloRAT was used by the Evilnum attackers in the later stage of the invasion Extension components.

In discovered cases, PikoloRAT was delivered via a downloader Trojan or packaged as a compressed file.

5.2.1 Function

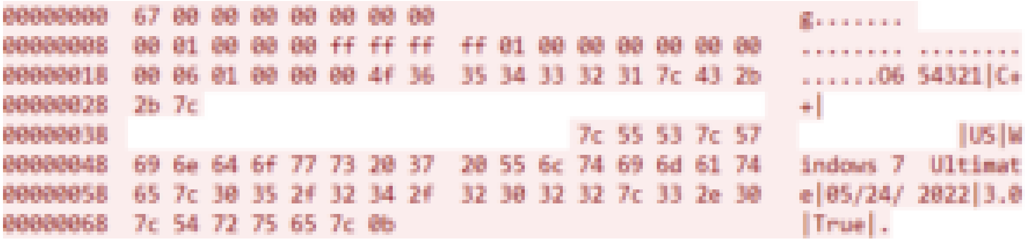
PikoloRAT is a typical RAT trojan program written in C#.



```
{ } Plugin
    Copikolo @02000002
    HVNC @02000003
    Installer @02000005
    Pikolo @02000006
    Program @02000007
    RunPE @02000009
{ } Plugin.Functions
    GetDirSize @0200001A
    MonitorDirSize @0200001B
    Outils @0200001C
{ } Plugin.Handle
    HandleNormalStartup @02000018
    HandleSchtask @02000019
{ } Plugin.Properties
    Resources @02000016
    Settings @02000017
```

Figure 5.4 PikoloRAT main frame

After the Trojan runs, it first collects and uploads the host information. The collected content includes Trojan mark, user name, computer name, geographic location, operating system version, Trojan running time, Trojan version, and anti-virus software information. The Trojan uses the separator "|" to separate the information, adds a fixed string "654321" at the front, and sends it to CnC.



```
00000000  67 00 00 00 00 00 00 00                            g.......
00000008  00 01 00 00 00 ff ff ff  ff 01 00 00 00 00 00 00   ........ ........
00000018  00 06 01 00 00 00 4f 36  35 34 33 32 31 7c 43 2b   ......O6 54321|C+
00000028  2b 7c                                              +|
00000038                                          7c 55 53 7c 57            |US|W
00000048  69 6e 64 6f 77 73 20 37  20 55 6c 74 69 6d 61 74   indows 7  Ultimat
00000058  65 7c 30 35 2f 32 34 2f  32 30 32 32 7c 33 2e 30   e|05/24/ 2022|3.0
00000068  7c 54 72 75 65 7c 0b                               |True|.
```

Figure 5.5 PikoloRAT online traffic

It can be seen that the content and format of this information are similar to the above-mentioned DarkMe Trojan.

Subsequently, PikoloRAT enters a controlled state, and controls the behavior of the host by obtaining CnC-side commands. The supported remote control operation commands are as follows:

script	operate
1	Exit the instruction loop
2	Left click pressed
3	right click press
4	Left click to lift
5	Right click to lift
6	left double click
7	Press the corresponding keyboard key
8	Move the mouse to the specified location
9	Get clipboard content

17	Set screenshot interval
18	Set screenshot quality
19	Set screenshot zoom size
twenty four	end its own process
55	Set temporary file path
4875	execute cmd command
4876	Execute the powershell command
8888	Load and run PEGASUS HVNC
8889	Uninstall the Trojan body
8890	Persistence, including adding self-starting items and scheduled tasks
8891	delete persistent content

Table 5.2 PikoloRAT command comparison table

It can be seen that in addition to the basic remote control functions, PikoloRAT can also perform more sophisticated remote control by releasing the built-in PEGASUS HVNC module, a recently leaked hVNC tool.

6. Technical and tactical analysis

6.1 Override side loading

In this attack process A, the Evilnum attacker delivered a malicious python39.dll file and loaded the malicious file through the legitimate file python.exe.

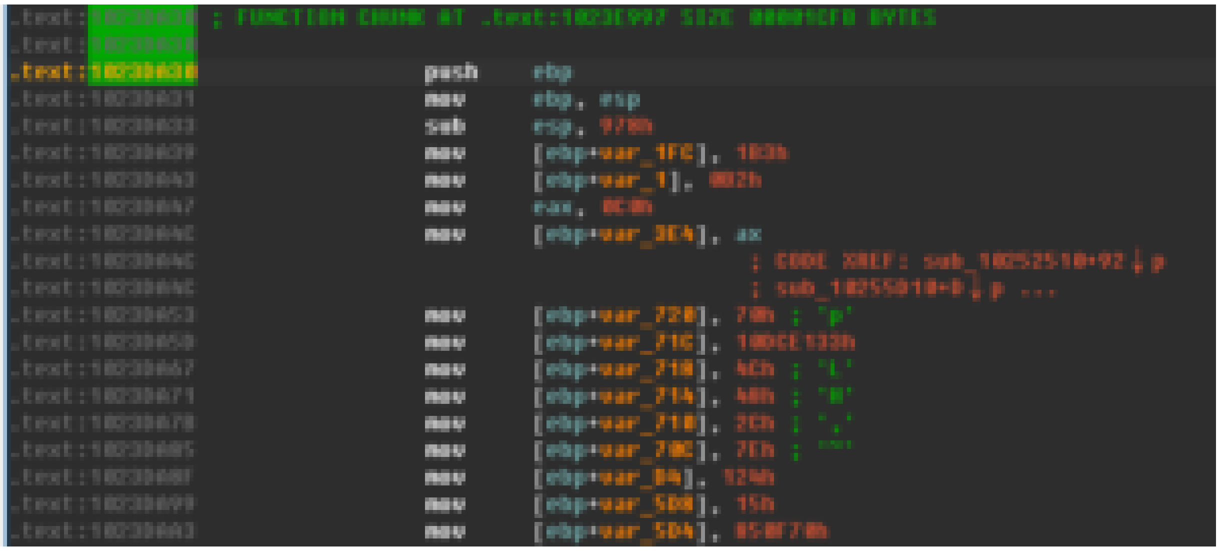
Unlike common sideloading build logic, this malicious python39.dll is actually obtained by directly overwriting the original python39.dll file. The Evilnum attacker directly writes a piece of shellcode to the location of the function PyImport_AddModuleObject of the original python39.dll, so that python39.dll automatically starts the shellcode when it is loaded.

The benefits of this design are:

Easy to operate, no need to compile a separate dll program and implement its export method;

Wide applicability, in theory, similar overwriting operations can be performed on any legal dll file to build a side-loading shellcode attack chain;

The concealment is strong, the dll file is very similar to the original dll file after overwriting, and it is not easy to be located.



```
.text:[illegible] ; FUNCTION CHUNK AT .text:[illegible] SIZE [illegible] BYTES
.text:[illegible]
.text:[illegible]                push    ebp
.text:[illegible]                mov     ebp, esp
.text:[illegible]                sub     esp, 970h
.text:[illegible]                mov     [ebp+var_1FC], [illegible]
.text:[illegible]                mov     [ebp+var_1], [illegible]
.text:[illegible]                mov     eax, [illegible]
.text:[illegible]                mov     [ebp+var_3E4], ax
.text:[illegible]                                        ; CODE XREF: sub_[illegible]
.text:[illegible]                                        ; sub_[illegible] ...
.text:[illegible]                mov     [ebp+var_720], 70h ; 'p'
.text:[illegible]                mov     [ebp+var_71C], [illegible]
.text:[illegible]                mov     [ebp+var_718], 4Ch ; 'L'
.text:[illegible]                mov     [ebp+var_714], 48h ; 'H'
.text:[illegible]                mov     [ebp+var_710], 2Ch ; ','
.text:[illegible]                mov     [ebp+var_70C], [illegible] ; [illegible]
.text:[illegible]                mov     [ebp+var_84], [illegible]
.text:[illegible]                mov     [ebp+var_5D8], 15h
.text:[illegible]                mov     [ebp+var_5D4], [illegible]
```

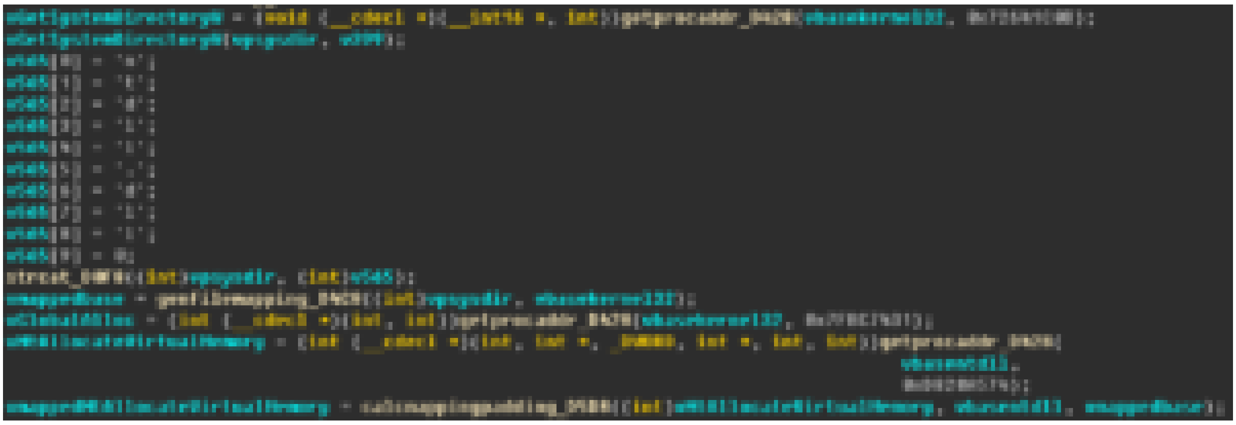
Figure 6.1 The overwritten PyImport_AddModuleObject function in python39.dll

6.2 Shellcode Framework

In this attack flow A, the Evilnum attacker used various shellcodes at different stages. These shellcodes have similar code implementation logic, and it can be seen that they come from the same shellcode programming framework, and the overall composition and code complexity have improved compared to previous Evilnum activities.

6.2.1 ntdll mapping

In the shellcode that appeared in this operation, the attacker still insisted on using two modules, kernel32 and ntdll, to build the main process. In order to avoid the api detection idea for such behavior, the attacker uses the following method to map the ntdll file and use the api of the mapped file:



```
vGetSystemDirectoryW = (void (__cdecl *)(__int16 *, int))getprocaddr_0420(vbasekernel32, 0x[illegible]);
vGetSystemDirectoryW(vpsysdir, 0x104);
v545[0] = 'n';
v545[1] = 't';
v545[2] = 'd';
v545[3] = 'l';
v545[4] = 'l';
v545[5] = '.';
v545[6] = 'd';
v545[7] = 'l';
v545[8] = 'l';
v545[9] = 0;
strcat_04F0((int)vpsysdir, (int)v545);
vmappedbase = getfilemapping_04C0((int)vpsysdir, vbasekernel32);
vGlobalAlloc = (int (__cdecl *)(int, int))getprocaddr_0420(vbasekernel32, 0x[illegible]);
vNtAllocateVirtualMemory = (int (__cdecl *)(int, int *, _DWORD, int *, int, int))getprocaddr_0420(
                                                                                   vbasentdll,
                                                                                   0x[illegible]);
vmappedNtAllocateVirtualMemory = calcmappingadding_0580((int)vNtAllocateVirtualMemory, vbasentdll, vmappedbase);
```

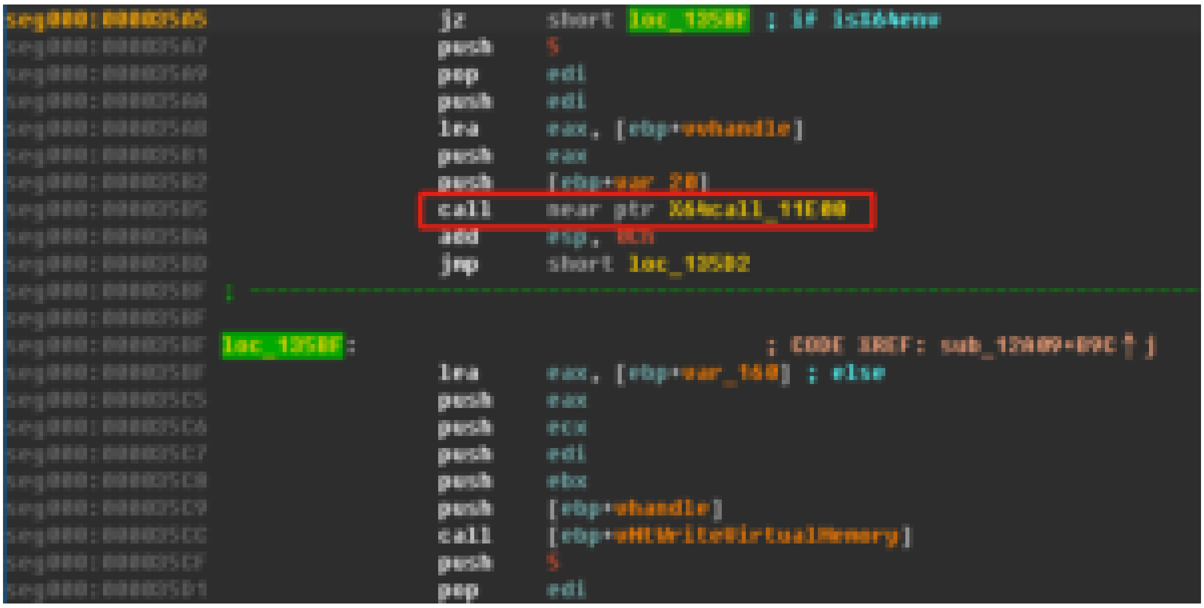
Figure 6.2 ntdll module mapping logic in Shellcode

In this implementation, the attacker reloads the ntdll module through file mapping, and records the corresponding api base address in the mapping module by calculating the offset after obtaining the api base address in the original ntdll. In the subsequent key process, the shellcode uses the mapping api to implement the corresponding behavior, so as to avoid the api call behavior and key parameters from being monitored and recorded.

6.2.2 X64call

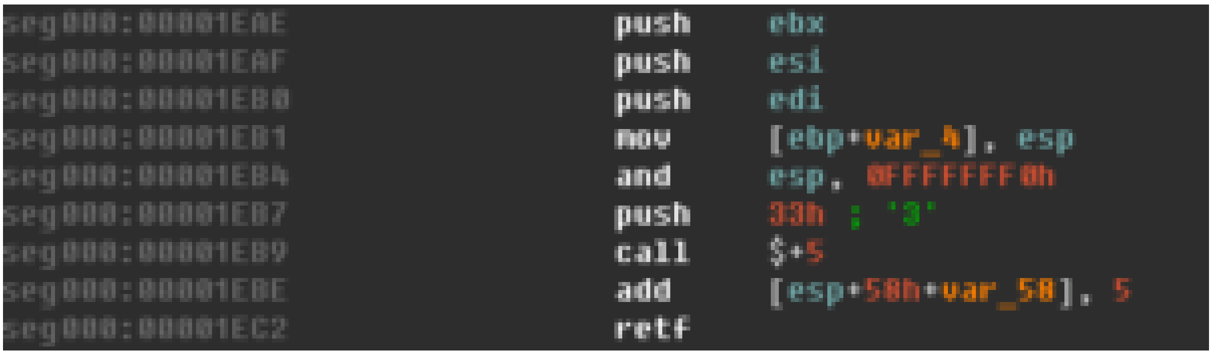
In this attack process A, the attacker used X64call to call key APIs in the operation of injecting cmd.exe.

The shellcode of the injection part first detects the process environment and the host cpu model. If the requirements are met, the 64-bit implementation is called when key injection apis such as NtAllocateVirtualMemory and NtWriteVirtualMemory are used:



```
seg000:000035A5                 jz      short loc_135BF ; if isX64env
seg000:000035A7                 push    5
seg000:000035A9                 pop     edi
seg000:000035AA                 push    edi
seg000:000035AB                 lea     eax, [ebp+vxhandle]
seg000:000035B1                 push    eax
seg000:000035B2                 push    [ebp+var_20]
seg000:000035B5                 call    near ptr X64call_11E00
seg000:000035BA                 add     esp, 0Ch
seg000:000035BD                 jmp     short loc_135D2
seg000:000035BF ; ---------------------------------------------------------------------------
seg000:000035BF
seg000:000035BF loc_135BF:                              ; CODE XREF: sub_12A09+B9C↑j
seg000:000035BF                 lea     eax, [ebp+var_160] ; else
seg000:000035C5                 push    eax
seg000:000035C6                 push    ecx
seg000:000035C7                 push    edi
seg000:000035C8                 push    ebx
seg000:000035C9                 push    [ebp+vhandle]
seg000:000035CC                 call    [ebp+vNtWriteVirtualMemory]
seg000:000035CF                 push    5
seg000:000035D1                 pop     edi
```

Figure 6.3 X64call calling logic in Shellcode injection code



```
seg000:00001EAE                 push    ebx
seg000:00001EAF                 push    esi
seg000:00001EB0                 push    edi
seg000:00001EB1                 mov     [ebp+var_4], esp
seg000:00001EB4                 and     esp, 0FFFFFFF0h
seg000:00001EB7                 push    33h ; '3'
seg000:00001EB9                 call    $+5
seg000:00001EBE                 add     [esp+58h+var_58], 5
seg000:00001EC2                 retf
```

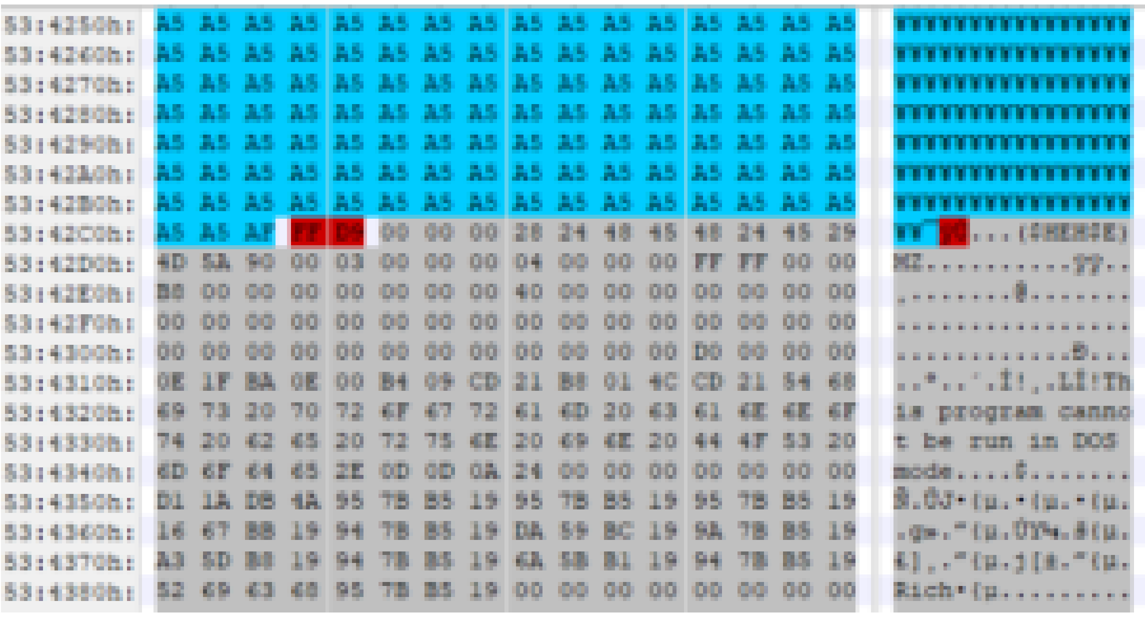
Figure 6.4 X64call calling code

This technique can also have the effect of avoiding api detection.

6.3 Image Steganography

In this operation, the Evilnum attackers used two forms of steganographic images.

In process B, the image named IMG.jpg uses redundant steganography, stores the malicious code at the end of the file, and uses a fixed string (HEHE) as the separation flag:



```
53:4250h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:4260h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:4270h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:4280h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:4290h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:42A0h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:42B0h: A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5  ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
53:42C0h: A5 A5 AF FF D9 00 00 00 28 24 48 45 48 24 45 29  ¥¥¯ÿÙ...($HEH$E)
53:42D0h: 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
53:42E0h: B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ¸.......@.......
53:42F0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
53:4300h: 00 00 00 00 00 00 00 00 00 00 00 00 D0 00 00 00  ............Ð...
53:4310h: 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..º..´.Í!¸.LÍ!Th
53:4320h: 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
53:4330h: 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS 
53:4340h: 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
53:4350h: D1 1A DB 4A 95 7B B5 19 95 7B B5 19 95 7B B5 19  Ñ.ÛJ•{µ.•{µ.•{µ.
53:4360h: 16 67 BB 19 94 7B B5 19 DA 59 BC 19 9A 7B B5 19  .g».”{µ.ÚY¼.š{µ.
53:4370h: A3 5D B8 19 94 7B B5 19 6A 5B B1 19 94 7B B5 19  £]¸.”{µ.j[±.”{µ.
53:4380h: 52 69 63 68 95 7B B5 19 00 00 00 00 00 00 00 00  Rich•{µ.........
```

Figure 6.5 Steganographic information in IMG.jpg

In process A, the image carrying the payload uses RGB value steganography, and the malicious code is stored in the R bit of the RGB value:

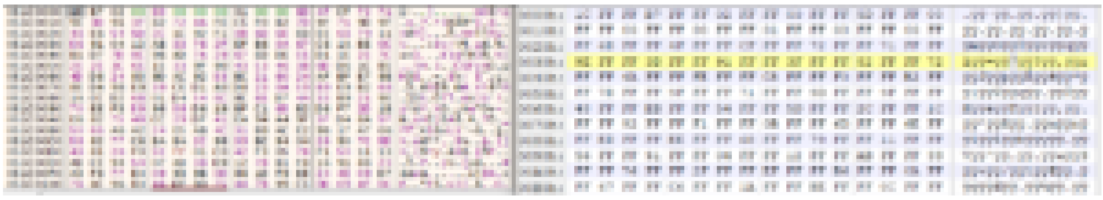


Figure 6.6 RGB values in the steganographic image sKr93l.png (right) and the extracted compressed data content (left)

Such a construction makes the steganographic image show blue-green splotches in white areas and red splotches in black areas:

Figure 6.7 Appearance of steganographic image sKr93l.png

Figure 6.8 Appearance of steganographic image Fruit.png

6.4 Socket window

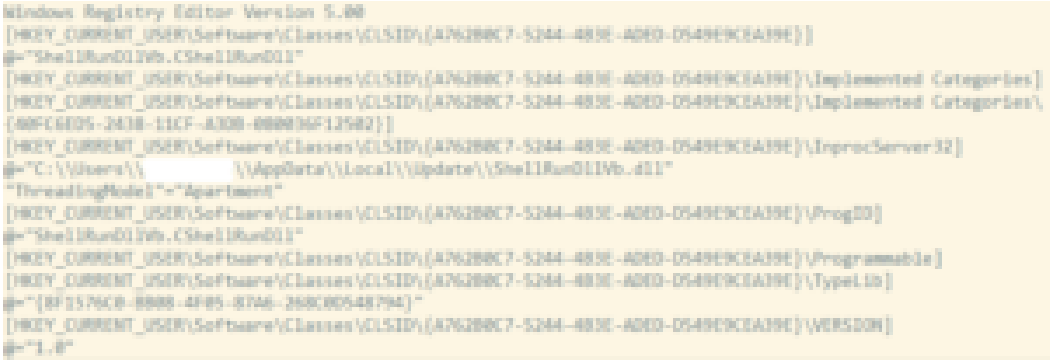
In this DarkCasino operation, the DarkMe Trojan used by Evilnum used SOCKET_WINDOW communication.

This is an ancient VisualBasic socket programming technique that hooks winsock messages through a SOCKET_WINDOW window and handles event messages passed by WSAAsyncSelect in the window callback function.

The original framework can be referred to: https://github.com/dzzie/RE_Plugins/blob/master/IdaVbScript/vb%20src/MSocketSupport.bas

6.5 COM component execution

In this DarkCasino operation, some DarkMe Trojans were delivered in the form of COM components. The Evilnum attacker writes registry manipulation logic in the preloaded Trojan payload, allowing it to generate and execute a Register.reg file with the following content:



```
Windows Registry Editor Version 5.00
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}]
@="ShellRunDllVb.CShellRunDll"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\Implemented Categories]
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\Implemented Categories\
{40FC6ED5-2438-11CF-A3DB-080036F12502}]
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\InprocServer32]
@="C:\\Users\\[redacted]\\AppData\\Local\\Update\\ShellRunDllVb.dll"
"ThreadingModel"="Apartment"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\ProgID]
@="ShellRunDllVb.CShellRunDll"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\Programmable]
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\TypeLib]
@="{BF1576C0-8808-4F05-87A6-268C0D548794}"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{A76280C7-5244-483E-ADED-D549E9CEA39E}\VERSION]
@="1.0"
```

Figure 6.9 Register.reg file content

Then the preloaded trojan load starts the DarkMe trojan through the cmd command in the form of rundll /sta [CLSID] 'Hello'.

This method avoids direct calls to the DarkMe Trojan, reducing exposure risks to a certain extent.

7. Summary

Operation DarkCasino is an ongoing APT attack on the cash flow of online transactions. Evilnum used a variety of ever-improving attack techniques and tools in this operation, showing its keen sense of confrontation.

Analysis shows that the scope of the DarkCasino operation is not limited to Europe. Under the operation of the Evilnum attackers, this attack eventually radiated to some Asian countries, which may cause unexpected harm.

In order to effectively prevent this APT operation, special attention should be paid to the LNK, PIF, SCR, COM type files transmitted through various channels, and the vigilance of files with keywords such as offer, visa, and casino should be raised to avoid Evilnum's network. Attacks cause direct economic losses.

8. IoCs

Attack Process A Decoy File

43eda4ff53eef4513716a5b773e6798653ee29544b44a9ae16aa7af160a996f2	offer deal visa 2022.lnk

Attack Process B Decoy File

5fb252474237a4ca96cc0433451c7d7a847732305d95ceeaeb10693ecef2eeee	Scatters Casino offers Daily Promotions.pif
8e4a4c5e04ff7ebacb5fe8ff6b27129c13e91a1acc829dbb3001110c84dc8633	new casino crypto.com
d0899cb4b94e66cb8623e823887d87aa7561db0e9cf4028ae3f46a7b599692b9	Promo CPL CPA Traffic.com

Attack Process C Decoy File

4ffa29dead7f6f7752f2f3b0a83f936f270826d2711a599233dc97e442dee85f	333TER.exe
9cf7f8a93c409dd61d019ca92d8bc43cc9949e244c9080feba5bfc7aac673ac3	d33v3TER.exe
259cebed2cd89da395df2a3588fadde82cd6542bc9ff456890f7ee2087dc43c9	d333TER.exe
0cdf27bb8c0c90fc1d60fb07bd30b7e97b16d15e3f58fb985350091ecad51ba6	ed333TER.exe
5ba84191a873d823ccf336adfa219cc191a004e22b56b99c6d0e1642144129b8	wed333TER.exe
15a076c7bb6a38425d96aa08b8a15e9a838c9697d57c835aaca92fd01607b07a	PayRedeemUpdateIntegration190520
3329f5e3a67d13bd602dca5bbe8e2d0b5d3b5cb7cb308965fb2599a66668c207	offer crypto casino.scr
8a49a7f6c95fade72ef86455794cdedfca9129aa0f5281e09929dfebfb3417c4	DOCUMENTATION AGREEMENTS S

CONSULTING INTEGRATION.pif

Downloader Trojan

864dccbeda7d88cad91336b5ae9efd50972508d1d8044226e798d039a0bc1da2	AONNRJP.exe

PikoloRAT Trojan

eb5e42c726c7b125564455d56a02b9d42672ca061575ff911672b9165e8e309d	stub1.exe
be544a1f9f642bb35a9bd0942ae16a7a6e58a323d298a408a00fa4c948e8ea17	Stub1.exe

DarkMe Trojan

a826570f878def28b027f6e6b2fcd8be1727e82666f8b65175d917144f5d0569	Project1.exe
7b478cd8b854c9046f45f32616e1b0cbdc9436fa078ceddb13ce9891b24b30a5	Project1.exe
e72337c08d6b884b64fd9945c5a01557ccf40db93af866c00c48d36b6605f3a0	Project1.exe
414a11e8eabb64add97a866502edcd7e54108bd247f4ae12fe07feeae4e549f6	Project3.exe
7913cdf40cc17a28487a71ab0d7724b8bf3646a2a53e3905798ce23a657061b8	Project1.exe
3a6694567e9d722357b8e92153d9c878bbcab55a2f65cd0f9a2e6579fbeb935a	Project3.exe
a6a70c85b8c40932678c413fde202a55fcfc9d9cae23822708be5f28f9d5b6d2	Project3.exe
c50ebe13972e6e378248d80d53478d8e01e754c5d87113d9b6f93bf3b84380b4	Project1.exe
1ac7715b1762788b5dc1f5f2fc35243a072fe77053df46101ce05413cca62666	Project3.exe
4ecc2925cfb07332331461 1a3892d476a58ff2f6b510b434996686e2f0ac3af7	Project3.exe
541b3011953a3ce1a3a4a22c8c4f58c6a01df786a7cc10858649f8f70ee0a2f3	Project3.exe
f25cbc53d0cc14b715ee83e51946d5793e4e86e71e96f68e9b6c839b514e8cb8	Project3.exe
4244f274a12f4672f2dda1190559d96c5a9631c9ee573b853c89e30701819b63	Project24.pif
1f0d908c677fb3ec5b9422eb5f7d2a2b3ffa01659521afc07cc4dfaea27aa532	Nuovo.pif
028057e54a2e813787a14b7d33e6a2caa91485ed879ef1bbcb94df0e1cf91356	bvo.exe
0a9c183f0b5a225228da5e8589fac8b3affe2e51c790a08148ef72481de610c4	bvo.exe
3eb84676249cb26dd3d1962cfca2a9fde442d0feaa1b0351f6331313f3ac1138	bvo.exe
46fbfc263959084d03bd72c5b6ee643711f79f7d76b391d4a81f95b2d111b44e	bvofinal.pif
5e04dd49b82320eca63b483e87453d2a68a9f4873f47d37e5080d537bc811d0e	pppppesst.exe
dc8190279dcea4f9a36208ba48b14e6c8313ef061252027ef8110b2d0bd84640	pppppesst.exe
4959cdba7edee68b5116cc1b8ef5016978d3dff2016f027a4f76b080b7c3849a	faster.exe
24ace8fd73b2a5a13f3e5b459f0764dd4b5bda2cea2b0e13bbf88a88afe0cdac	fastest.exe
c66e6ee55e9799a8a32b7a2c836c26bb7ebea98d09c1535ad9ae59e9628835fb	fastest.exe
32ce8d0dcbfcc2517480d0e08f8896ab4f6ea13ccb0eefe7205cd352c7b359c3	h5a.exe
c192684d296ea587e93457d060cbef900143cf1a11301e6c2e34e264e3e55ef6	h5a.exe
1d01b143a56eba431387b9b973790d174deb48c2e3445d96b131a7d8e0a9d4ef	vvt1.exe
b8ba2c0478649dc099d0a869755a7e205173a9b0d15fad920317a89d07eaa930	vvt1.exe
d95853e6e16d90c00fd72aaeaca9885b953dae14d7d6aa7fedcc6150fb788667	656.exe
7add6700c6e1aa1ac8782fdd26a11283d513302c672e3d62f787572d8ad97a21	ShellRunDllVb.dll
17fe047b9a3695d4fd8ad9d2f7f37486c0bc85db0f9770471442d31410ff26a1	ShellRunDllVb.dll
2665a09ec5b4ca913f9f3185df62495f13611831dba9073779a36df088db143b	ShellRunDllVb.dll
7c06a03d712be8c0df410bea5d1c2004c6247bcde5a46ce51746f18de9621ac1	ShellRunDllVb.dll

URL

https[:]//puccino.altervista.org/wp-content/uploads/2022/05/6h.txt
https[:]//storangefilecloud.vip/IMG.jpg
https[:]//storangefilecloud.vip/PI.txt
https[:]//storangefilecloud.vip/PRGx.jpg
https[:]//bukjut11.com/FRIGO.JPG
https[:]//bukjut11.com:443/AEVC.JPG
https[:]//imagizer.imageshack.com/img922/1527/sKr93I.png
https[:]//imagizer.imageshack.com/img923/7651/jMwIGI.png
https[:]//i.imgur.com/fkNiY9Z.png
https[:]//laurentprotector.com/LRGBPFV.bin
https[:]//laurentprotector.com/NnQFqsOEUtkezvIEcLpfa.bin

Darkme CnC

aka7newmalp23.com
csmmmsp099q.com
muasaashshaj.com
cspapop110.com
938jss.com
8as1s2.com
kalpoipolpmi.net
pallomnareraebrazo.com
185.236.231.74

PikoloRAT CnC

51.195.57.232

Copyright Notice

The copyright holder of all contents of the "Technology Blog" on this site is NSFOCUS Technology Group Co., Ltd. ("NSFOCUS Technology"). As a platform for sharing technical information, NSFOCUS looks forward to interacting with the majority of users, and welcomes the full text to be forwarded with the source (NSFOCUS-Technical Blog) and website indicated.
For any form of use other than the above, it is necessary to apply for copyright authorization to NSFOCUS (010-68438880-5462) in advance. For unauthorized use, NSFOCUS reserves the right to pursue responsibility. At the same time, if a legal dispute arises due to the unauthorized use of blog content, the user shall bear all legal responsibilities and has nothing to do with NSFOCUS.