VB2020 localhost

30 September - 2 October, 2020 / vblocalhost.com

OPERATION LAGTIME IT: COLOURFUL PANDA FOOTPRINT

Fumio Ozawa, Shogo Hayashi & Rintaro Koike

NTT Security (Japan) KK

fumio.ozawa@global.ntt
syogo.hayashi@global.ntt
rintaro.koike@global.ntt

ABSTRACT

Operation LagTime IT by TA428 is an attack campaign targeting governmental organizations of East Asian countries, reported by *Proofpoint* in July 2019. It is still in the wild and active as of 2020. Through detailed research on two samples (document files on Qasem Soleimani and COVID-19) observed in January and February 2020, we have successfully unveiled and determined the whole attack picture, including how TA428 interacts with a target. Previous research on Operation LagTime IT only reported that it used the Royal Road RTF Weaponizer, Poison Ivy and Cotx RAT. However, according to the behaviour that we have observed, TA428 also performs user environment checking, credential stealing, lateral movement and highly sophisticated defence evasion.

In this paper we describe the operational steps that TA428 has taken from initial samples to reach the deepest part of the victim's system. We also reveal our analysis of the malware used by TA428 and the codes that decode encrypted communication. We also discuss how the techniques, tools and malware used in Operation LagTime IT overlap with those of various other APT actors.

INTRODUCTION

TA428

TA428 is an advanced persistent threat (APT) actor that mainly targets East Asia. TA428 is known as a Chinese APT actor, and its most recent attack campaign is called Operation LagTime IT. It's considered that the actor is related to Pirate Panda [1], Tropic Trooper and Key Boy.

Operation LagTime IT

Operation LagTime IT is an attack campaign operated by TA428 around March 2019. *Proofpoint* has reported [2] that the group used Poison Ivy and Cotx RAT to target government agencies in East Asia. It has been reported that an RTF file generated by a tool called 'Royal Road RTF Weaponizer' [3], which is related to Tick and Tonto, is used as a lure document for the attacks.

Our motivation

Similar to Tick and Tonto, TA428 is attacking East Asia using the 'Royal Road RTF Weaponizer'. However, detailed attack analysis of TA428 has not been shared to date. We wanted to find out the details of TA428's attack strategy in order to help defend against it, in particular what kind of breaches the group uses with Poison Ivy and Cotx RAT. Therefore, we focused on Operation LagTime IT, which is one of TA428's most active attack campaigns, and we observed and analysed the attack.

Since 2020, we have observed Operation LagTime IT attacks five times. We performed a detailed analysis of the two attacks we observed in January and February. As a result, we have been able to uncover several pieces of malware and compromise tools that have never before been reported, as well as the attacker's specific method of operation.

CASE 1

Overview and attack flow

In early January 2020, we observed a file called 'How Suleimani's death will affect India and Pakistan.doc'. This file is a lure document that is the launch point for Operation LagTime IT. When this file is opened in a vulnerable version of *Microsoft Office Word*, it will exploit the vulnerability and create a file called 'useless.wll' in the *Microsoft Word* startup directory.

The .wll file located in the *Microsoft Office Word* startup directory will automatically be loaded and executed when the user starts *Word*.

The file named 'useless.wll' is Poison Ivy. It is used to download three cab files ('o.cab', 'nbt.cab' and 'in.cab') from the C&C server, and execute the files stored in the cab files. The file 'o.cab' contains 'o.exe', which is a tool used to dump *Outlook* credentials. The file 'nbt.cab' contains 'n.exe', which is an NTB scan tool. The file 'in.cab' contains a file named 'intel.dll', which will be executed from 'rundll32.exe' by the attacker later on.

The file 'intel.dll' creates two files, 'intel.exe' and 'RasTls.dll'. 'Intel.exe' is a legitimate *Intel* executable file that is digitally signed. 'Intel.exe' is used to perform DLL side-loading as the executable will load 'RasTls.dll', located in the same directory. 'RasTls.dll' is a Cotx RAT.

After executing 'o.exe' and 'n.exe' and persisting the Cotx RAT, the operation by the attacker stopped.

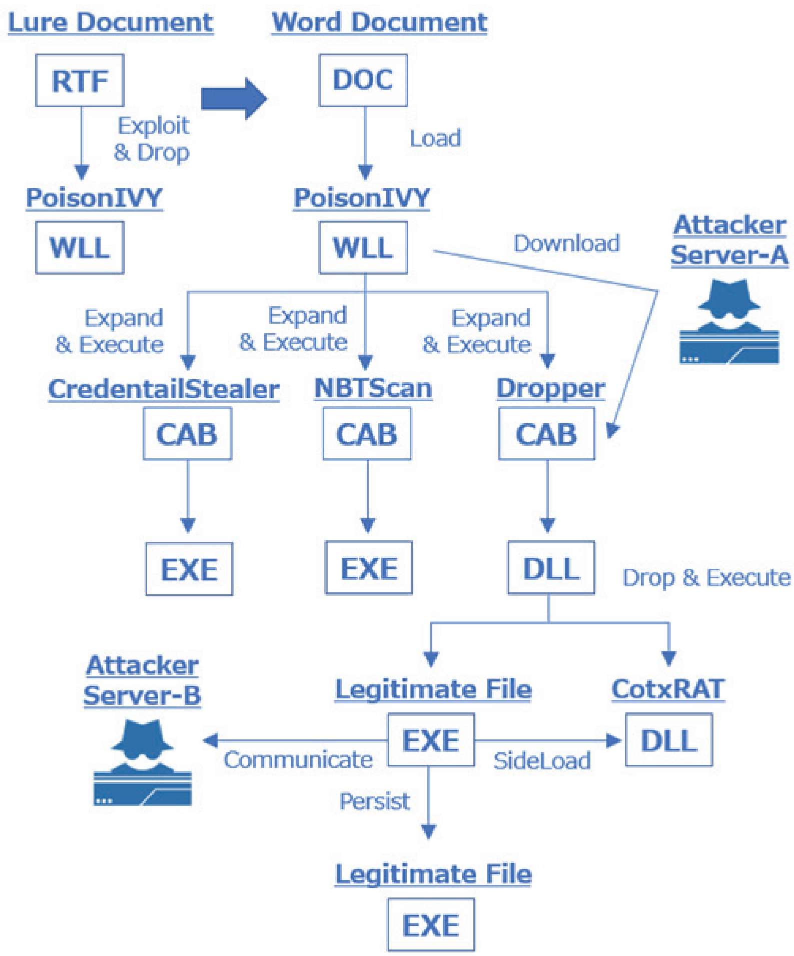


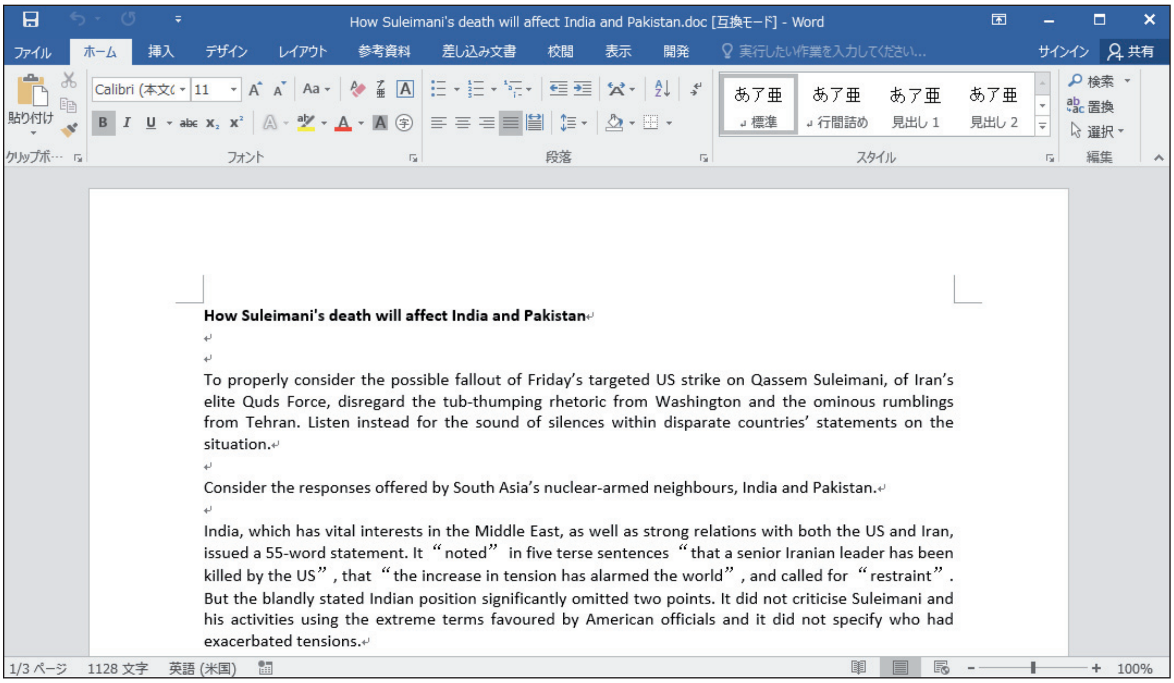
Figure 1: Whole picture of attack Case 1.

Item	**File path**	**Description**
Lure document	How Suleimani's death will affect India and Pakistan.doc	RTF file that attacker sends
Word document	-	Any Microsoft Office Word file
Poison Ivy	%APPDATA%\Microsoft\Word\STARTUP\useless.wll	Poison Ivy RAT
Credential stealer	%USERPROFILE%\AppData\Local\Comms\o.cab %USERPROFILE%\AppData\Local\Comms\o.exe	Dump tool for Outlook credentials
NBTScan	%APPDATA%\Adobe\nbt.cab %APPDATA%\Adobe\n.exe	NBT scan tool
Dropper	%ALLUSERSPROFILE%\Comms\in.cab %ALLUSERSPROFILE%\Comms\intel.dll	Dropper of Cotx RAT
Legitimate file	%ALLUSERSPROFILE%\Comms\intel.exe %APPDATA%\Intel\Intel(R) Processor Graphics\ IntelGraphicsController.exe	Legitimate executable file of Intel Corporation
Cotx RAT	%ALLUSERSPROFILE%\Comms\RasTls.dll	Cotx RAT is side-loaded by intel.exe

Table 1: Malware and files observed during attack Case 1.

Lure document

'How Suleimani's death will affect India and Pakistan.doc' is an RTF file relating to the death of Commander Soleimani of the Islamic Revolutionary Guard.


How Suleimani's death will affect India and Pakistan

To properly consider the possible fallout of Friday's targeted US strike on Qassem Suleimani, of Iran's elite Quds Force, disregard the tub-thumping rhetoric from Washington and the ominous rumblings from Tehran. Listen instead for the sound of silences within disparate countries' statements on the situation.

Consider the responses offered by South Asia's nuclear-armed neighbours, India and Pakistan.

India, which has vital interests in the Middle East, as well as strong relations with both the US and Iran, issued a 55-word statement. It "noted" in five terse sentences "that a senior Iranian leader has been killed by the US", that "the increase in tension has alarmed the world", and called for "restraint". But the blandly stated Indian position significantly omitted two points. It did not criticise Suleimani and his activities using the extreme terms favoured by American officials and it did not specify who had exacerbated tensions.

Figure 2: 'How Suleimani's death will affect India and Pakistan.doc'.

This RTF file contains malicious code for exploiting CVE-2018-0798 and an object called '8.t'. The inclusion of these objects suggests that it was created using the Royal Road RTF Weaponizer.

```
$ rtfobj "How Suleimani's death will affect India and Pakistan.doc"
rtfobj 0.54 on Python 2.7.15 - http://decalage.info/python/oletools
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues

===============================================================================
File: "How Suleimani's death will affect India and Pakistan.doc" - size: 112050 bytes
---+----------+---------------------------------------------------------------
id |index     |OLE Object
---+----------+---------------------------------------------------------------
0  |0000A105h |format_id: 2 (Embedded)
   |          |class name: 'Package'
   |          |data size: 28360
   |          |OLE Package object:
   |          |Filename: u'8.t'
   |          |Source path: u'C:\\Aaa\\tmp\\8.t'
   |          |Temp path = u'C:\\Users\\ADMINI~1\\AppData\\Local\\Temp\\8.t'
   |          |MD5 = '128d63a9141545fe9cd70ef2f7b68a04'
---+----------+---------------------------------------------------------------
1  |00017F25h |format_id: 2 (Embedded)
   |          |class name: 'Equation.2\x00\x124Vx\x90\x124VxvT2'
   |          |data size: 6436
   |          |MD5 = 'cedca7a7d475b12161359cd54d054433'
---+----------+---------------------------------------------------------------
2  |00017F0Bh |Not a well-formed OLE object
---+----------+---------------------------------------------------------------
```

Figure 3: Objects included in the RTF file.

When this RTF file is opened with *Microsoft Word*, it will load the malicious code that exploits CVE-2018-0798 and execute the two-byte XOR-encoded shellcode.

```
0x0000006e      inc     edx
0x00000070      pop     rdi
0x00000071      add     edi, 0x1a
0x00000074      xor     ecx, ecx
0x00000076      mov     cx, 0x8ba
0x0000007a      cmp     word [rdi], 0
0x0000007e      je      0x85
0x00000080      xor     word [rdi], 0xc390
0x00000085      loop    0x7a
0x00000089      jns     0xad
0x0000008b      xchg    eax, edx
0x0000008c      ret
```

Figure 4: Decoding shellcode.

The shellcode decodes the 8.t object by the following operations:

```
0x00000453      mov     eax, 0x48b53a6c
0x00000458      xor     edx, edx
0x0000045a      test    ebx, ebx
0x0000045c      jle     0x48e
0x0000045e      mov     esi, ebx
0x00000460      push    7
0x00000462      pop     rbx
0x00000463      mov     ecx, eax
0x00000465      shr     ecx, 0x1a
0x00000468      xor     ecx, eax
0x0000046a      shr     ecx, 3
0x0000046d      xor     ecx, eax
0x0000046f      add     eax, eax
0x00000471      and     ecx, 1
0x00000474      or      eax, ecx
0x00000476      jne     0x463
0x0000047a      mov     ecx, dword [rbp - 0xc]
0x0000047d      xor     byte [rdx + rcx], al
0x00000480      cmp     edx, esi
0x00000483      jl      0x460
0x00000485      mov     ebx, dword [rbp - 4]
0x00000488      lea     esi, [rdi + 0x2a5]
0x0000048e      xor     eax, eax
```

Figure 5: Decoding the 8.t object.

The result of decoding 8.t is a DLL file that will be written to the *Microsoft Office Word* startup directory with the file name 'useless.wll'.

Poison Ivy

The useless.wll file created in the *Microsoft Office Word* startup directory will automatically be loaded and executed the next time *Microsoft Office Word* is started [4].

This .wll file will first check for the existence of the string 'WORD.EXE' in the result of GetCommandLineA by using the strstr function. If the string exists, it will execute again using rundll32.exe. This time, it will execute a function called 'DllEntry10', rather than 'DllEntryPoint'.

When DllEntry10 is executed, it first decodes some data with 'XOR 0xad'. One of the decoded strings is an RC4 key. The core part of Poison Ivy will be decoded using the RC4 key and additional simple operation.

The decoded data includes Poison Ivy's configuration data, which is shown in Table 2.

```
C645 BC 72    mov byte ptr ss:[ebp-44],72    72:'r'
C645 BD 75    mov byte ptr ss:[ebp-43],75    75:'u'
C645 BE 6E    mov byte ptr ss:[ebp-42],6E    6E:'n'
C645 BF 64    mov byte ptr ss:[ebp-41],64    64:'d'
C645 C0 6C    mov byte ptr ss:[ebp-40],6C    6C:'l'
C645 C1 6C    mov byte ptr ss:[ebp-3F],6C    6C:'l'
C645 C2 33    mov byte ptr ss:[ebp-3E],33    33:'3'
C645 C3 32    mov byte ptr ss:[ebp-3D],32    32:'2'
C645 C4 2E    mov byte ptr ss:[ebp-3C],2E    2E:'.'
C645 C5 65    mov byte ptr ss:[ebp-3B],65    65:'e'
C645 C6 78    mov byte ptr ss:[ebp-3A],78    78:'x'
C645 C7 65    mov byte ptr ss:[ebp-39],65    65:'e'
C645 C8 20    mov byte ptr ss:[ebp-38],20    20:' '
C645 C9 22    mov byte ptr ss:[ebp-37],22    22:'\"'
C645 CA 25    mov byte ptr ss:[ebp-36],25    25:'%'
C645 CB 73    mov byte ptr ss:[ebp-35],73    73:'s'
C645 CC 22    mov byte ptr ss:[ebp-34],22    22:'\"'
C645 CD 2C    mov byte ptr ss:[ebp-33],2C    2C:','
C645 CE 44    mov byte ptr ss:[ebp-32],44    44:'D'
C645 CF 6C    mov byte ptr ss:[ebp-31],6C    6C:'l'
C645 D0 6C    mov byte ptr ss:[ebp-30],6C    6C:'l'
C645 D1 45    mov byte ptr ss:[ebp-2F],45    45:'E'
C645 D2 6E    mov byte ptr ss:[ebp-2E],6E    6E:'n'
C645 D3 74    mov byte ptr ss:[ebp-2D],74    74:'t'
C645 D4 72    mov byte ptr ss:[ebp-2C],72    72:'r'
C645 D5 79    mov byte ptr ss:[ebp-2B],79    79:'y'
C645 D6 31    mov byte ptr ss:[ebp-2A],31    31:'1'
C645 D7 30    mov byte ptr ss:[ebp-29],30    30:'0'
C645 D8 00    mov byte ptr ss:[ebp-28],0
```

Figure 6: Executing the DllEntry10 function.

```
0x10001d90    mov eax, dword [var_42bch]
0x10001d96    add eax, 1
0x10001d99    mov dword [var_42bch], eax
0x10001d9f    cmp dword [var_42bch], 0x42ac
0x10001da9    jge 0x10001df9
0x10001dab    mov ecx, dword [var_42bch]
0x10001db1    movzx edx, byte [ecx + 0x10004010]
0x10001db8    not edx
0x10001dba    mov eax, dword [var_42bch]
0x10001dc0    mov byte [eax + 0x10004010], dl
0x10001dc6    mov ecx, dword [var_42bch]
0x10001dcc    movzx ecx, byte [ecx + 0x10004010]
0x10001dd3    mov eax, dword [var_42bch]
0x10001dd9    xor edx, edx
0x10001ddb    mov esi, 0x200      ; 512
0x10001de0    div esi
0x10001de2    movzx edx, byte [edx + 0x10008320]
0x10001de9    xor ecx, edx
0x10001deb    mov eax, dword [var_42bch]
0x10001df1    mov byte [eax + 0x10004010], cl
0x10001df7    jmp 0x10001d90
```

Figure 7: The simple operation.

C&C server	95.179.131.29:443
	95.179.131.29:8080
Campaign ID	hold
Group ID	hold
Mutex	99x7nmpWW
Password	3&U<9f*lZ>!MIQ

Table 2: Poison Ivy's configuration data.

This version of Poison Ivy has similar traffic characteristics to those of a variant called SPIVY [5]. The first byte of traffic is a value from 0x01 to 0x0f. It shows the size of the padding data that immediately follows it. When the padding data ends, double the padding data size follows to indicate the end. After that is the data body.

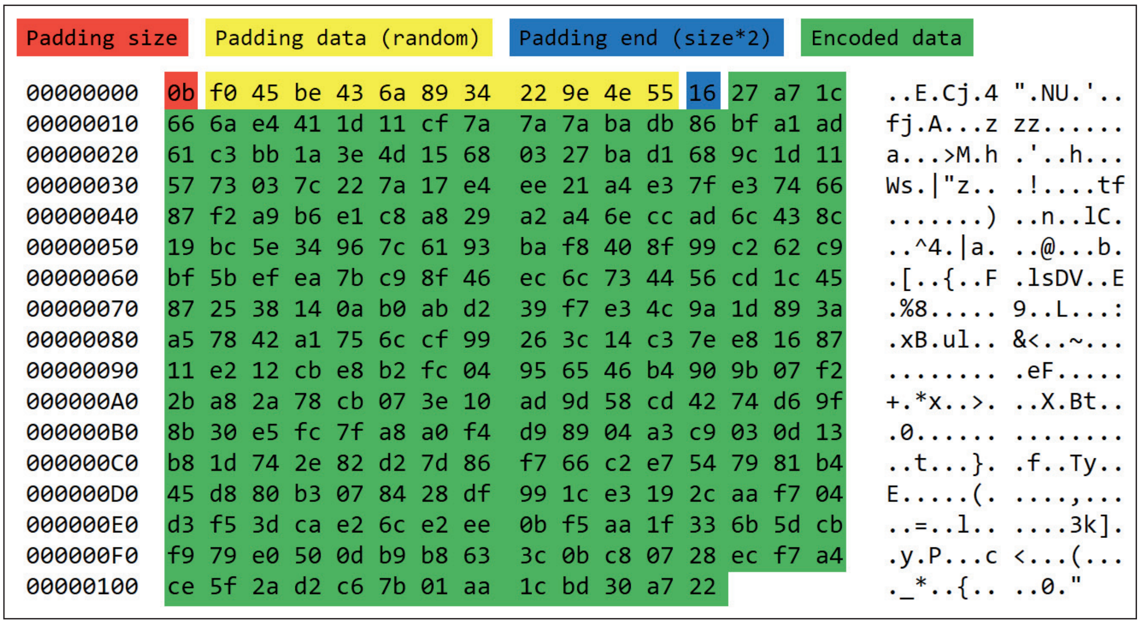


Figure 8: Traffic data structure.

The data is Camellia-encrypted using ECB mode. The encryption key is contained in the configuration data. The structure of the subsequent data is the same as in the normal Poison Ivy [6].

Cotx RAT

The Cotx RAT is the original RAT used by the TA428 group. The *Proofpoint* report [2] named it Cotx RAT because it saved the configuration data in the '.cotx' section. However, in the Cotx RAT that we analysed, the configuration data was included in the '.pdata' section.

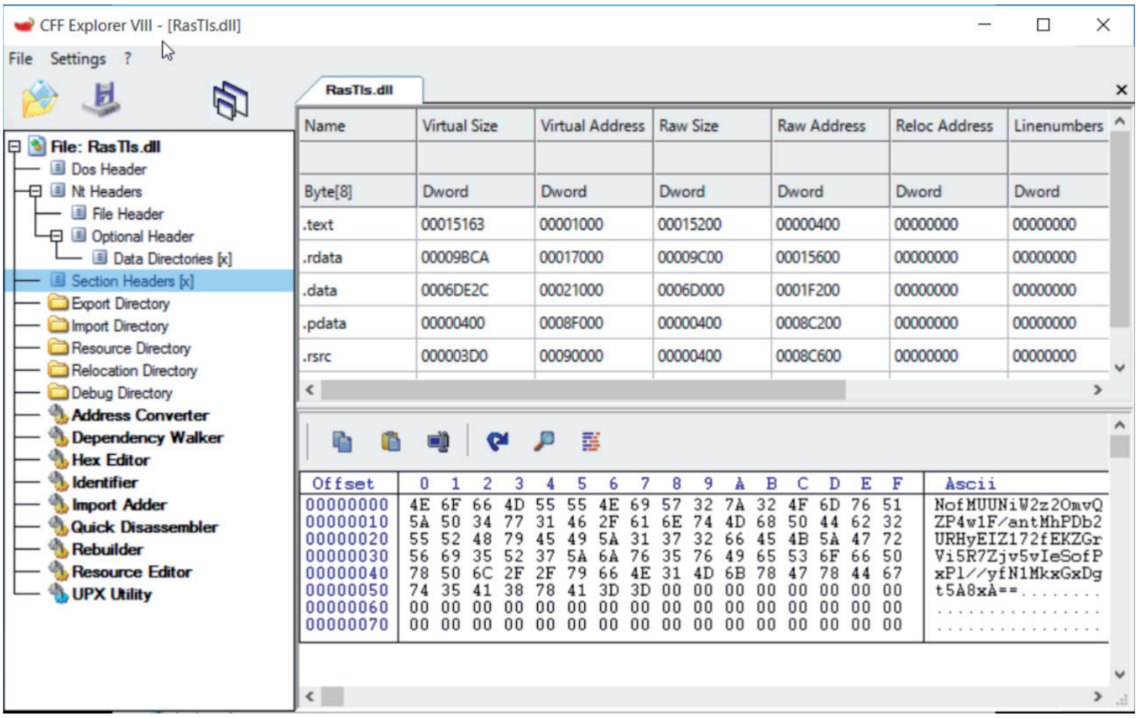


Figure 9: Configuration data in the '.pdata' section.

The configuration data is encrypted with AES-192 in CBC mode. The encryption key and initialization vector 'IV' are identical to those in the *Proofpoint* report.

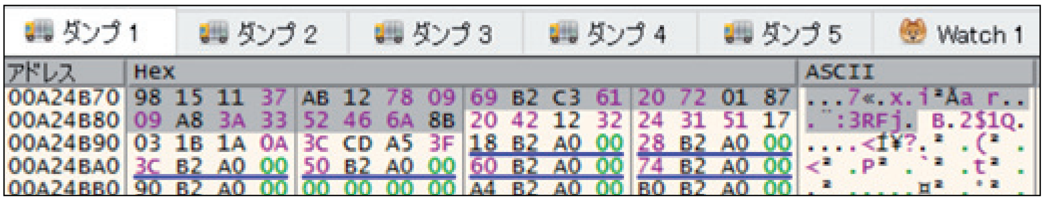


Figure 10: The key to decode configuration data.

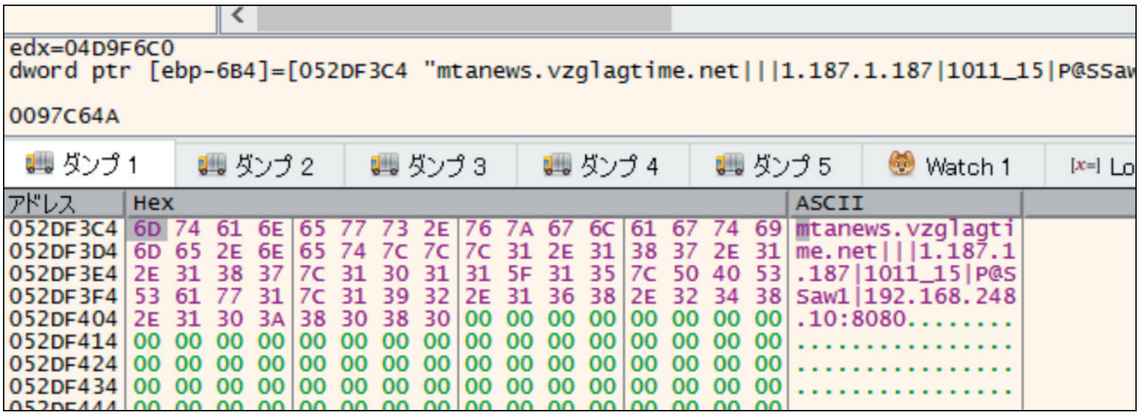


Figure 11: Decoding results of the configuration data.

This RAT functionality was also unchanged from the *Proofpoint* [2] report.

Credential stealer

The 'o.exe' file that was downloaded and executed by Poison Ivy is a commercial tool [7] called 'Outlook Password Dump v3.0'. When the tool is executed, it is possible to steal credentials stored in *Microsoft Office Outlook*. In the victim's environment, the attacker could not get anything because the credentials were not stored in *Outlook*.

```
$ o.exe


*****************************************************************

    Outlook Password Dump v3.0 by SecurityXploded

    http://securityxploded.com/outlook-password-dump.php

*****************************************************************



 Email Address                    Username              Password         Account Type     Email Server
 ================================================================================================================
```

Figure 12: An execution result of o.exe.

Environment scanner

The 'n.exe' file that was downloaded and executed by Poison Ivy is a public NBTScan tool [8]. When the tool is executed, it is possible to scan for hosts on the target network. The attacker was scanning neighbouring networks and looking for existing hosts.

```
$ n.exe
nbtscan 1.0.35 - 2008-04-08 - http://www.unixwiz.net/tools/

usage: n.exe [options] target [targets...]

   Targets are lists of IP addresses, DNS names, or address
   ranges. Ranges can be in /nbits notation ("192.168.12.0/24")
   or with a range in the last octet ("192.168.12.64-97")

   -V        show Version information
   -f        show Full NBT resource record responses (recommended)
   -H        generate HTTP headers
   -v        turn on more Verbose debugging
   -n        No looking up inverse names of IP addresses responding
   -p <n>    bind to UDP Port <n> (default=0)
   -m        include MAC address in response (implied by '-f')
   -T <n>    Timeout the no-responses in <n> seconds (default=2 secs)
   -w <n>    Wait <n> msecs after each write (default=10 ms)
   -t <n>    Try each address <n> tries (default=1)
   -1        Use Winsock 1 only
   -P        generate results in perl hashref format
```

Figure 13: Execution of n.exe.

CASE 2

Overview and attack flow

In the middle of February 2020, we observed a file called 'English_2020.02.17_13.00_MOH_daily update.doc'. This file looks like a document related to COVID-19. However, it is actually a lure document that is the starting point for the attack of Operation LagTime IT. As in Case 1, a .wll file is created ('woldfunc.wll') and copied to the *Microsoft Office Word* startup directory.

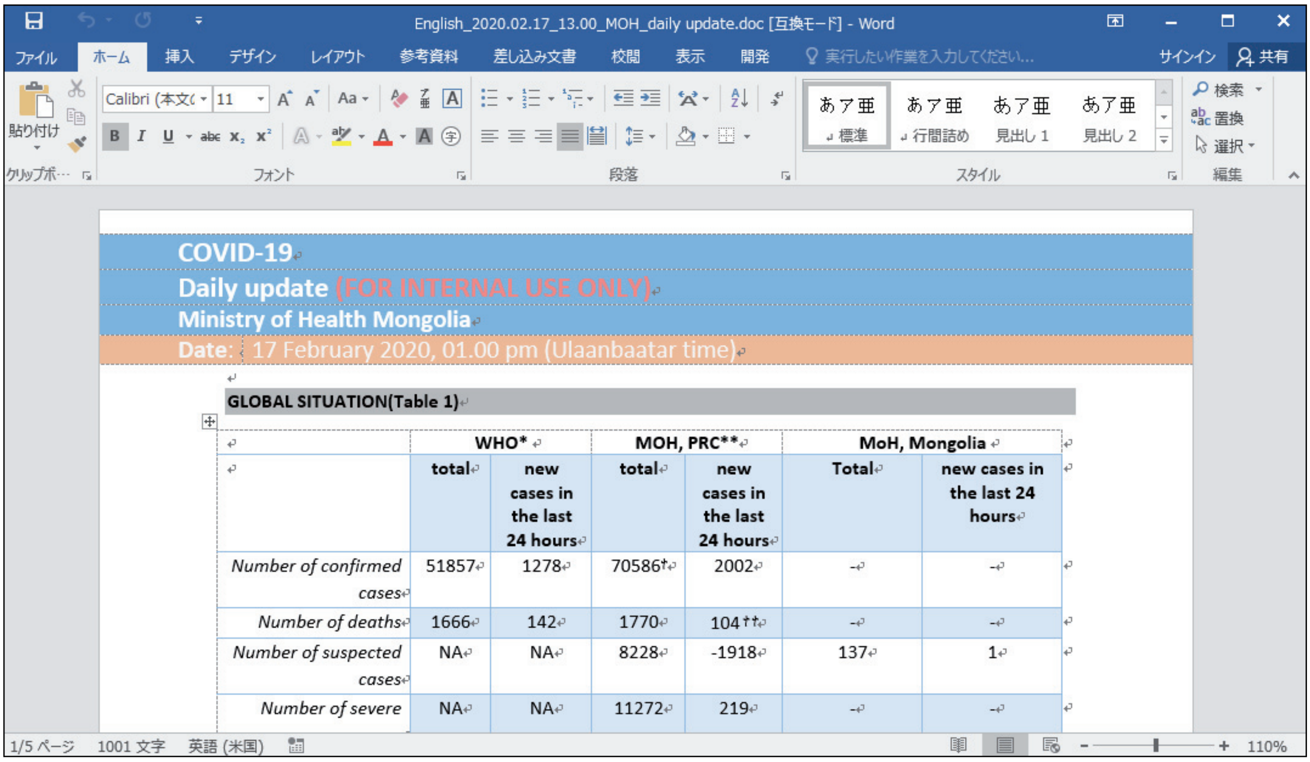


Figure 14: 'English_2020.02.17_13.00_MOH_daily update.doc'.

Also as in Case 1, ‘woldfunc.wll’ is Poison Ivy, and it downloads three cab files from the C&C server and runs Cotx RAT in exactly the same way.

In Case 1, ‘o.exe’ and ‘n.exe’ were only used to investigate the environment and steal information. However, in Case 2, there was more of a breach.

First, ‘s.cab’ and ‘w.cab’ were downloaded, unpacked and executed by Poison Ivy. ‘S.cab’ contains an executable file called ‘s.exe’. This is a checker to investigate whether it can be compromised by exploiting MS17-010 against the host passed as an argument. An attacker who finds a laterally deployable host with ‘s.exe’ then uses the ‘w.exe’ contained in ‘w.cab’ to do the actual compromise. ‘W.exe’ is a tool that actually exploits MS17-010. The attacker used it to inject a DLL file into the compromised host, in the lsass.exe process, to execute it.

The injected DLL file is the second Poison Ivy. However, it accesses a different C&C server from the one accessed by the Poison Ivy that was initially executed. Using the second Poison Ivy, the attacker continued to compromise. We observed lateral movement on two hosts and investigated further breaches on each host.

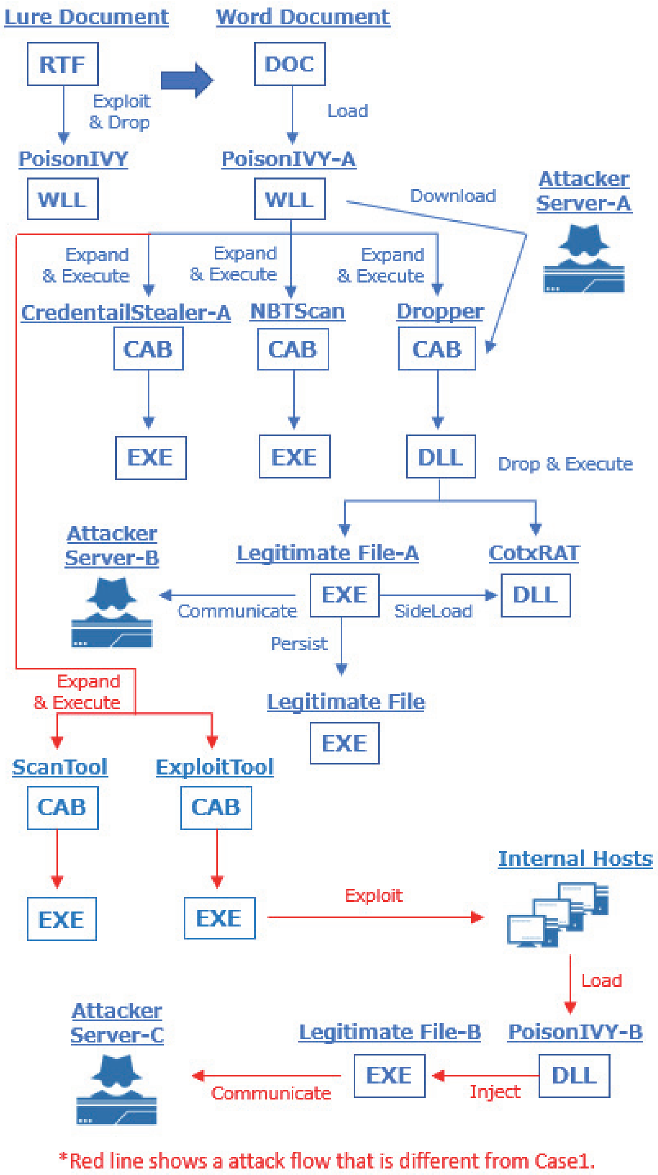


Figure 15: Whole picture of attack Case 2.

Table 3 shows the malware and files observed during attack Case 2.

On one of the two hosts (Internal Host-A), three cab files (‘nbt.cab’, ‘sh.cab’, ‘ss.cab’) were downloaded, unpacked and executed. Of these, ‘sh.cab’ contains a file called ‘show.exe’. This is a tool that steals username, domain and password from the ‘lsass.exe’ process. Also, ‘ss.cab’ contains a file called ‘dwm.exe’. This is the RAT we call Tmanger.

Item	File path	Description
Lure document	English_2020.02.17_13.00_MOH_daily update.doc	RTF file that attacker sends
Word document	-	Any Microsoft Office Word file
Poison Ivy-A	%APPDATA%\Microsoft\Word\STARTUP\ woldfunc.wll	Poison Ivy RAT
Credential stealer-A	%ALLUSERSPROFILE%\Comms\o.cab %ALLUSERSPROFILE%\Comms\o.exe	Dump tool for Outlook credentials
NBT scan	%ALLUSERSPROFILE%\Comms\nbt.cab %ALLUSERSPROFILE%\Comms\n.exe	NBT scan tool
Dropper	%ALLUSERSPROFILE%\Comms\in.cab %ALLUSERSPROFILE%\Comms\intel.dll	Dropper of Cotx RAT
Legitimate File-A	%ALLUSERSPROFILE%\Comms\intel.exe %APPDATA%\Intel\Intel(R) Processor Graphics\ IntelGraphicsController.exe	Legitimate executable file of Intel Corporation
Cotx RAT	%ALLUSERSPROFILE%\Comms\RasTls.dll	Cotx RAT is side-loaded by intel.exe
ScanTool	%ALLUSERSPROFILE%\Comms\s.cab %ALLUSERSPROFILE%\Comms\s.exe	Scan tool for MS17-010
ExploitTool	%ALLUSERSPROFILE%\Comms\w.cab %ALLUSERSPROFILE%\Comms\w.exe	Exploit tool for MS17-010
Poison Ivy-B	%ALLUSERSPROFILE%\Comms\x86.dll %ALLUSERSPROFILE%\Comms\x64.dll	Poison IVY is injected into lsass.exe
Legitimate File-B	%SYSTEMROOT%\System32\lsass.exe	Legitimate executable file of Microsoft Corporation

Table 3: Malwares and files observed during attack Case 2.

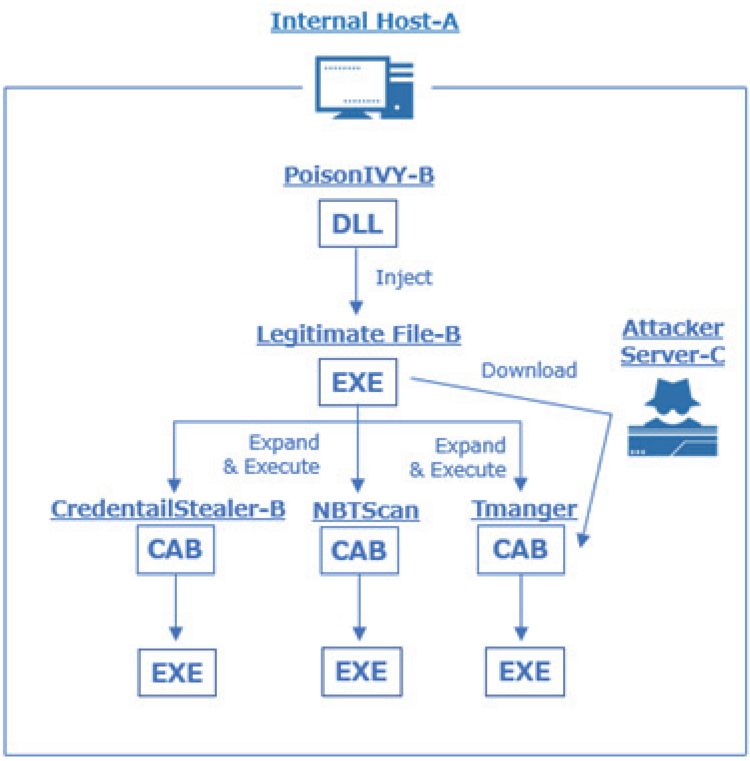


Figure 16: Attack flow in Internal Host-A.

Item	File path	Description
Credential stealer-B	%ALLUSERSPROFILE%\GroupPolicy\sh.cab %ALLUSERSPROFILE%\GroupPolicy\show.exe	Dump tool from lsass.exe
NBTScan	%ALLUSERSPROFILE%\GroupPolicy\nbt.cab %ALLUSERSPROFILE%\GroupPolicy\n.cab	NBT scan tool
Tmanger	%ALLUSERSPROFILE%\Microsoft\DRM\ss.cab %ALLUSERSPROFILE%\Microsoft\DRM\dwm.exe	Tmanger RAT

Table 4: Malware and files observed during the attack Internal Host-A.

On the other host (Internal Host-B), the files 'In.cab' and 'WindowsResKits.dll' were downloaded. 'In.cab' contains a file called 'Instsrv.exe'. This impersonates the legitimate tool provided as a resource kit and registers 'WindowsResKits.dll' as a service. 'WindowsResKits.dll' is a new type of malware that we call nccTrojan.

The attacker attempted further breaches using Active Directory administrator passwords stolen by 'show.exe'. However, the passwords stolen by the attackers were old. As a result, the breach failed, and the activity ended.

In Case 2, the flow until the attacker used Cotx RAT was the same as in Case 1. Therefore, we will explain the lateral movement in detail below, except for the case shown in Case 1.

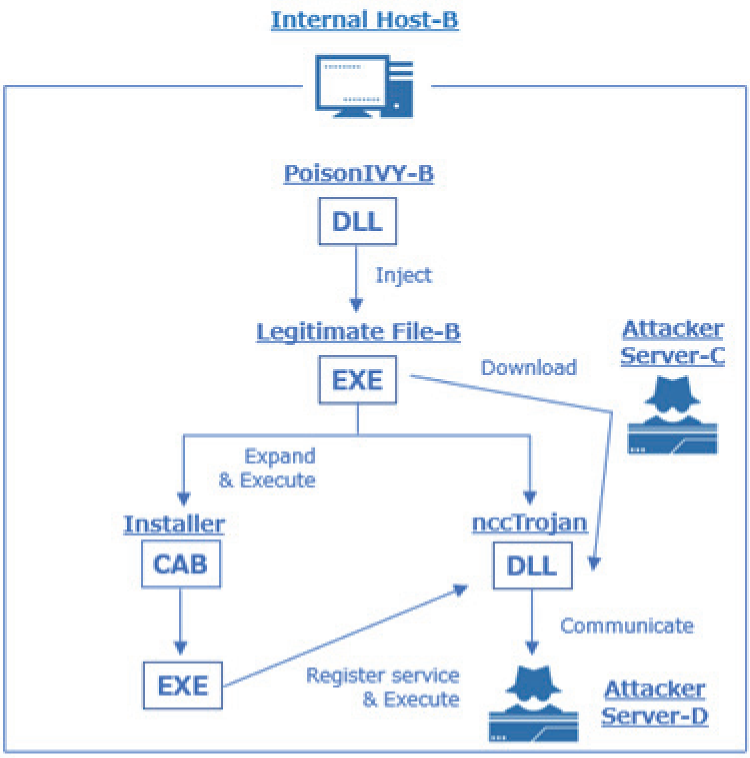


Figure 17: Attack flow in Internal Host-B.

Item	File path	Description
Installer	%ALLUSERSPROFILE%\Microsoft\Crypto\In.cab %ALLUSERSPROFILE%\Microsoft\Crypto\Instsr.exe	Register nccTrojan as a service and execute
nccTrojan	%ALLUSERSPROFILE%\Microsoft\Crypto\ WindowsResKits.dll	nccTrojan RAT

Table 5: Malware and files observed during the attack Internal Host-B.

Lateral movement

The attacker used two tools for lateral movement. The first is s.exe. This is a tool that checks if it is possible to exploit the MS17-010 vulnerability on the specified host. It is a PE file converted from the public Python script [9] by PyInstaller.

```
$ s.exe

                    _____ _______         _______ _____ _______
 .--------.-----| _   |  _   |______|  _   | _   |  _   |
 |        |__ --|.|   |___|   |______|.  |   |.|   |.  |   |
 |__|__|__|_____`-|.  |  /   /       |.  |   `-|.  |.  |   |
                  |:  | |   |        |:  1   | |:  |:  1   |
                  |::.| |   |        |::.. . | |::.|::.. . |
                  `---' `---'        `-------' `---`-------'
 .--------.---.-.-----.-----.   .-----.----.---.-.-----.-----.-----.----.
 |        |  _  |__ --|__ --|   |__ --|  __|  _  |     |     |  -__|   _|
 |__|__|__|___._|_____|_____|   |_____|____|___._|__|__|__|__|_____|__|

                                 MS17-010-m4ss-sc4nn3r v1.0
                    Written by:
                  Claudio Viviani

                 http://www.homelab.it

                  info@homelab.it
               homelabit@protonmail.ch

              https://twitter.com/homelabit


[+] Usage: s.exe ip or ip/CIDR or ip/subnet

    Example: s.exe 192.168.0.1
             s.exe 192.168.0.0/24
             s.exe 192.168.0.0/255.255.255.0
```

Figure 18: Execution of s.exe.

After the attacker found a vulnerable host using s.exe, the second tool, ‘w.exe’, was used. This is a tool which enables the MS17-010 vulnerability to be exploited. It is a PE file which seems to have been created based on the public Python script [10].

The w.exe tool sends x86.dll or x64.dll in the same directory to the target host depending on the environment and injects it into the lsass.exe process.

```
$ w.exe
[1]------check backdoor and system version------
get_smb_signature 59437391
**** OS is Win 7 x86
**** backdoor is already installed!
[2]------Inject dll    ------
get_smb_signature 59437391
key 23f5a57b
dll_hex 200704
len_part: 204144
0 ---> 0x52
1 ---> 0x52
2 ---> 0x52

-- Snip --

49 ---> 0x52
50 ---> 0x52
**** dll is now injected!
```

Figure 19: Execution of w.exe.

Poison Ivy (second)

The behaviour of x86.dll and x64.dll is the same. Both are internally named ‘blu.dll’. The path of PDB was left in x64.dll as follows:

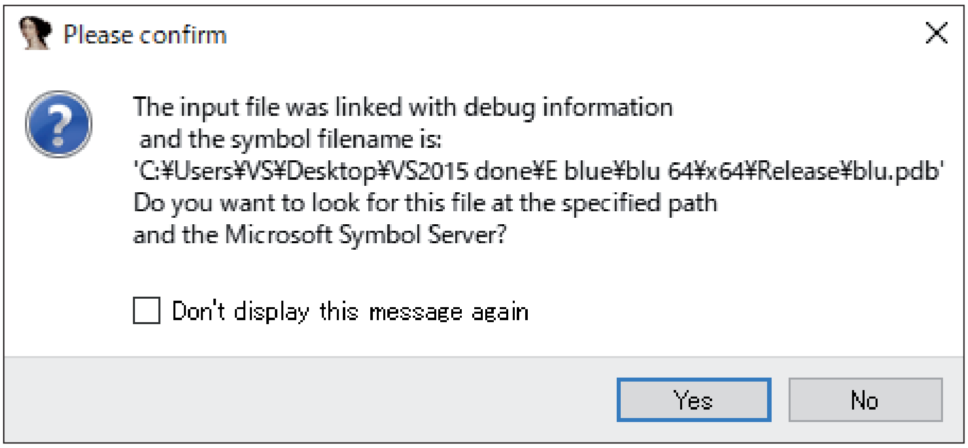


Figure 20: PDB path of x64.dll.

When the DLL file is injected into lsass.exe, the ‘Register’ function is executed. When executed, it creates three files (PotPlayerMini.exe, PotPlayer.dll and PAME13.tmp) under C:\Windows\Temp and executes PotPlayerMini.exe.

```
mov     edi, str.PotPlayerMini.exe ; 0x1002ccd0 ; "PotPlayerMini.exe"
or      ecx, 0xffffffff            ; -1
xor     eax, eax
push    0x1002cccc                 ; "wb"
repne   scasb al, byte es:[edi]
not     ecx
sub     edi, ecx
push    0x1002c0f8                 ; "C:\Windows\Temp\"
mov     esi, edi
mov     edx, ecx
mov     edi, 0x1002c0f8            ; "C:\Windows\Temp\"
or      ecx, 0xffffffff            ; -1
repne   scasb al, byte es:[edi]
mov     ecx, edx
dec     edi
shr     ecx, 2
rep     movsd dword es:[edi], dword ptr [esi]
mov     ecx, edx
and     ecx, 3
rep     movsb byte es:[edi], byte ptr [esi]
call    fcn.10001357
mov     esi, eax
push    esi
push    0x13e28
push    1                          ; 1
push    0x10008030                 ; "MZ\x90"
call    fcn.100011ed
```

Figure 21: Creating PotPlayerMini.exe.

PotPlayerMini.exe is a legitimate binary created by *Daum* and has a digital signature. PotPlayer.dll is loaded in the same directory, causing DLL side-loading.

PotPlayer.dll is the body of Poison Ivy (SPIVY). First, PAME13.tmp is decoded with RC4 to get configuration data. After that, it communicates with the C&C server just like the first Poison Ivy.

```
push    0x200                         ; 512
push    0x1000b284
push    0x10008064
call    fcn.10001520                  ; decode_rc4_key
mov     ecx, 0x10a7
xor     eax, eax
lea     edi, [var_864fh]
mov     byte [var_8650h], 0
rep     stosd dword es:[edi], eax
stosw   word es:[edi], ax
stosb   byte es:[edi], al
mov     edi, 0x1000b284
or      ecx, 0xffffffff               ; -1
xor     eax, eax
lea     edx, [var_8650h]
repne   scasb al, byte es:[edi]
not     ecx
dec     ecx
push    ecx                           ; int32_t arg_ch
push    0x1000b284                    ; int32_t arg_8h
push    edx                           ; int32_t arg_4h
call    fcn.10001550                  ; rc4_ksa
lea     eax, [var_43b0h]
push    0x42a0                        ; int32_t arg_ch
lea     ecx, [var_8650h]
push    eax                           ; int32_t arg_8h
push    ecx                           ; int32_t arg_4h
call    fcn.10001600                  ; rc4_prga
add     esp, 0x24
```

Figure 22: Decoding PAME13.tmp.

The configuration data of this Poison Ivy is shown in Table 6.

C&C server	45.76.211.18:443
	45.76.211.18:8080
Campaign ID	TOEI
Group ID	TOEI
Mutex	G9u3cUoJs
Password	kos@On

Table 6: Poison Ivy's configuration data.

This Poison Ivy was executed on two hosts. On the first host, the attacker downloaded and executed a credential stealer and a RAT called 'Tmanger'. On the other host, the attacker downloaded and executed 'nccTrojan'.

Credential stealer (second)

The 'sh.cab' file contains a file named 'show.exe', which is a tool enabling the stealing of *Windows* credentials. Show.exe steals username, domain and password from the lsass.exe process. The attacker executed show.exe and retrieved the credentials, however the penetration to another host didn't succeed because the credentials in our environment were old.

```
$ show.exe
U: Administrator
DO: [Reducted]
ps: [Reducted]

U: ANONYMOUS LOGON
DO: NT AUTHORITY
Specific LUID NOT found

U: LOCAL SERVICE
DO: NT AUTHORITY
ps:
```

Figure 23: Execution of show.exe.

Tmanger

The PDB path was left in dwm.exe, which was included in ‘ss.cab’. We call it ‘Tmanger’ because of the string contained in this pathname.

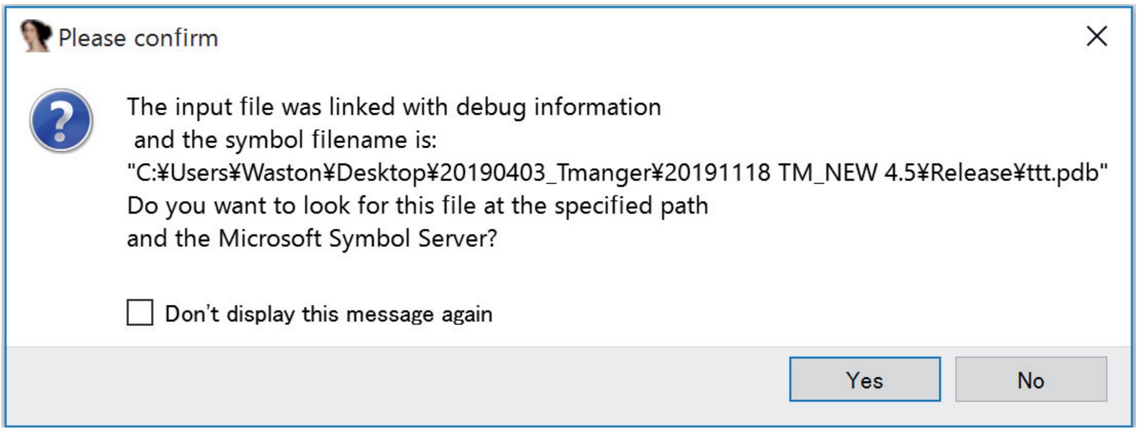


Figure 24: PDB path of dwm.exe.

When ‘dwm.exe’ is executed, it creates ‘test.dll’ under the Temp folder of the user account. The data in it is in the resource section of the ‘dwm.exe’ file.

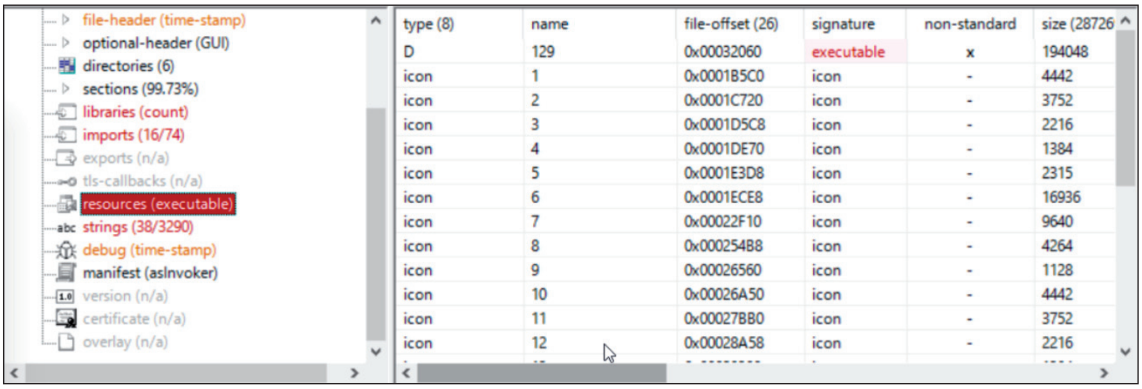


Figure 25: Test.dll embedded in the resource section of dwm.exe.

It also copies itself under the Temp folder with the filename ‘master.exe’. And then it persists to execute master.exe with the key ‘HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\Master’.

It then uses rundll32.exe to execute a function called ‘Entery’ in ‘test.dll’. This allocates an area in memory, writes the code and executes it. We believe this is the body of the RAT. After collecting information from the PC, it attempts to communicate with the C&C server on ports 80, 443 and 5222, in that order.

An example of communication with the C&C server is shown below. As a result of analysis, it was found that the first four bytes are the data size and the rest are encoded data.

```
                Data Size                Encoded Data
00000000  15 00 00 00 1b f5 42 5a  9e 55 92 03 7a 0e b8 b6   ......BZ .U..z...
00000010  f8 8c 36 19 12 9e 54 62  56                        ..6...Tb V
```

Figure 26: Traffic data.

The data part is encrypted with RC4 and the key is hex encoded as follows.

```
アドレス  Hex                                              ASCII
0093C970 00 0C 7C 17 A7 1C D2 07 DA 9E EE C5 8B 0B D7 86 ..|.§.Ò.Ú.îÅ..×.
0093C980 AB 7E 5E 1C 55 C5 6E 2E 75 10 A0 FC C2 C8 7A 99 «~^.UÅn.u. üÂÈz.
0093C990 DB 6C 5C B5 2A C6 32 EE 03 C5 4C A4 4D 0A 20 24 Ûl\µ*Æ2î.ÅL¤M. $
0093C9A0 92 CD D9 CB 8C 89 81 80 A5 90 D1 AF 02 B6 5F 15 .ÍÙË....¥.Ñ¯.¶_.
```

Figure 27: RC4 encryption key.

The result of decoding the data part is shown in Figure 28.

```
          Encoded PID  Command
00000000  33 35 34 38  01 80 be 39  00 73 79 73 74 65 6d 69  |3548...9.systemi|
00000010  6e 66 6f 0d 0a                                     |nfo..|
00000015
```

Figure 28: Decoding results of traffic data.

The first four bytes are converted from the PID value as follows:

((CurrentProcessID % 9) × 1000) + ((CurrentProcessID % 1000) + 1000)

Also, we found that the fifth byte is a command to the RAT.

As a result of our analysis, we consider that the functions of this RAT are as follows:

- Command execution by PowerShell
- Sending file information on a PC
- Sending the contents of a file on a PC
- Deleting files on a PC
- Command execution by the CreateProcess function
- Sending screen capture images
- Keylogger.

No further infringement occurred on this host.

nccTrojan

Using the second Poison Ivy, the attacker placed the installer 'Instsrv.exe' and the RAT 'WindowsResKits.dll' on 'C:\ProgramData\Microsoft\Crypto\'. When Instsrv.exe is executed, it registers a fake service as Windows Resource Kits, as shown in Table 7, copies WindowsResKits.dll to 'C:\Windows\SysWOW64\' or 'C:\Windows\System32\', and starts the service. When the service starts, the svchost.exe process loads WindowsResKits.dll.

Name	Image path
Microsoft Windows Resource Kits	C:\Windows\System32\svchost.exe -k WindowsResKits

Table 7: Registered fake service.

When we analysed WindowsResKits.dll, we found the PDB file path and the compilation date and time as shown in Table 8.

Item	Value
PDB file path	C:\Users\abc\Desktop\cTrojan\2.1\HK\Release\Client.pdb
Compilation timestamp	2019-12-27 01:07:25

Table 8: The meta information of WindowsResKits.dll.

WindowsResKits.dll decrypts config information and communication contents using the method shown in Tables 9 and 10. We confirmed that the character string 'ncc' exists in the decrypted data, and that the character string is necessary to start the process for commands received from the C&C server. Figure 29 shows a function being called to check if the received data is 'ncc'. The first argument is the decrypted character string, the second argument is the received data, and the third argument is the size of the string to compare. For this reason, we call this RAT 'nccTrojan'.

Method	**AES (CFB mode)**
Key (hex-encoded)	12AB56FF56CDCCED99EE3CBA02270567908CAF772F6BAC7C6C2BF1DDEEC9D6BB (256 bits)
IV (hex-encoded)	02242123421315713AB6A8A0C8DC5AF3 (128 bits)

Table 9: Encryption for config information.

Method	**AES (CFB mode)**
Key (hex-encoded)	981511371412780969AFC3AB2072018709A83A3332466A8B56FF3FAB8E6C3DAA (256 bits)
IV (hex-encoded)	2042123224315117031B1A0A3CCDA53F (128 bits)

Table 10: Encryption for communication contents.

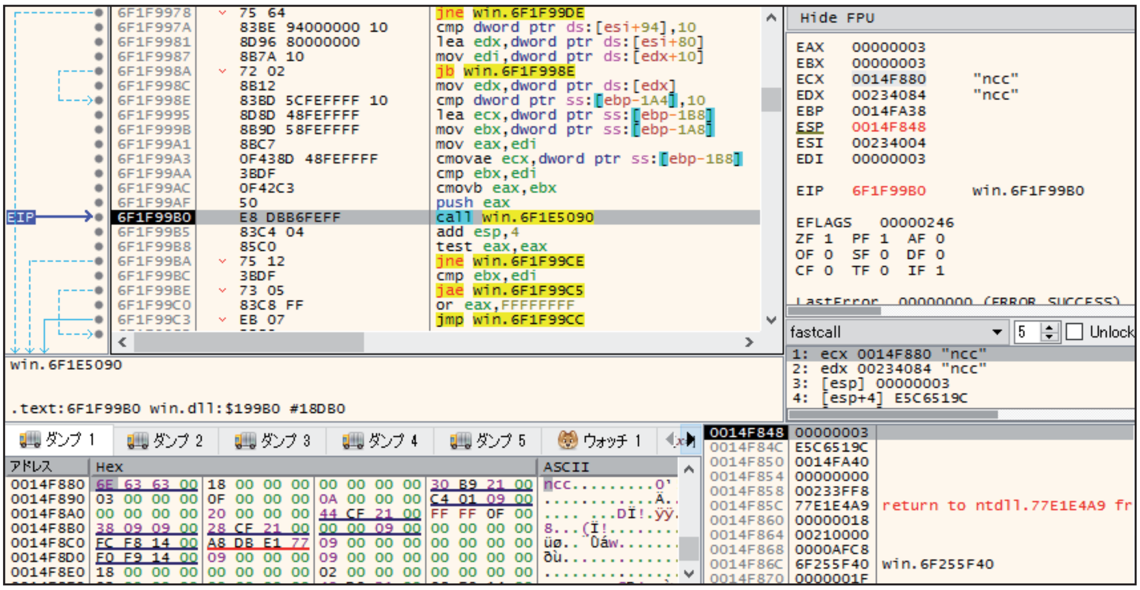


Figure 29: Comparison of character string 'ncc' and received data.

The nccTrojan connected to 45[.]77.129.213 on port 443/TCP and communicated with the C&C server. As shown in Figure 30, the TCP payload consisted of an eight-byte SIZE field and a following DATA field. It is a feature that the SIZE field was described as a decimal character string and the invalid digit was 'x'.

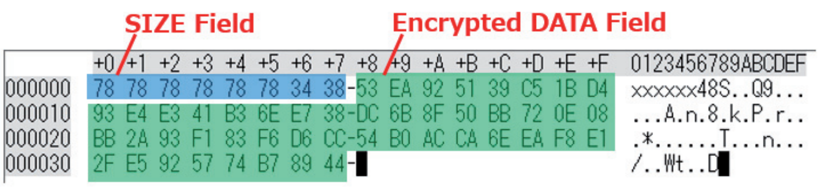


Figure 30: An example of received TCP payload.

We confirmed that the same following functions are implemented in the nccTrojan as the RAT.

- Remote Shell
- Send Disk Information
- Send File List
- Send Process List

- Download File (Read File)
- Upload File
- Operate File (Copy, Move, Delete)
- Kill Process.

CORRELATION

TA428 has been reported to actively use the Royal Road RTF Weaponizer in Operation LagTime IT [2, 3]. The RTF file generated by Royal Road RTF Weaponizer has several characteristics. It can be classified according to the RTF object, encoding algorithm, etc. TA428, Tick and Tonto, are said to belong to Group-B [3]. Attack groups belonging to Group-B mainly target East Asia, especially Russia, Mongolia, South Korea and Japan – countries which have much overlap with the target countries of TA428.

The Poison Ivy used by TA428 has a different traffic structure from the normal Poison Ivy. This is a variant called SPIVY. One example of the use of SPIVY was in Hong Kong in March 2016 [5]. In this attack, similar to the TA428 attack this time, the malware was executed by DLL side-loading using a legitimate *Symantec* binary and RasTls.dll.

This time we have found that TA428 uses PotPlayerMini for DLL side-loading. This technique is extremely rare. Until now, only a few cases of DLL side-loading using PotPlayerMini have been reported [11, 12] – these are said to be the attacks associated with DragonOK (and Danti). A case in Hong Kong, reported by *Palo Alto Networks* [11], uses PotPlayerMini to execute Poison Ivy, similar to this TA428 attack. In addition, the TA428 attack that is believed to have targeted Kazakhstan around April 2019 is said to have used malware related to Danti [13]. DragonOK targets East Asian countries such as Japan and Taiwan and is consistent with the target area of TA428.

CONCLUSION

Operation LagTime IT by TA428 has been observed since at least March 2019 and has not changed TTPs for more than a year. It mainly targets government agencies in East Asia and uses RTF files generated by the Royal Road RTF Weaponizer, Poison Ivy and Cotx RAT. It also uses tools that exploit MS17-010 for lateral movement, NBTScan for environmental investigations, and tools to steal credentials. It also uses previously unknown advanced RATs such as Tmanger and nccTrojan.

TA428 is included in Group-B alongside Tick and Tonto. It may also be associated with previous SPIVY-based attacks against Hong Kong by DragonOK and Danti.

It is expected that attacks by TA428 will continue to be aggressive. To protect your system from attacks by TA428, we recommend that you use the information presented in this paper for detection and defence.

REFERENCES

[1] Malpedia. Pirate Panda. https://malpedia.caad.fkie.fraunhofer.de/actor/pirate_panda.

[2] Proofpoint. Chinese APT “Operation LagTime IT” Targets Government Information Technology Agencies in Eastern Asia. https://www.proofpoint.com/us/threat-insight/post/chinese-apt-operation-lagtime-it-targets-government-information-technology.

[3] nao_sec. An Overhead View of the Royal Road. https://nao-sec.org/2020/01/an-overhead-view-of-the-royal-road.html.

[4] MITRE ATT&CK. Office Application Startup. https://attack.mitre.org/techniques/T1137/.

[5] Palo Alto Networks. New Poison Ivy RAT Variant Targets Hong Kong Pro-Democracy Activists. https://unit42.paloaltonetworks.com/unit42-new-poison-ivy-rat-variant-targets-hong-kong-pro-democracy-activists/.

[6] Conix Cybersécurité. Poison Ivy RAT. https://www.conix.fr/wp-content/uploads/2013/10/Poison-Ivy-RAT-conf-comms.pdf.

[7] Security Xploded. Outlook Password Dump. https://securityxploded.com/outlook-password-dump.php.

[8] Steve Friedl’s Unixwiz.net Tools. nbtscan. http://www.unixwiz.net/tools/nbtscan.html.

[9] GitHub claudioviviani. ms17-010-m4ss-sc4nn3r. https://github.com/claudioviviani/ms17-010-m4ss-sc4nn3r/blob/master/ms17-010-m4ss-sc4nn3r.py.

[10] GitHub pythonone. MS17-010. https://github.com/pythonone/MS17-010/blob/master/exploits/eternalblue/eternalblue.py.

[11] Palo Alto Networks. Unit 42、日本を対象に開発されたDragonOKバックドアマルウェアの新種を発見. https://unit42.paloaltonetworks.jp/unit-42-identifies-new-dragonok-backdoor-malware-deployed-against-japanese-targets/.

[12] Kaspersky. CVE-2015-2545: overview of current threats. https://securelist.com/cve-2015-2545-overview-of-current-threats/74828/.

[13] nao_sec. Royal Road IoC. https://nao-sec.org/jsac2020_ioc.html.