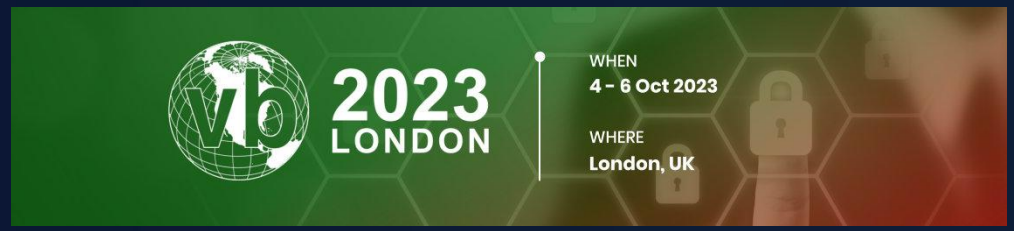


Unveiling Activities of Tropic Trooper 2023: Deep Analysis for Xiangoop Loader and EntryShell payload

ITOCHU Cyber & Intelligence Inc.

Co.Tomorrowing

MACNICA

Yusuke Niwa (Itochu Cyber & Intelligence Inc.)
Suguru Ishimaru (Itochu Cyber & Intelligence Inc.)
Hajime Yanagishita (Macnica, Inc.)

Who we are?

Suguru Ishimaru
Itochu Cyber & Intelligence Inc.

Yusuke Niwa
Itochu Cyber & Intelligence Inc.

Hajime Yanagishita
Macnica, Inc.

Agenda

1. Overview
2. Initial Infection Vectors
3. Xiangoop Loader
    > payload: Cobalt Strike Beacon
    > payload: EntryShell
4. CrowDoor
5. Attribution
6. Conclusions

Overview


vb 2023 LONDON
Virus Bulletin International Conference
4 -6 October 2023
RED ROOM
VB2023 LONDON W6
VIRUSBULLETIN.COM

Motivation

- The threat actor group Tropic Trooper, which uses a RAT called Keyboy, has been very active this year.

- In particular, the latest attack campaign has been observed to target overseas companies based in China. We believe it is necessary to continue to monitor attack trends closely.

Who is Tropic Trooper

- Tropic Trooper, a cyberespionage group that has been targeting organizations in the Asia-Pacific region.

- Tropic Trooper (a.k.a. Pirate Panda, Keyboy, and APT23) is a group that has been active since 2011, according to Trend Micro.

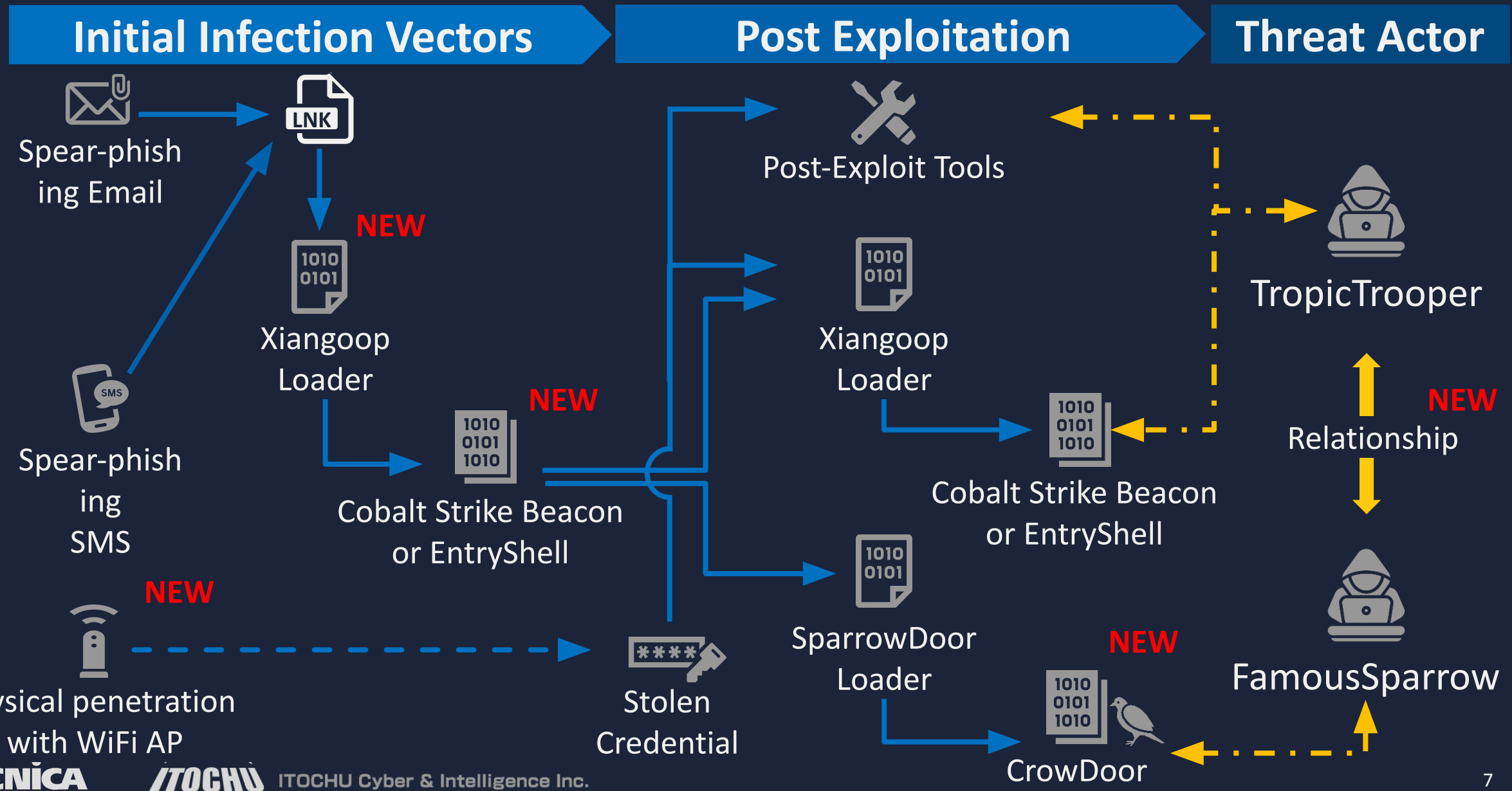
https://documents.trendmicro.com/assets/wp/wp-operation-tropic-trooper.pdf
https://citizenlab.ca/2016/11/parliament-keyboy/
https://www.macnica.co.jp/business/security/security-reports/pdf/cyberespionage_report_2022.pdf

What’s NEW!!

Throughout this attack campaign, there were three new findings

- Discovery of a new loader malware we named Xiangoop Loader and its deep analysis
- Features of EntryShell, a major updated version of the KeyBoy malware
- Discovery of unreported RAT, we named CrowDoor, a likely to have been used by FamousSparrow, and the relationship between threat actors

Campaign Overview



Initial Infection Vectors

Initial Infection Vectors

- We observe and estimate three main types of initial entry paths in this campaign.
  - Type 1: Spear Phishing E-mails
  - Type 2: Spear Phishing SMS
  - Type 3: Physical penetration

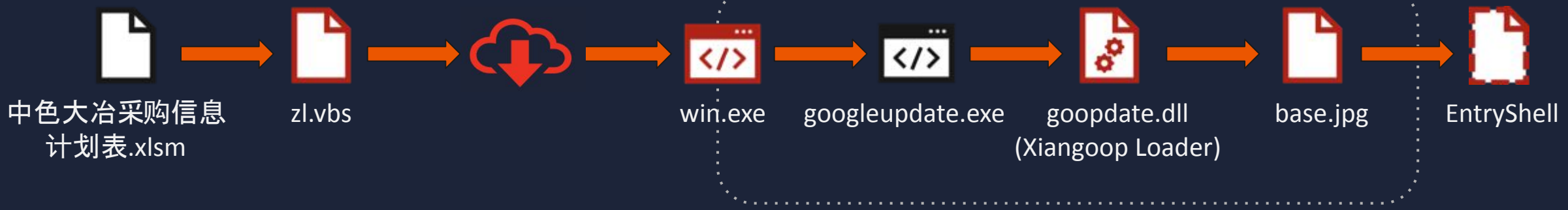
Type 1: Persistent Spear Phishing E-mail

More than 10 spear-phishing E-mails were observed in 2023.

Type 2: Spear Phishing Using WeChat Application

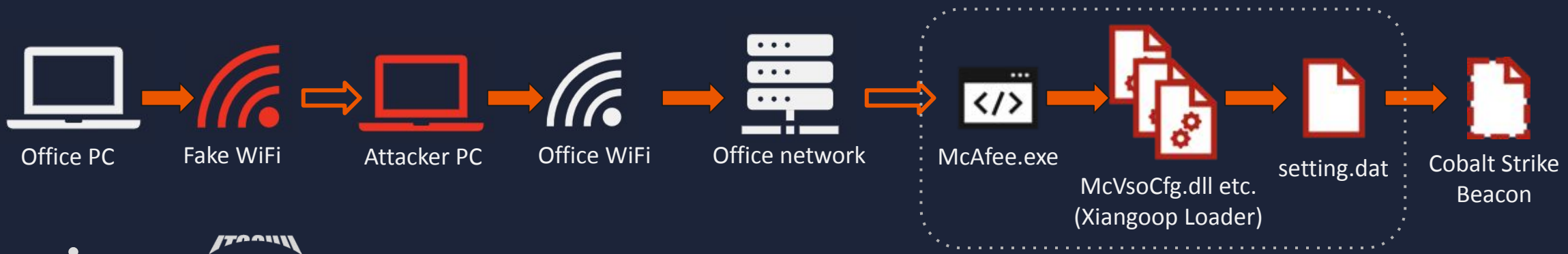
We observed a spear-phishing SMS by the Windows version of the WeChat application.

http[:]//mail[.]mraden[.]com/win.rar



Type 3: Physical Penetration with a fake WiFi AP

- An unmanaged host was connected to the IP address of a wireless controller at one of the sites. Possibly physically connected by an attacker.
- Prior to the intrusion, a suspicious WiFi AP was reported from the same site.
- The threat actor might have physically entered the office building temporarily to compromise the internal network, rather than spear-phishing!!



Xiangoop Loader

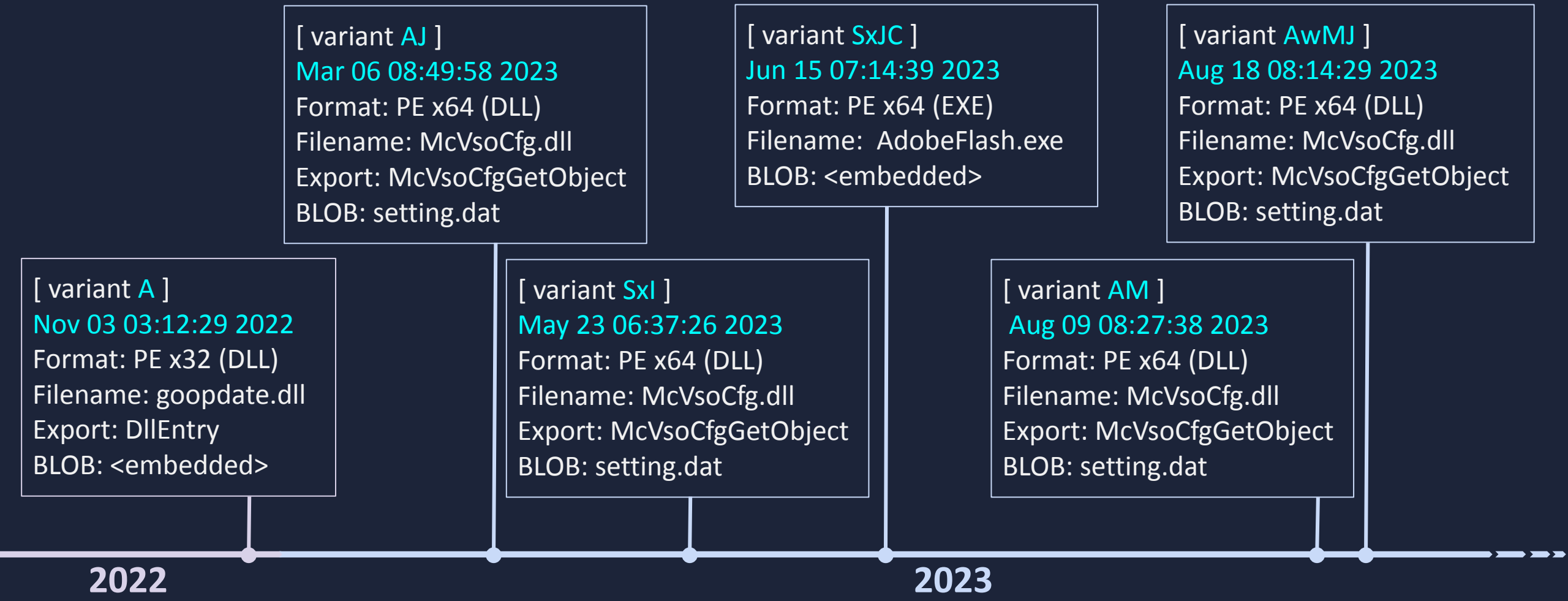
Xiangoop Loader

Why we named Xiangoop Loader for this malware?
Because, we discovered an artifact including a PDB file by the operator’s mistake.

Two specific words from the folder name of the developer’s env,
and combination to generate the loader name.

xiang(mu) + goop(date.dll) -> Xiangoop

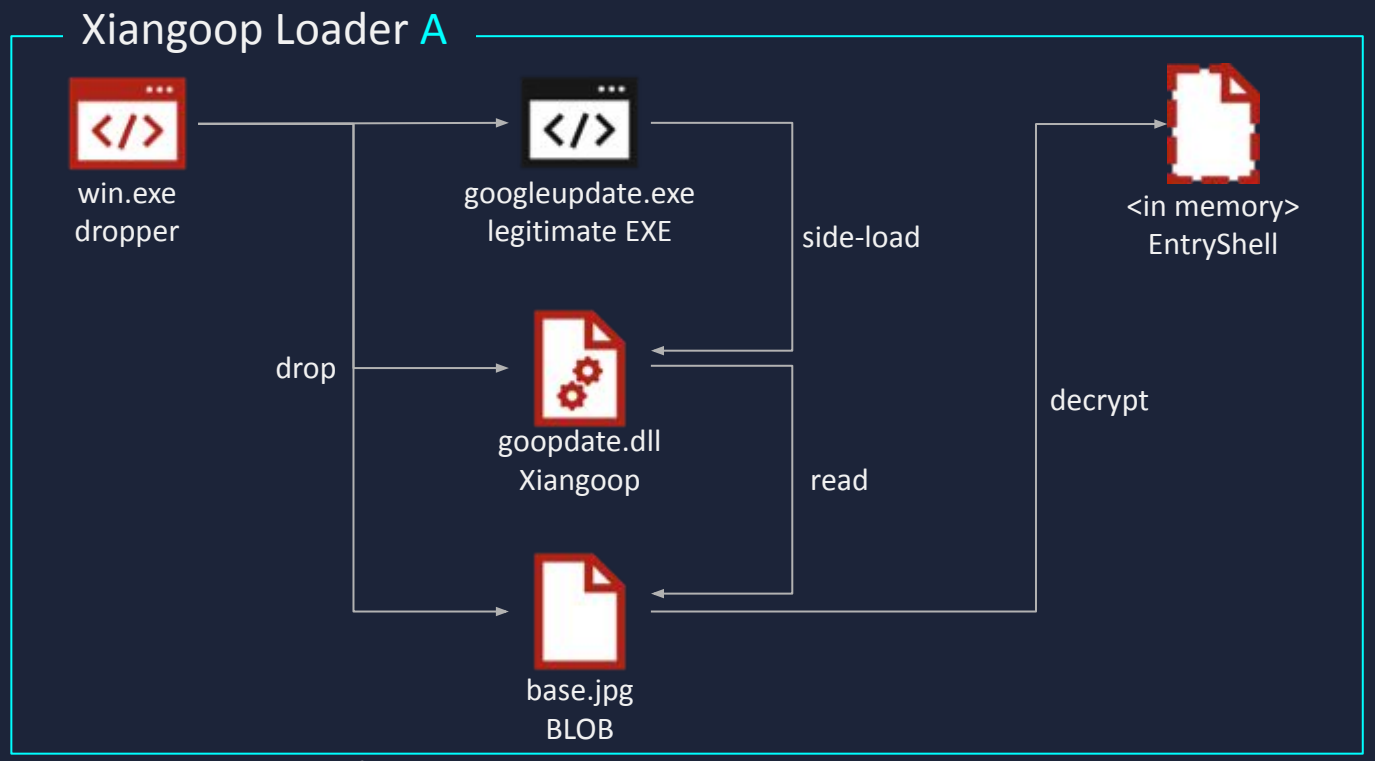
Xiangoop Loader: Evolution of the Xiangoop



200+ samples -> 6 variants

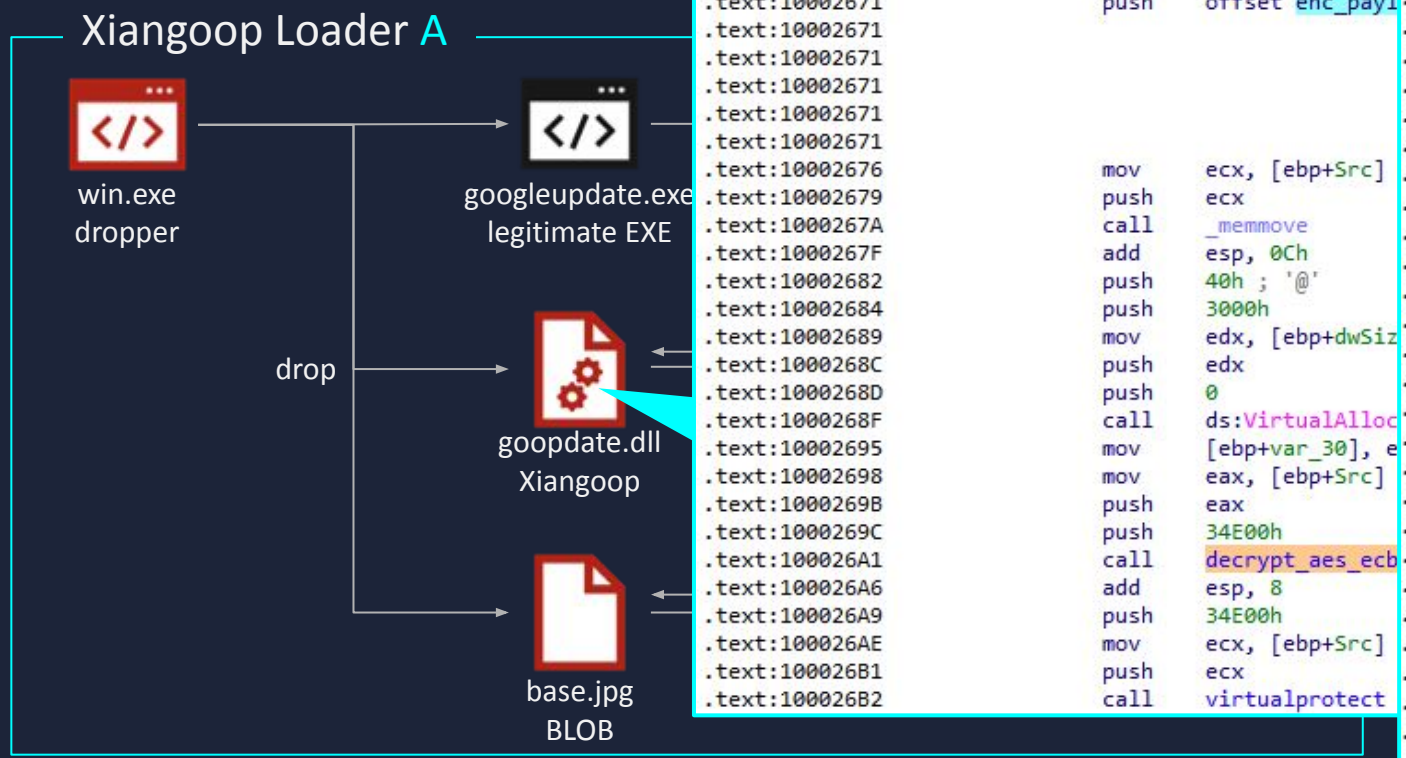
Xiangoop Loader: Variant A in Nov 2022

Xiangoop Loader A is
- a simple loader
- AES ECB mode
- hardcoded key = "123456AAAAAAAAAA”



Xiangoop Loader: Variant A in Nov 2022

Xiangoop Loader A is
a simple loader
AES ECB mode
hardcoded key = "123456AAAAAAAAAA"



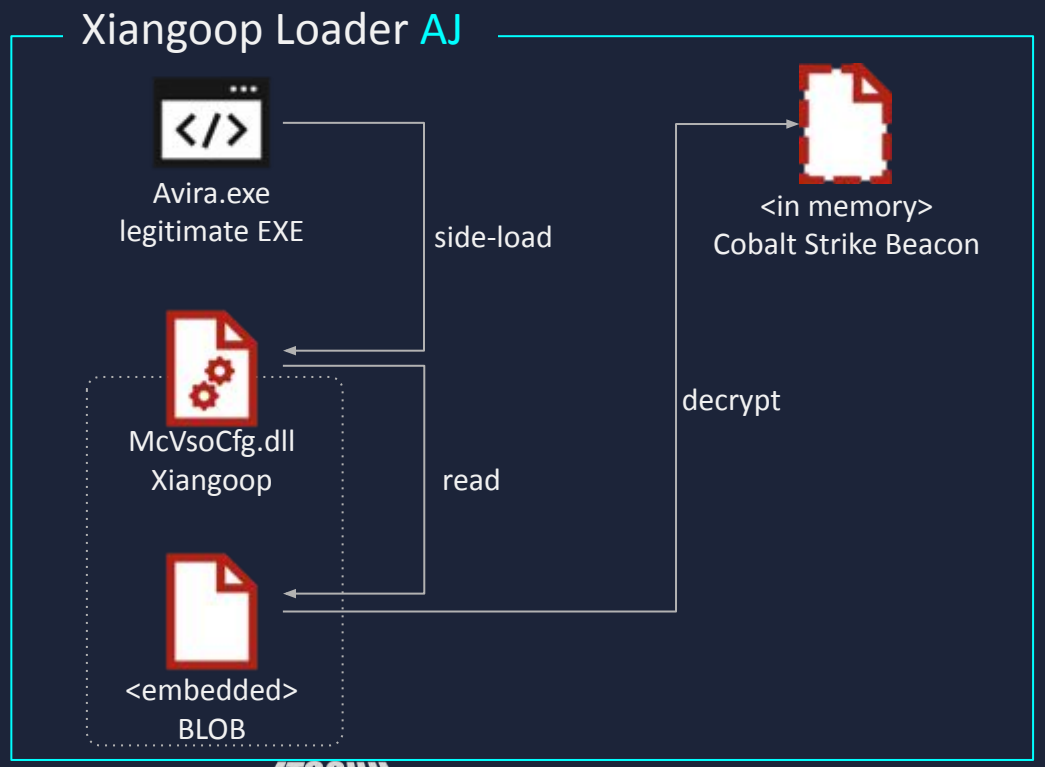

```
.text:1000266C                 push    34E00h
.text:10002671                 push    offset enc_payl
.text:10002671
.text:10002671
.text:10002671
.text:10002671
.text:10002671
.text:10002676                 mov     ecx, [ebp+Src]
.text:10002679                 push    ecx
.text:1000267A                 call    _memmove
.text:1000267F                 add     esp, 0Ch
.text:10002682                 push    40h ; '@'
.text:10002684                 push    3000h
.text:10002689                 mov     edx, [ebp+dwSiz
.text:1000268C                 push    edx
.text:1000268D                 push    0
.text:1000268F                 call    ds:VirtualAlloc
.text:10002695                 mov     [ebp+var_30], e
.text:10002698                 mov     eax, [ebp+Src]
.text:1000269B                 push    eax
.text:1000269C                 push    34E00h
.text:100026A1                 call    decrypt_aes_ecb
.text:100026A6                 add     esp, 8
.text:100026A9                 push    34E00h
.text:100026AE                 mov     ecx, [ebp+Src]
.text:100026B1                 push    ecx
.text:100026B2                 call    virtualprotect
```

```
.text:10002523                 sub     esp, 1FCh
.text:10002529                 mov     [ebp+aes_key], 31h ; '1'
.text:1000252D                 mov     [ebp+var_13], 32h ; '2'
.text:10002531                 mov     [ebp+var_12], 33h ; '3'
.text:10002535                 mov     [ebp+var_11], 34h ; '4'
.text:10002539                 mov     [ebp+var_10], 35h ; '5'
.text:1000253D                 mov     [ebp+var_F], 36h ; '6'
.text:10002541                 mov     [ebp+var_E], 41h ; 'A'
.text:10002545                 mov     [ebp+var_D], 41h ; 'A'
.text:10002549                 mov     [ebp+var_C], 41h ; 'A'
.text:1000254D                 mov     [ebp+var_B], 41h ; 'A'
.text:10002551                 mov     [ebp+var_A], 41h ; 'A'
.text:10002555                 mov     [ebp+var_9], 41h ; 'A'
.text:10002559                 mov     [ebp+var_8], 41h ; 'A'
.text:1000255D                 mov     [ebp+var_7], 41h ; 'A'
.text:10002561                 mov     [ebp+var_6], 41h ; 'A'
.text:10002565                 mov     [ebp+var_5], 41h ; 'A'
.text:10002569                 push    10h
.text:1000256B                 lea     eax, [ebp+aes_key]
.text:1000256E                 push    eax
.text:1000256F                 lea     ecx, [ebp+var_1FC]
.text:10002575                 push    ecx
.text:10002576                 call    aes_init
.text:1000257B                 add     esp, 0Ch
.text:1000257E                 mov     [ebp+var_18], 0
.text:10002585                 mov     [ebp+var_4], 0
.text:1000258C
.text:1000258C loc_1000258C:                           ; CODE XREF:
.text:1000258C                 mov     edx, [ebp+arg_0]
.text:1000258F                 cmp     edx, [ebp+var_18]
.text:10002592                 jbe     short loc_100025CB
.text:10002594                 mov     eax, [ebp+var_4]
.text:10002597                 shl     eax, 4
.text:1000259A                 add     eax, [ebp+arg_4]
.text:1000259D                 push    eax
.text:1000259E                 mov     ecx, [ebp+var_4]
.text:100025A1                 shl     ecx, 4
.text:100025A4                 add     ecx, [ebp+arg_4]
.text:100025A7                 push    ecx
.text:100025A8                 lea     edx, [ebp+var_1FC]
.text:100025AE                 push    edx
.text:100025AF                 call    aes_dec
```

Xiangoop Loader: Variant AJ in Mar 2023

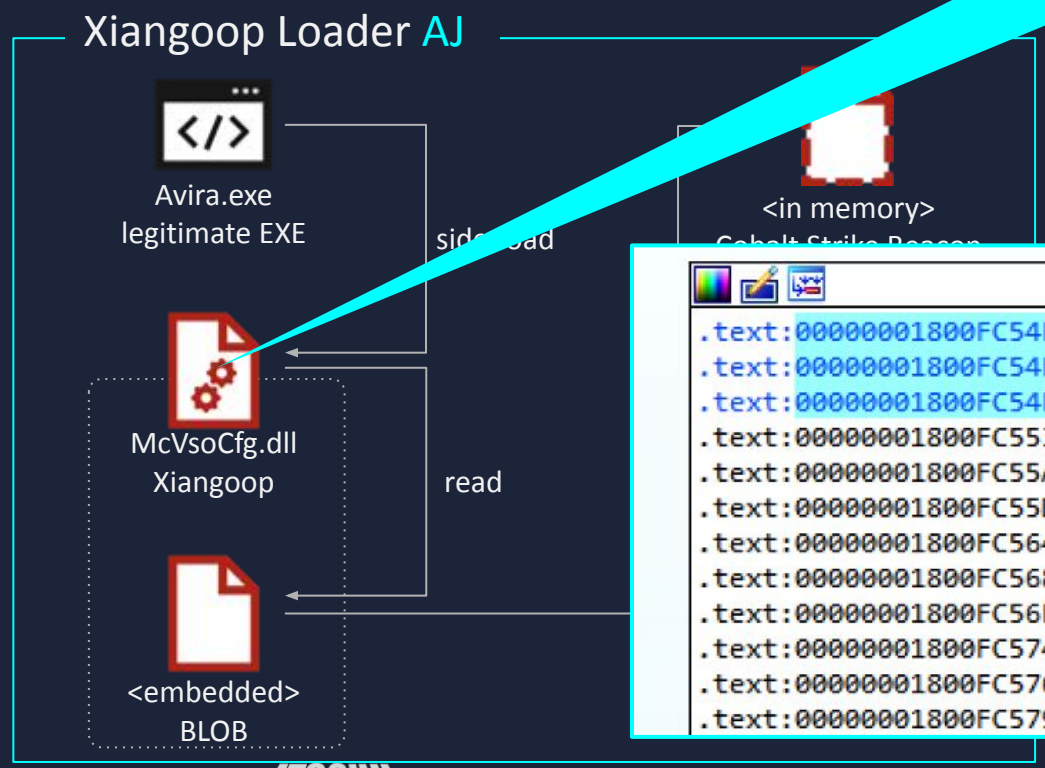
Xiangoop Loader AJ is almost the same as the variant A:

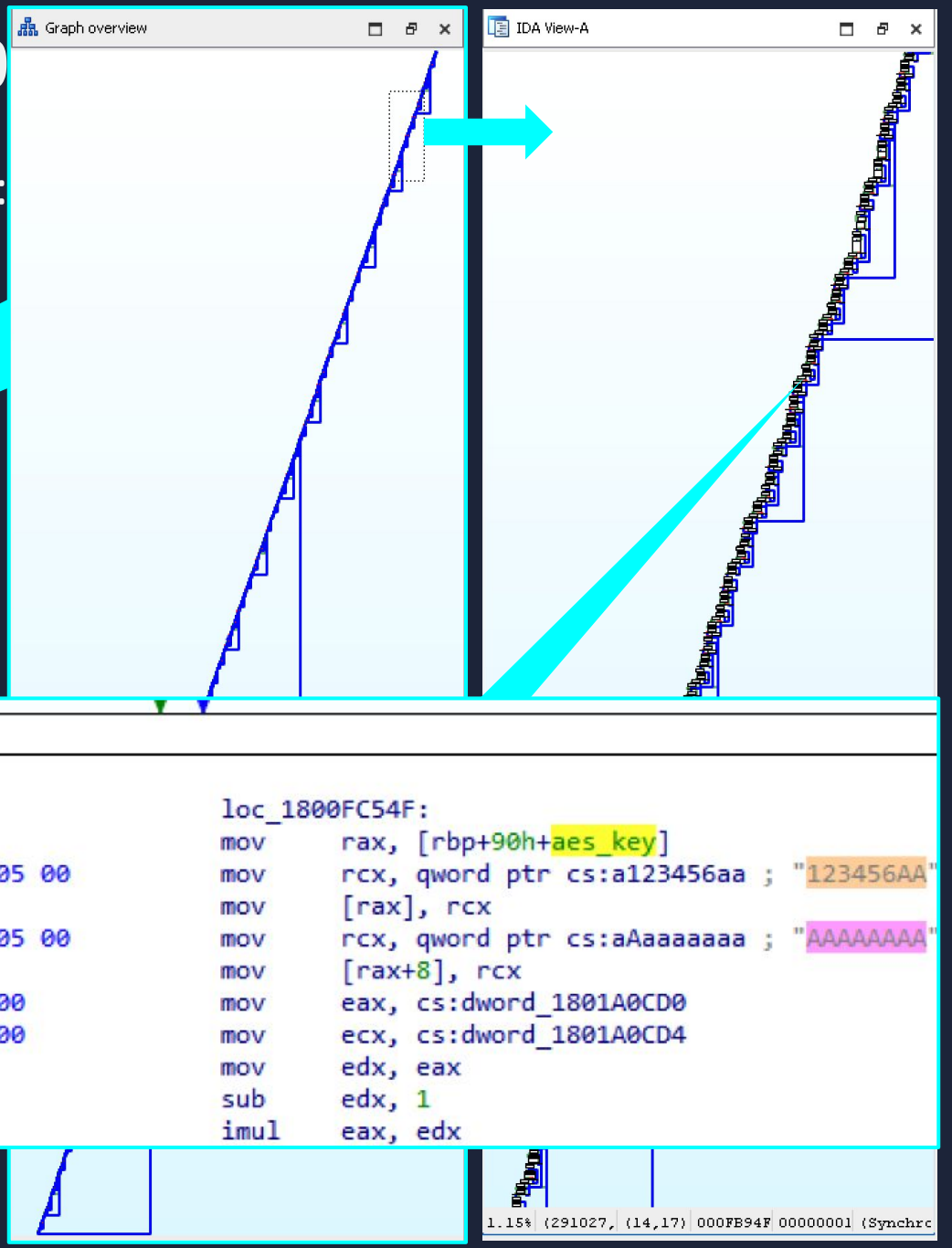
- a simple loader
- AES ECB mode
- hardcoded key = "123456AAAAAAAAAA"
- the HUGE Junk code



Xiangoop Loader: Variant AJ in Mar 20

Xiangoop Loader AJ is almost the same as the variant A:
- a simple loader
- AES ECB mode
- hardcoded key = "123456AAAAAAAAAA"





Xiangoop Loader: Variant AJ in Mar 2023

The biggest function size was 586 KB
How about using Hex-rays Decompiler?

Functions

Function name	Segment	Start	Length
sub_180001000	.text	0000000180001000	0008F414
sub_180090420	.text	0000000180090420	000368A2
sub_1800C6CD0	.text	00000001800C6CD0	0000B1E6
McVsoCfgGetObject	.text	00000001800D1EC0	00004618
sub_1800D64E0	.text	00000001800D64E0	00004618
sub_1800DAB00	.text	00000001800DAB00	00023A73
sub_1800FE580	.text	00000001800FE580	0000C223

Warning

Decompilation failure:
180001000: too big function

Please refer to the manual to find appropriate actions

☐ Don't display this message again (for this session only)

OK

does not work for HUGE func

works well for some small funcs

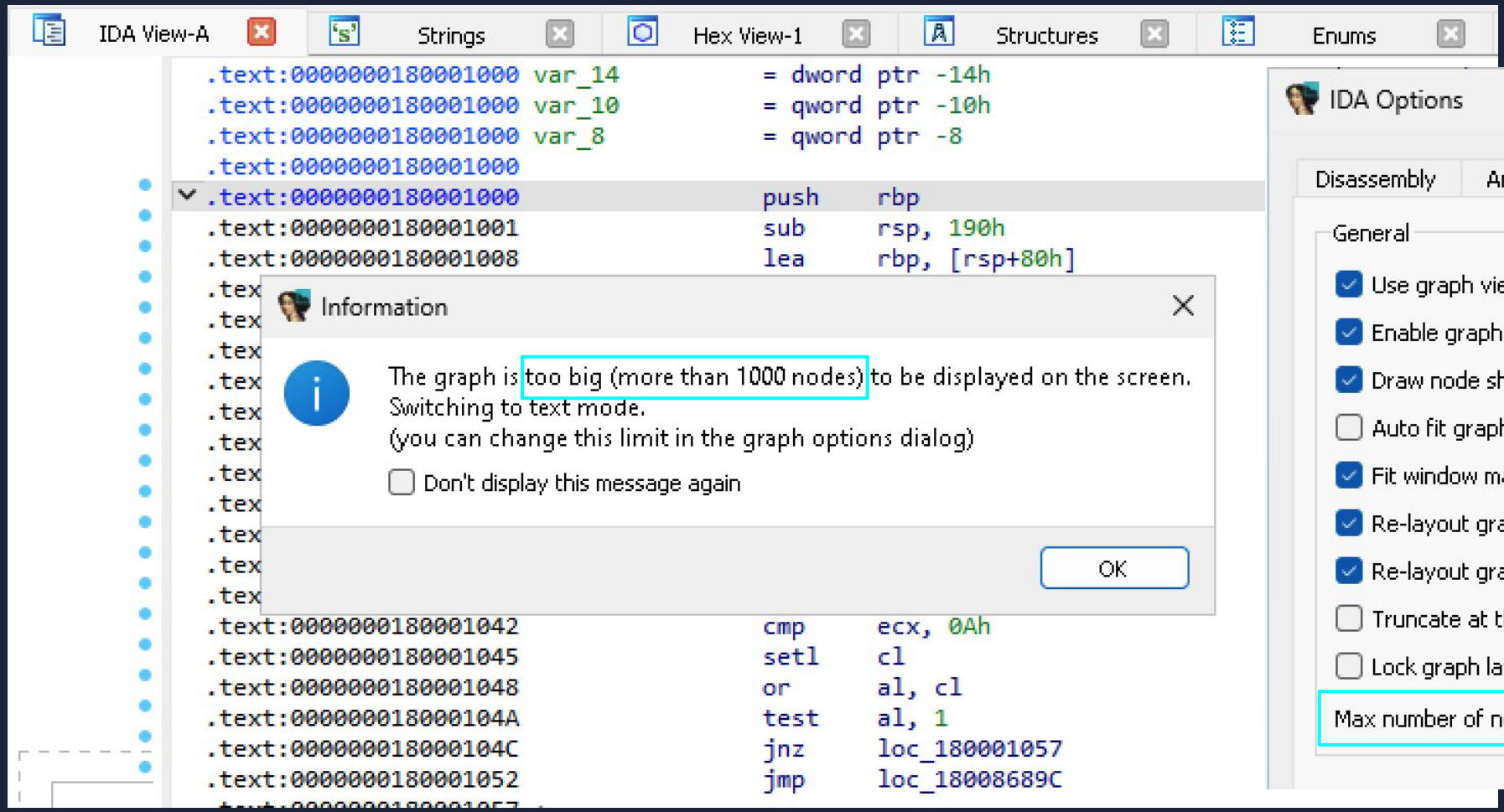
IDA View-A | Pseudocode-A

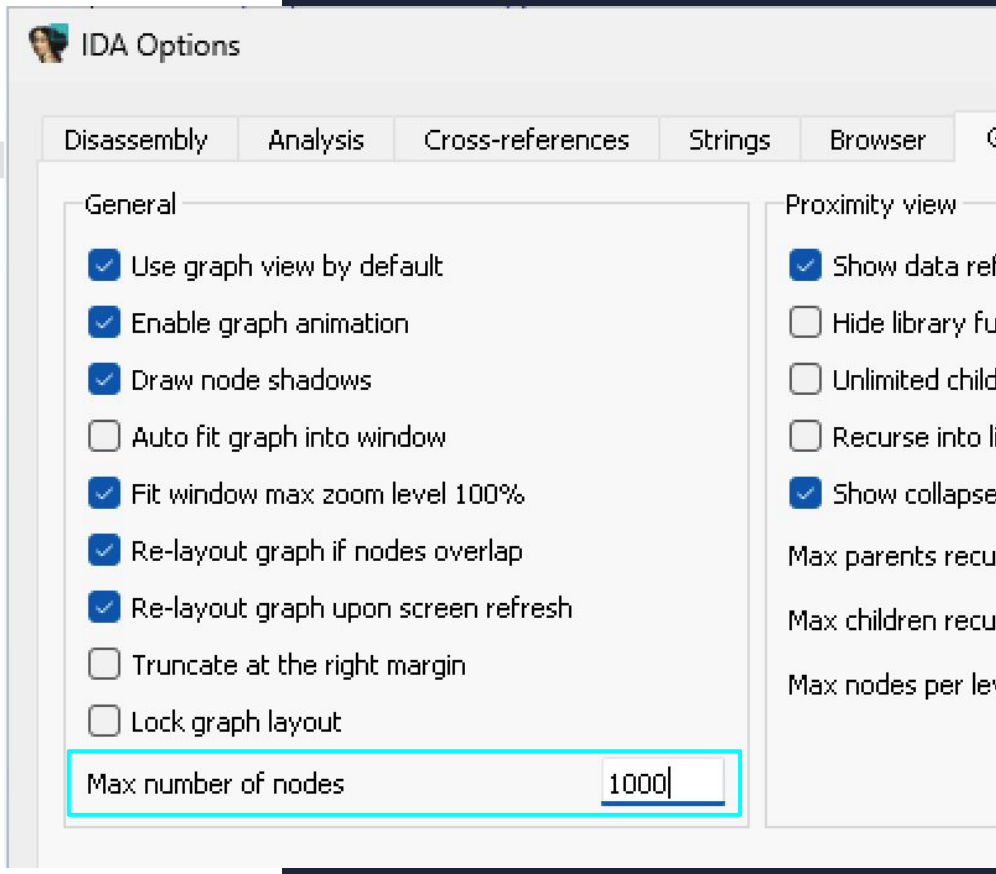
```
__int64 McVsoCfgGetObject()
{
  __int64 result; // rax

  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_ [covered by callout] CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( 1 )
  {
    sub_1800FE580();
    if ( dword_1801A0CE4 < 10 || ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) == 0 )
      break;
    sub_1800FE580();
  }
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
    ;
  while ( dword_1801A0CE4 >= 10 && ((((_BYTE)dword_1801A0CE0 - 1) * (_BYTE)dword_1801A0CE0) & 1) != 0 )
```

the junk pattern is the same

Xiangoop Loader: Variant AJ in Mar 2023

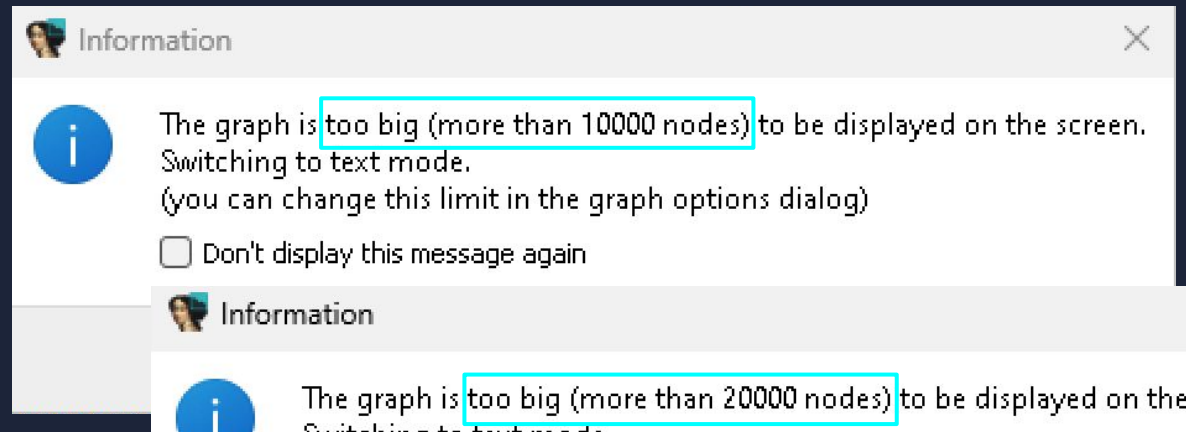
If you want to see a flow of this function, but...

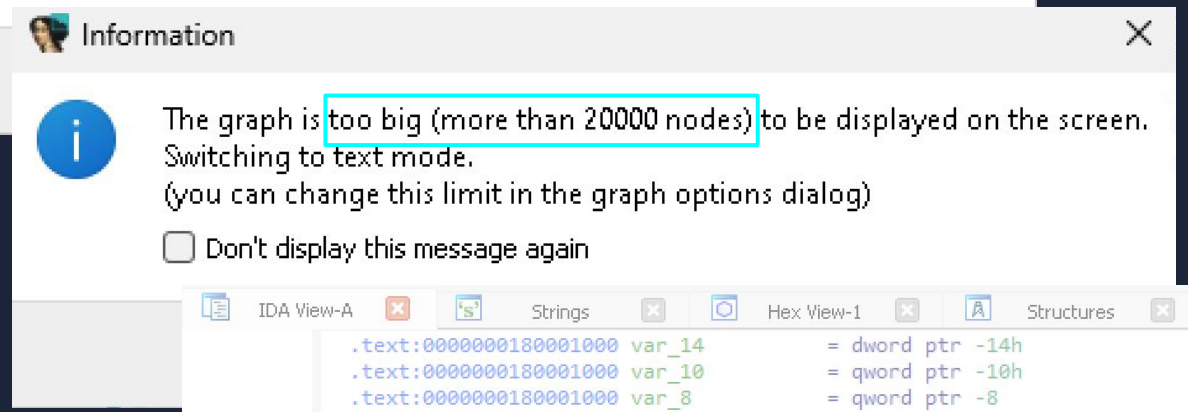


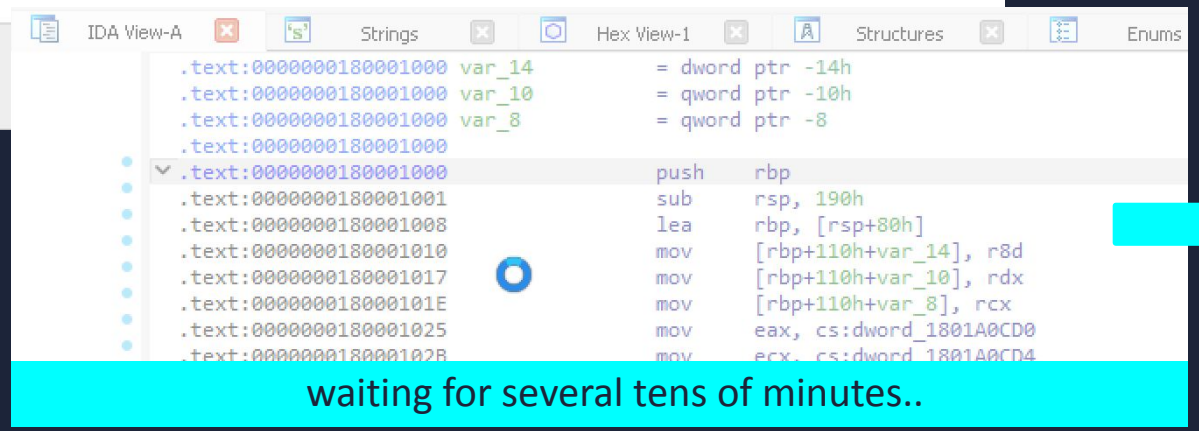


Xiangoop Loader: Variant AJ in Mar 2023

Tried to change the node limit to show flow in the IDA configuration!

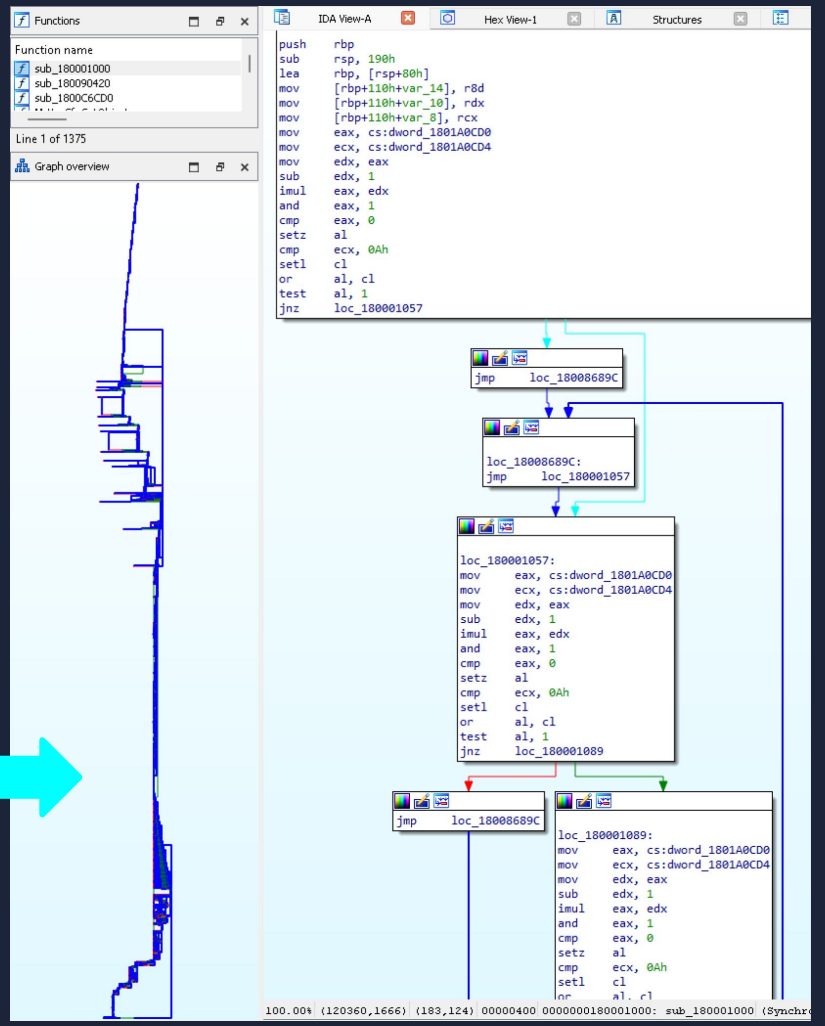






Node limit
-> 30,000

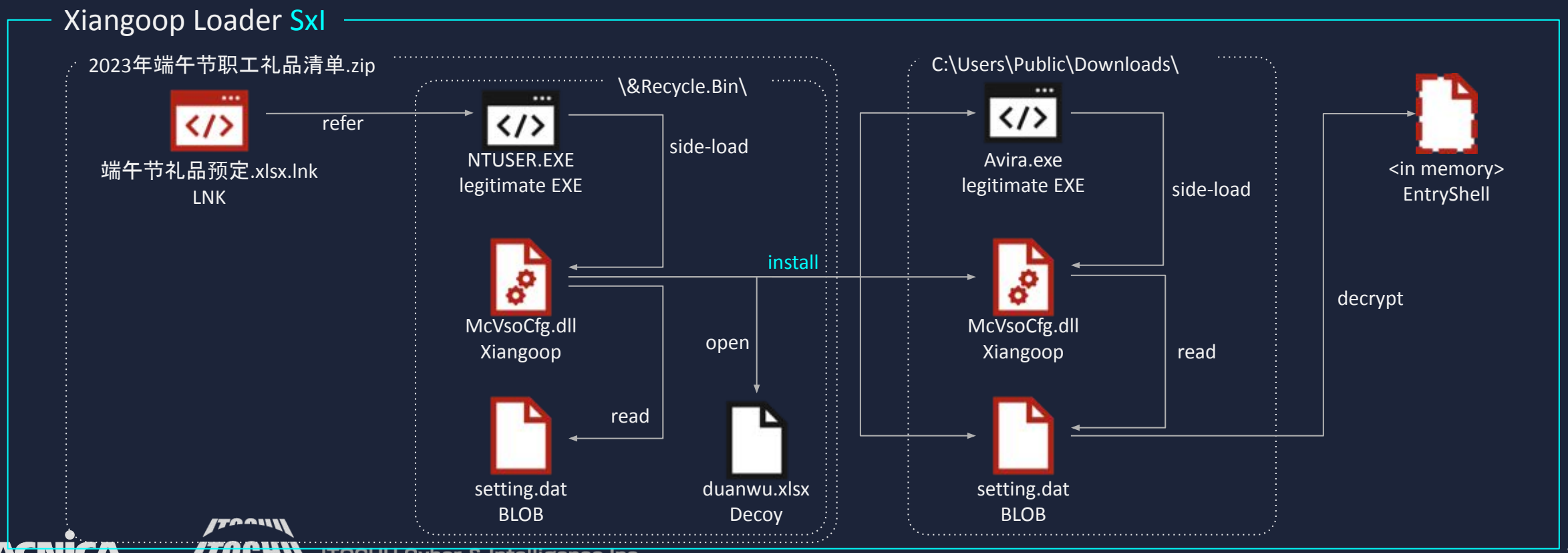
waiting for several tens of minutes..



Xiangoop Loader: Variant SxI in May 2023

loader feature + highly complex with uncommon crypto algorithms:
- Salsa20 to decrypt payload from BLOB
- x25519 + hsalsa20 + hsalsa20 to generate crypto key
- Poly-1305 to calculate check value for success of key generation

Installer feature was implemented



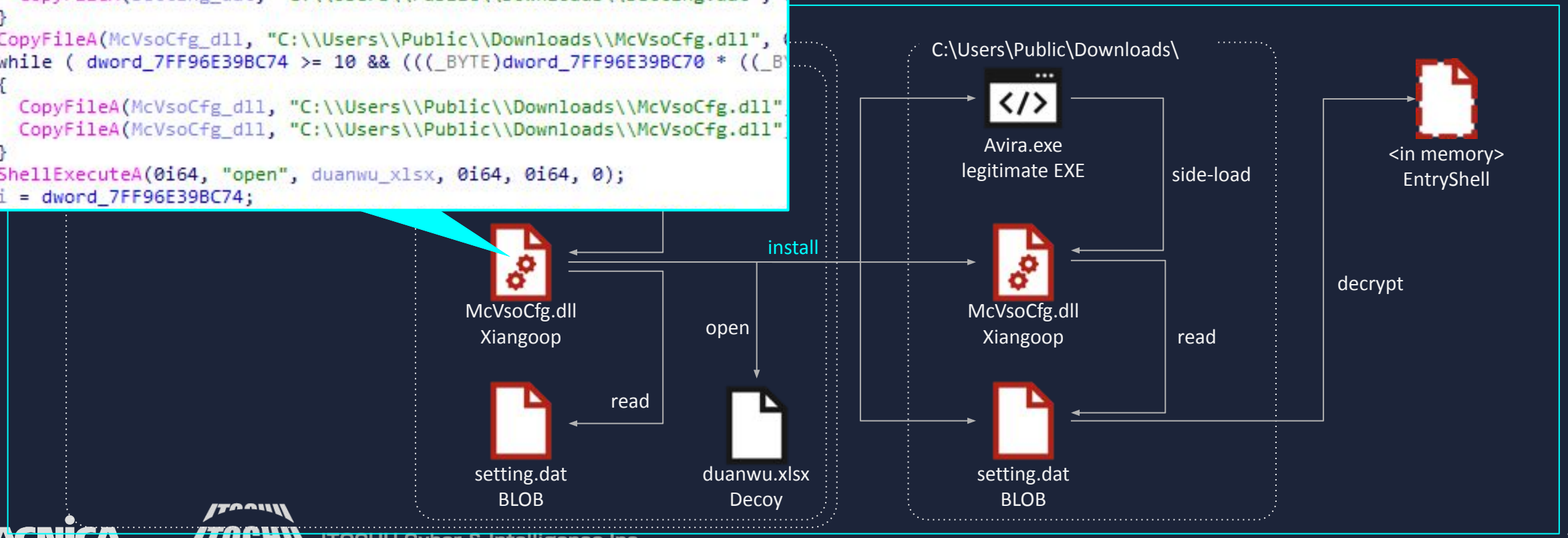
Xiangoop Loader: Variant SxI in May 2023

```
CopyFileA(current_exe, "C:\\Users\\Public\\Downloads\\Avira.exe", 0);
if ( dword_7FF96E39BC74 >= 10 && (((_BYTE)dword_7FF96E39BC70 * ((_BYTE
{
  while ( 1 )
    ;
}
CopyFileA(setting_dat, "C:\\Users\\Public\\Downloads\\setting.dat", 0)
while ( dword_7FF96E39BC74 >= 10 && (((_BYTE)dword_7FF96E39BC70 * ((_B
{
  CopyFileA(setting_dat, "C:\\Users\\Public\\Downloads\\setting.dat",
  CopyFileA(setting_dat, "C:\\Users\\Public\\Downloads\\setting.dat",
}
CopyFileA(McVsoCfg_dll, "C:\\Users\\Public\\Downloads\\McVsoCfg.dll",
while ( dword_7FF96E39BC74 >= 10 && (((_BYTE)dword_7FF96E39BC70 * ((_B
{
  CopyFileA(McVsoCfg_dll, "C:\\Users\\Public\\Downloads\\McVsoCfg.dll"
  CopyFileA(McVsoCfg_dll, "C:\\Users\\Public\\Downloads\\McVsoCfg.dll"
}
ShellExecuteA(0i64, "open", duanwu_xlsx, 0i64, 0i64, 0);
i = dword_7FF96E39BC74;
```

crypto algorithms:

ypto key

ss of key generation



Xiangoop Loader: Variant SxI in May 2023

```
gen_key1 = x25519(hardcoded_key2, hardcoded_key1)
                  ⇩
gen_key2 = hSalsa20(gen_key1, b"\x00" * 16)
                  ⇩
gen_key3 = hSalsa20(gen_key2, blob[0x00:0x10])
                  ⇩
dec1 = Salsa20_xor(gen_key3, blob[0x10:0x18], b"\x00" * 32 +
blob[0x28:0x48])
                  ⇩
gen_key4 = dec1[0x00:0x20]
                  ⇩
if (poly1305(blob[0x28:0x45a00], gen_key4) == blob[0x18:0x28]):
                  ⇩
    dec2 = Salsa20_xor(gen_key3, blob[0x10:0x18], blob[0x48:0x459e0])
                  ⇩
    payload = dec1[0x20:0x40] + dec2
```

McVsoCfg.dll

hardcoded_key2 =
73 7A A0 25 F2 0E 49 6B 6C F9 FA B1 6C 6F 1D 60
10 8F 05 2A 94 23 72 E1 F8 34 2A 79 9A E9 98 08

hardcoded_key1 =
45 57 F9 4E 14 71 06 03 7A 54 89 94 A8 98 84 8B
21 81 B6 9D B1 6A 4E 99 56 55 FC BC BA FD 2A 59

setting.dat

blob[0x00:0x10] =
1A DB 15 6E DE A9 44 69 E9 96 BD 73 DD E1 8E 10

blob[0x10:0x18] =
A7 C0 A9 C3 82 AA 83 4F

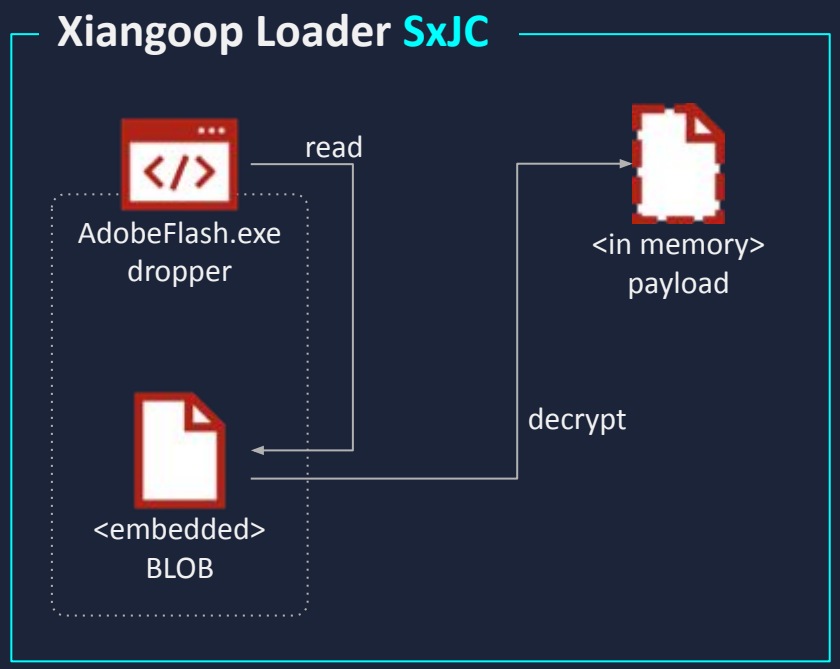
blob[0x18:0x28] =
3B 0E AA E1 04 F9 34 F3 04 FE 0A 05 46 AC 5A D4

blob[0x18:0x28] =
3B 0E AA E1 04 F9 34 F3 04 FE 0A 05 46 AC 5A D4

blob[0x28:0x45a00] =
F0 C0 4E 94 06 77 B5 64 E4 82 2D 70 AD 86 32 1B
53 41 B7 FD 94 87 CF 30 0A 80 2D 68 D7 DB 56 D2
F8 66 BE B0 4D 7F 98 1B CE 98 60 73 53 57 17 2B
B4 CD 86 59 82 C0 97 A0 E1 83 CE 64 53 61 E1 D1
[skipped]

Xiangoop Loader: Variant SxJC in Jun 2023

Salsa20 to decrypt payload from BLOB
x25519 + hsalsa20 + hsalsa20 to generate crypto key
Poly-1305 to calculate check value for success of key generation
huge Junk code + C



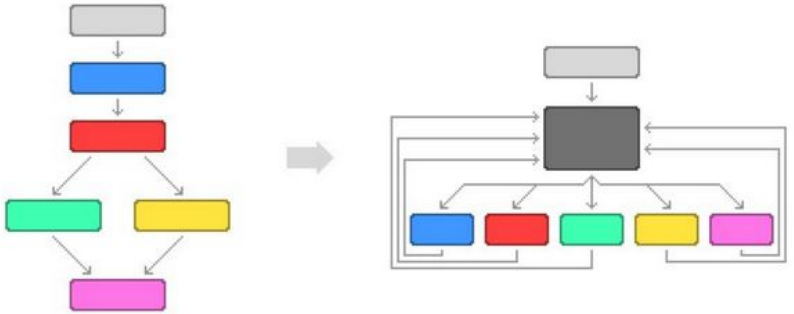
Control Flow Flattening

Control Flow Flattening aims to obfuscate the program flow by flattening it.

To achieve this, the transformation splits all the basic blocks of the source code, such as function body, loops, and conditional branches, and puts them all inside a **single infinite loop** with a switch statement that controls the program flow.

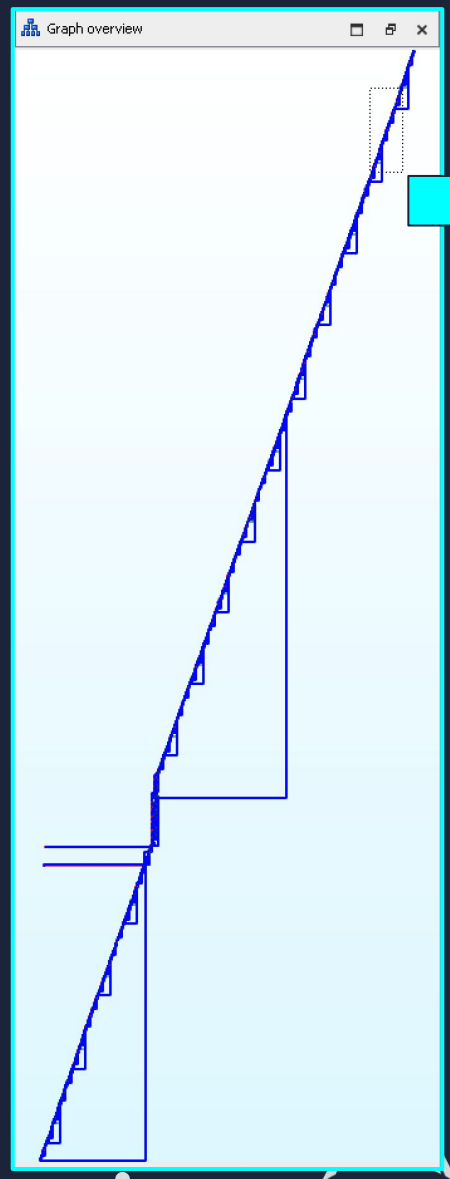
This makes the program flow significantly harder to follow because the natural conditional constructs that made the code easier to read are now gone.

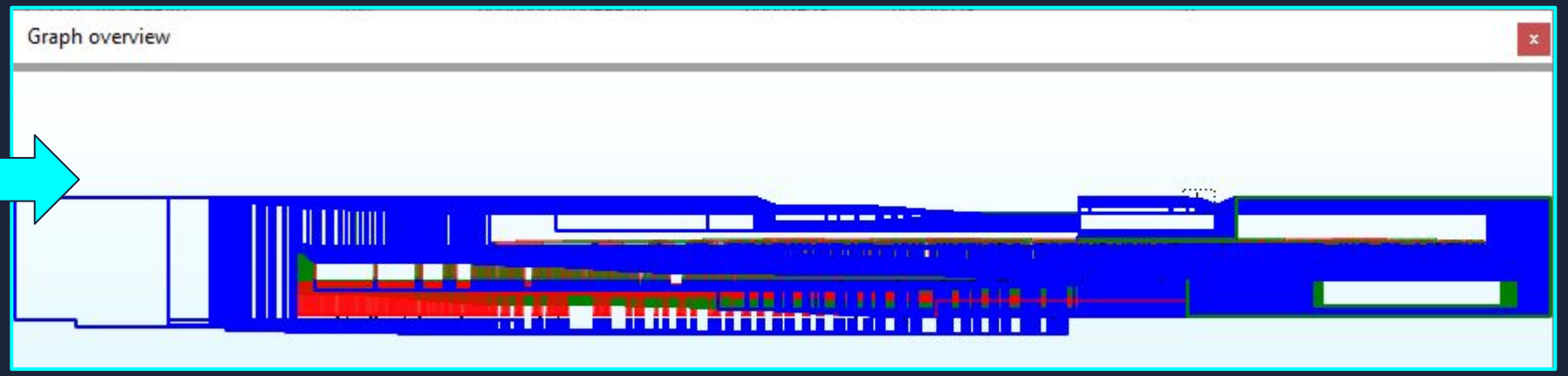
The following diagram is an abstract representation of what happens to control flow. It depicts a simplification of what a Control Flow Graph (CFG) would look like before (on the left) and after (on the right) flattening the program flow with a Static Code Analysis Tool.

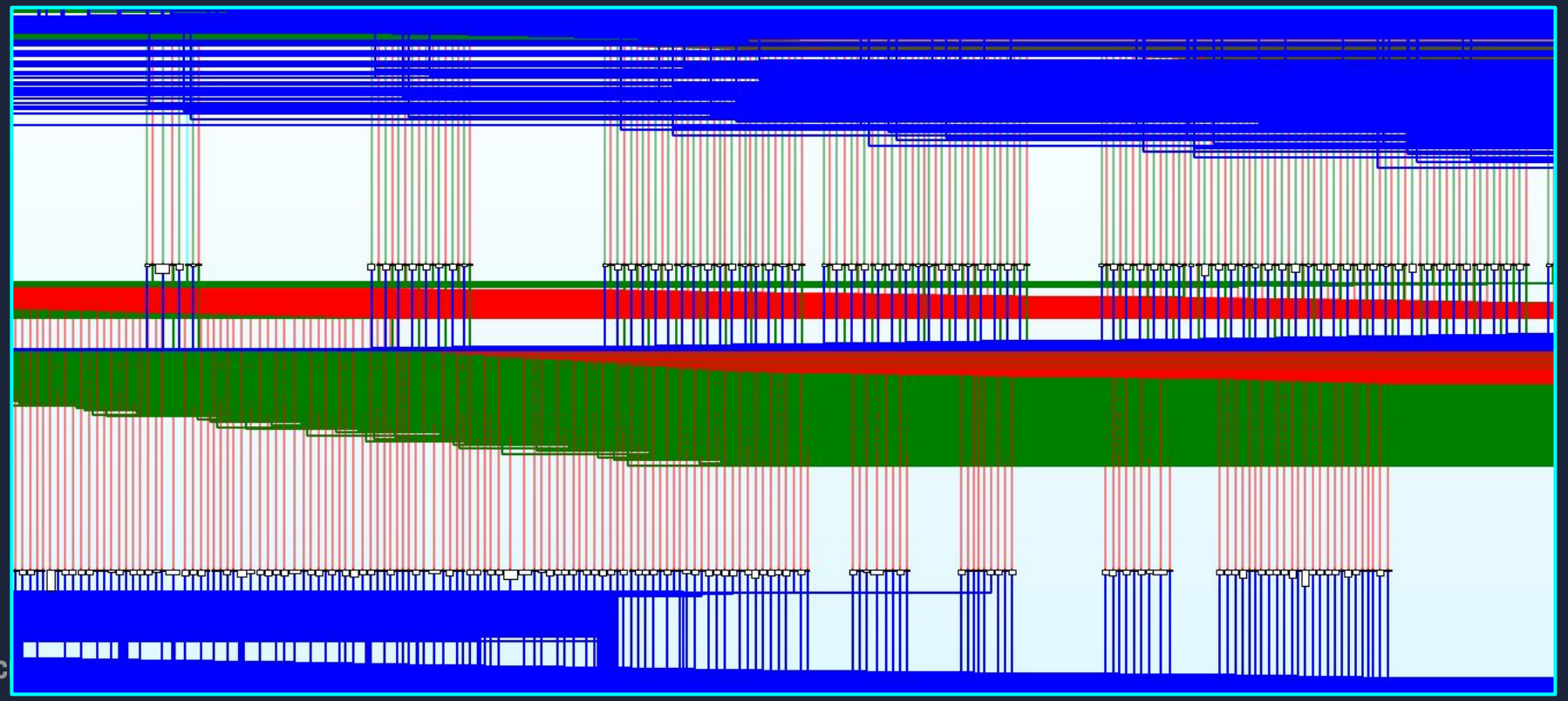


Ref-> https://jscrambler.com/blog/jscrambler-101-control-flow-flattening

Xiangoop Loader: Variant SxJC in Jun 2023

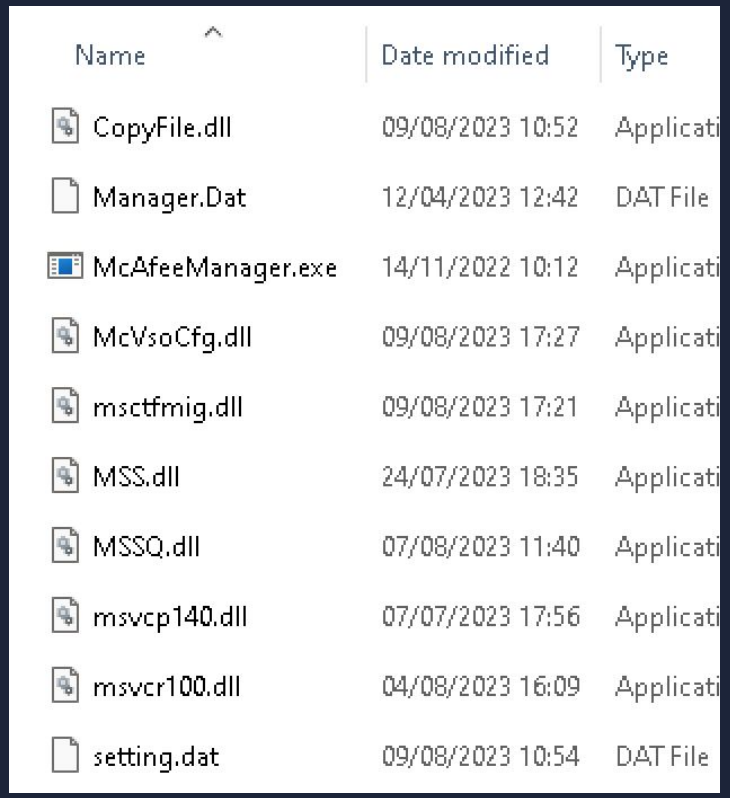


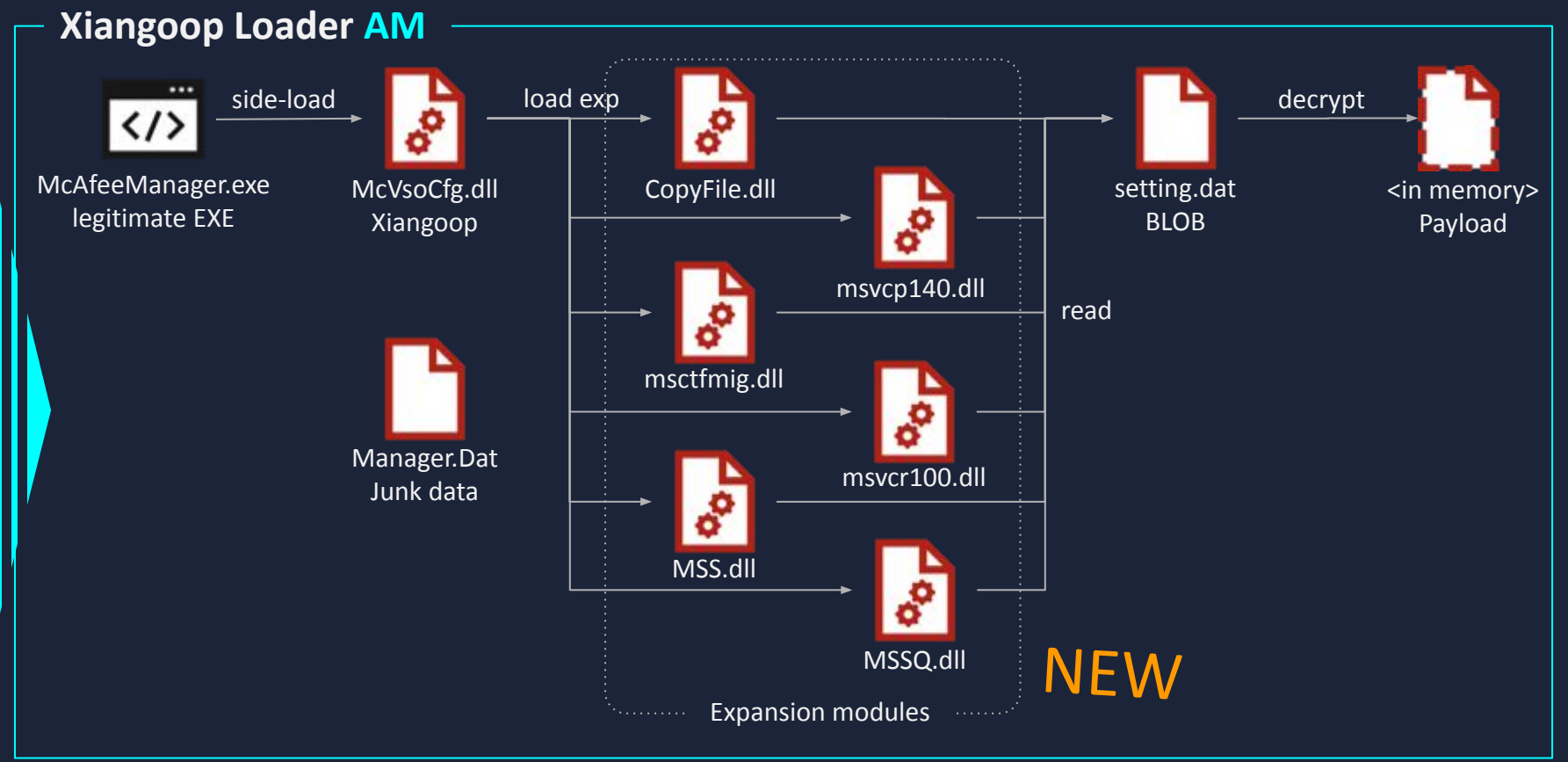




Xiangoop Loader: Variant AM in Aug 2023

Xiangoop Loader AM is very similar to the variant A
AES ECB mode + hardcoded key was updated “1234567890123456”
Malicious functions are divided into Multiple DLLs as expansion modules





Xiangoop Loader: Variant AM in Aug 2023

McVsoCfg.dll

```
                        cs:?sub_180006510@@YAXPEAX_KK@Z ; sub_180006510(void *,unsigned _
                        r9d, r9d        ; lpSecurityAttributes
                        [rsp+12B8h+hTemplateFile], rsi ; hTemplateFile
.text:0000000180001325 mov      [rsp+12B8h+dwFlagsAndAttributes], 80h ; dwFlagsAndAttributes
.text:000000018000132D lea      rcx, [rsp+12B8h+Filename] ; lpFileName
.text:0000000180001332 mov      edx, 80000000h  ; dwDesiredAccess
.text:0000000180001337 mov      [rsp+12B8h+dwCreationDisposition], 3 ; dwCreationDisposition
.text:000000018000133F lea      r8d, [r9+1]     ; dwShareMode
.text:0000000180001343 call     cs:CreateFileW
.text:0000000180001349 mov      rcx, rax        ; hFile
.text:000000018000134C xor      edx, edx        ; lpFileSizeHigh
.text:000000018000134E mov      rbx, rax
.text:0000000180001351 call     cs:GetFileSize
.text:0000000180001357 mov      rcx, rbx        ; hObject
.text:000000018000135A mov      edi, eax
.text:000000018000135C call     cs:CloseHandle
.text:0000000180001362 lea      rcx, [rsp+12B8h+Filename] ; currentdir_setting.dat
.text:0000000180001367 call     cs:?qwertyu@@YAPEADPEB_W@Z ; qwertyu(wchar_t const *)
.text:000000018000136D mov      rcx, rax
.text:0000000180001370 mov      edx, edi
.text:0000000180001372 mov      rsi, rax
.text:0000000180001375 call     cs:?AES@@YAXPEAX_K@Z ; AES(void *,unsigned __int64)
.text:000000018000137B mov      rcx, [rsp+12B8h+arg_0] ; void *
.text:0000000180001383 mov      r8d, edi        ; Size
.text:0000000180001386 mov      rdx, rsi        ; Src
.text:0000000180001389 call     memmove
.text:000000018000138E mov      rcx, [rsp+12B8h+arg_0]
.text:0000000180001396 call     cs:?sub_180006512@@YAX_K@Z ; sub_180006512(unsigned __int64)
.text:000000018000139C lea      r11, [rsp+12B8h+var_8]
```

msvcr100.dll

```
                        edx, edx        ; Val
                        rcx, [rsp+64978h+Buffer] ; void *
                        r8d, 64900h     ; Size
.text:0000000180001104 call     memset
.text:0000000180001109 lea      rax, _guard_check_icall_nop
.text:0000000180001110 mov      r8d, 648FFh     ; nNumberOfBytesToRead
.text:0000000180001116 lea      r9, [rsp+64978h+Overlapped] ; lpOverlapped
.text:000000018000111B mov      qword ptr [rsp+64978h+dwCreationDisposition], rax ; lpCompletionRoutine
.text:0000000180001120 lea      rdx, [rsp+64978h+Buffer] ; lpBuffer
.text:0000000180001125 mov      rcx, rbx        ; hFile
.text:0000000180001128 call     cs:ReadFileEx
.text:000000018000112E test     eax, eax
.text:0000000180001130 jnz      short loc_180001146
```

msctfmig.dll

```
                        [rsp+arg_8], rdx
                        [rsp+arg_0], rcx
.text:0000000180002ABA sub      rsp, 248h
.text:0000000180002AC1 mov      rax, cs:__security_cookie
.text:0000000180002AC8 xor      rax, rsp
.text:0000000180002ACB mov      [rsp+248h+var_18], rax
.text:0000000180002AD3 mov      [rsp+248h+aes_key], 31h ; '1'
.text:0000000180002ADB mov      [rsp+248h+var_27], 32h ; '2'
.text:0000000180002AE3 mov      [rsp+248h+var_26], 33h ; '3'
.text:0000000180002AEB mov      [rsp+248h+var_25], 34h ; '4'
.text:0000000180002AF3 mov      [rsp+248h+var_24], 35h ; '5'
.text:0000000180002AFB mov      [rsp+248h+var_23], 36h ; '6'
.text:0000000180002B03 mov      [rsp+248h+var_22], 37h ; '7'
.text:0000000180002B0B mov      [rsp+248h+var_21], 38h ; '8'
.text:0000000180002B13 mov      [rsp+248h+var_20], 39h ; '9'
.text:0000000180002B1B mov      [rsp+248h+var_1F], 30h ; '0'
.text:0000000180002B23 mov      [rsp+248h+var_1E], 31h ; '1'
.text:0000000180002B2B mov      [rsp+248h+var_1D], 32h ; '2'
                        B33 mov      [rsp+248h+var_1C], 33h ; '3'
                        B3B mov      [rsp+248h+var_1B], 34h ; '4'
                        B43 mov      [rsp+248h+var_1A], 35h ; '5'
                        B4B mov      [rsp+248h+var_19], 36h ; '6'
                        B53 mov      r8d, 10h
                        B59 lea      rdx, [rsp+248h+aes_key]
.text:0000000180002B61 lea      rcx, [rsp+248h+var_218]
.text:0000000180002B66 call     aes_init
.text:0000000180002B6B mov      [rsp+248h+var_224], 0
```

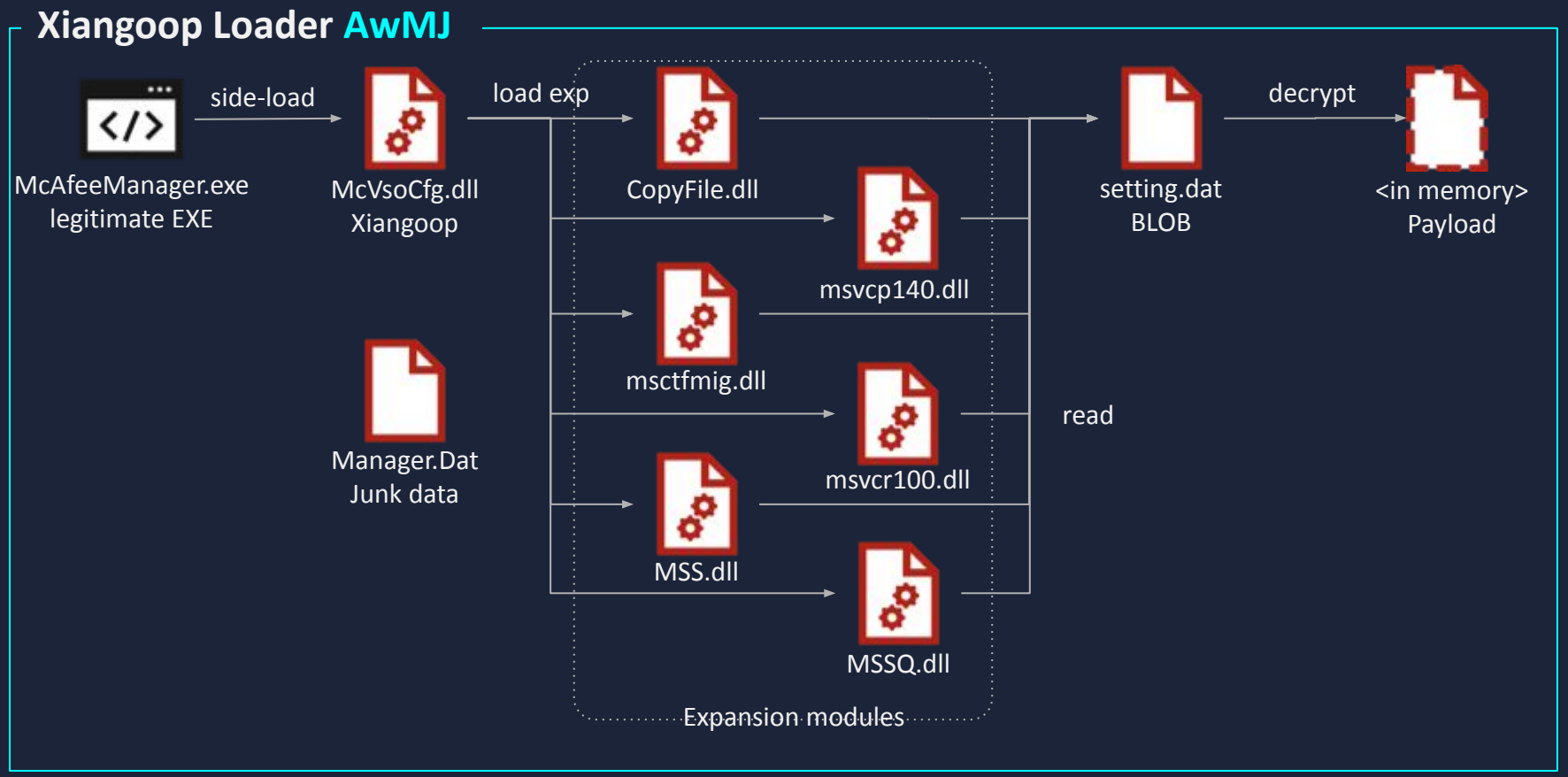
MSSQ.dll

```
                        10D0 mov      [rsp+arg_0], rcx
                        10D5 sub      rsp, 38h
.text:00000001800010D9 mov      rax, [rsp+38h+arg_0]
.text:00000001800010DE mov      cs:qword_18001ABF0, rax
.text:00000001800010E5 lea      rdx, Handler    ; Handler
.text:00000001800010EC mov      ecx, 1          ; First
.text:00000001800010F1 call     cs:AddVectoredExceptionHandler
.text:00000001800010F7 mov      [rsp+38h+var_18], 1
.text:0000000180001100 mov      [rsp+38h+var_10], 2
```

The new AES key = “1234567890123456”

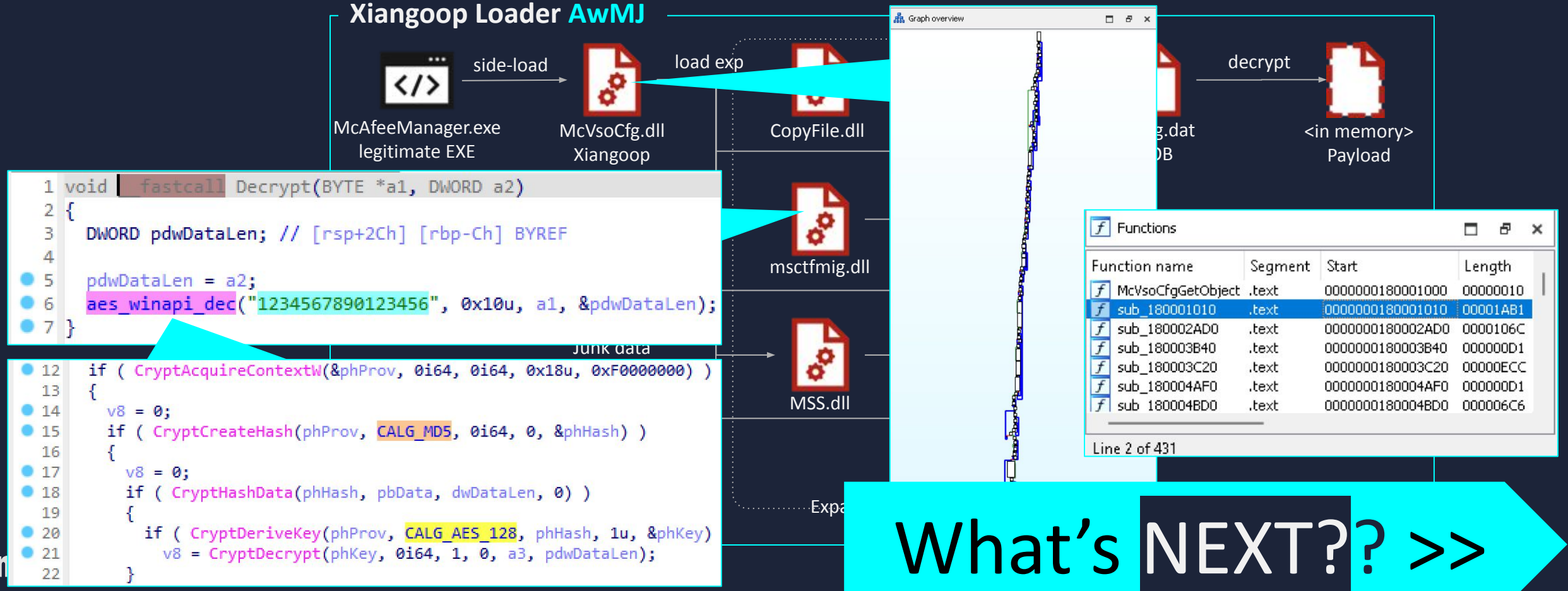
Xiangoop Loader: Variant AwMJ in Aug 2023

Xiangoop Loader AwMJ appeared immediately after the variant AM
AES ECB mode using windows crypto API + hardcoded key is “1234567890123456”
Divided into Multiple DLLs as expansion modules + huge Junk code



Xiangoop Loader: Variant AwMJ in Aug 2023

Xiangoop Loader AwMJ appeared immediately after the variant AM
AES ECB mode using windows crypto API + hardcoded key is “1234567890123456”
Divided into Multiple DLLs as expansion modules + huge Junk code




```
1  void __fastcall Decrypt(BYTE *a1, DWORD a2)
2  {
3    DWORD pdwDataLen; // [rsp+2Ch] [rbp-Ch] BYREF
4
5    pdwDataLen = a2;
6    aes_winapi_dec("1234567890123456", 0x10u, a1, &pdwDataLen);
7  }
```

```
12   if ( CryptAcquireContextW(&phProv, 0i64, 0i64, 0x18u, 0xF0000000) )
13   {
14     v8 = 0;
15     if ( CryptCreateHash(phProv, CALG_MD5, 0i64, 0, &phHash) )
16     {
17       v8 = 0;
18       if ( CryptHashData(phHash, pbData, dwDataLen, 0) )
19       {
20         if ( CryptDeriveKey(phProv, CALG_AES_128, phHash, 1u, &phKey)
21           v8 = CryptDecrypt(phKey, 0i64, 1, 0, a3, pdwDataLen);
22       }
```

Xiangoop Loader
> payloads Cobalt Strike Beacon

Cobalt Strike Beacon: Config

BeaconType - HTTPS
Port - 8443
SleepTime - 3000
MaxGetSize - 1398104
Jitter - 10
MaxDNS - Not Found
PublicKey_MD5 - 804f157f4176fea2d1ac6db1251ae37b
C2Server - cdn[.]cloudfarle[.]com,/filemaneger/ #Modified . > [.]
UserAgent - Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
HttpPostUri - /auth/
Malleable_C2_Instructions - Base64 decode
.....
HttpGet_Verb - GET
HttpPost_Verb - POST
HttpPostChunk - 0
Spawnto_x86 - %windir%\syswow64\svchost.exe -k wksvc
Spawnto_x64 - %windir%\sysnative\svchost.exe -k netsvc
CryptoScheme - 0
Proxy_Config - Not Found
Proxy_User - Not Found
Proxy_Password - Not Found
Proxy_Behavior - Use IE settings
Watermark_Hash - Not Found
Watermark - 520

Config is encoded with xor 0x2E, version 4

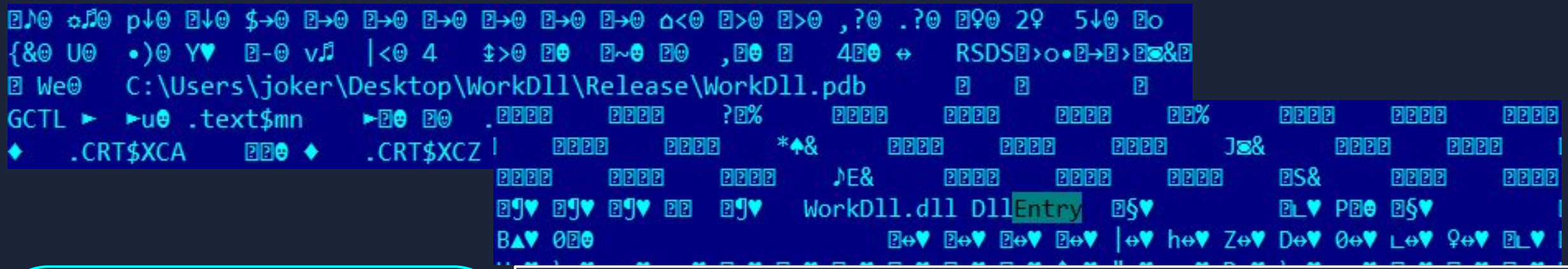
Watermark value is consistent

Examples) UserAgent of other Xiangoop loader's Cobalt Strike Beacon
Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; InfoPath.2; InfoPath.3)

Xiangoop Loader
> payload: EntryShell

What is EntryShell ?

EntryShell is a memory DLL that has only one Export function ‘DllEntry’, a variant of KeyBoy



Updates
- String obfuscation
- Config encoding, field
- C2 traffic
- Header code, Command ID
- Junk codes

```
$ yara ~/programs/yara/keyboy.yara /Avira.dmp -s -m
new_keyboy_header_codes [author="Matt Brooks, @cmatthewbrooks",desc="Matches
the 2016 sample\'s header
codes",date="2016-08-28",md5="495adb1b9777002ecfe22aaf52fcee93"] /Avira.dmp
0x28a941:$s1: *\x00l\x00*\x00
0x28a9c1:$s2: *\x00a\x00*\x00
0x28aa41:$s3: *\x00s\x00*\x00
...
keyboy_commands .... /Avira.dmp
keyboy_errors ... /Avira.dmp
keyboy_systeminfo ... /Avira.dmp
```

EntryShell: String obfuscation

```
'7FA2B5F545A273FA559A0382BACE839C'  'EEF7B4B46085911C147DBA7BF791CAD2'
'D040C8DBF7A4794FE079B2EEDC232062'  'BCC4F3F3F5DCFDC5867769B95F42DE78'
'0B23F7A0EF08F7E5C6F6EE9882725B43‘  'C69EC9959DD00E211CBBA2BEF3BA6B5A'
'26D61F18CE7B1E058C384B5D243A2958'  '584D718898D2E88AE29EE23853C32814'
'ADD24D415FEC26FF2D321F50AB7574F8'  '16700D53FA061DE86A5F4D33ED3A5034'
'3184526F7FE518EC686692BDC47D5012'  '46E0F662E3622FDFB528AB9EB2DAE14B'
'56A1A6951DA31192FFAB15B950F1D468'  'B880EF5F77DB454B89439C1F3C157A5A'
'BFD14534C01E5123ED7D757383DB8AF5'
'E1B5F3A4F0831CF765AC96C1D59CB489'
'F06B09A64EB34CA7A8AAAAFE2BDDF2FB'
'85556311638D0E34F862DB5090896BC4'
'7E02395FC094912719C2B09F8624A26A'
'78AC92A56B2FEC8032146B651E0C21E8'
'2D7394D9317A1E339A1133A98C672726'
'91F49683EC34675763F4A04B29A12B7B'
'BCC7A6993A71D44E91783C037A75ED5B'
'A4CD25265631F38E039408F16AA6E9ED'
```

New

AES-128-ECB

key: ‘afkngaikfaf’ + null

```
'login_OK'      '*1*'
'Update'        '*a*'
'UpdateAndRun'  '*s*'
'Refresh'       '*d*'
'OnLine'        '*f*'
'Disconnect'    '*g*'
'Pw_Error'      '*h*'
'Pw_OK'
'Ctrl_End'
'Sysinfo'
'Download'
'UploadFileOk'
'RemoteRun'
'Computer'
'Shell'
'ChangeCfg'
'Cfg_Error'
```

EntryShell: Config

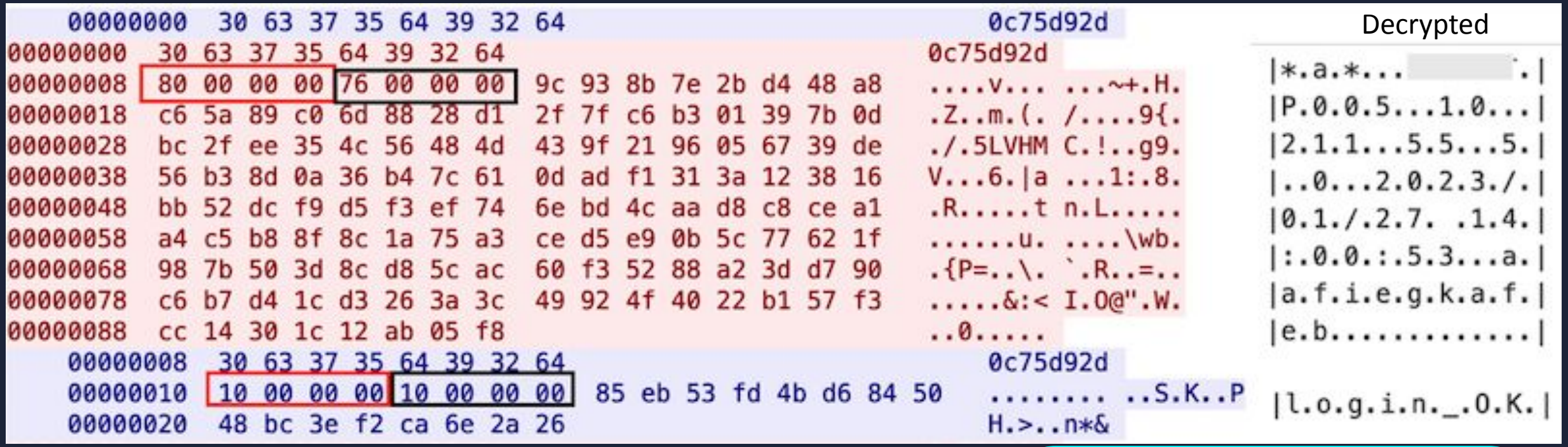
New encoding is applied, config is now hardcoded and possible dynamic update by ‘ChangeCfg’ request from C2 server

```
v_enc_mod = (v_enc & (1 << (7 - n_count))) != 0;
result = v_enc_mod + 2 * v_enc_next;
v_enc_next = v_enc_mod + 2 * v_enc_next;
```

Check Code (0123456789) \r\n C2 address #1 (85[.]209[.]43[.]142) \r\n C2 address #2 (0) \r\n C2 address #3 (0) \r\n Port Number #1 (4431) \r\n Port Number #2 (0) \r\n Port Number #3 (0) \r\n Password for C2 Operation (1003) \r\n Campaign ID (0) \r\n Proxy (0) \r\n Proxy Port (0) \r\n Proxy User (0) \r\n Proxy Password (0) \r\n

New

EntryShell: C2 traffic (TCP socket connection)



md5 AES-128-ECB key (63574154+null)

0c75d92d ➡ 88103b5663574154b95037cfa76f3dfe

4 byte	4 byte	
Length of Encrypted String	Length of Decrypted String (Wide)	Encrypted String (AES-128-ECB)

Header code: *a*
Computer Name: _P005
IP address: 10.211.55.5
Campaign ID: 0
Timestamp: 2023/01/27 14:00:53
Version Identifier: aafiegkafeb

EntryShell: Header code and Command ID

Header code	
a	Initial Connection to C2
d	Remote shell
l	Beacon
f	File Handling (e.g. Error2:)
s	Send system information
g	File Download (e.g. DownloadFile: Ready Download OK)
h	File Upload (e.g. UploadFile:)
(Empty)	Send Message (e.g. login_OK, Cfg_Error) New

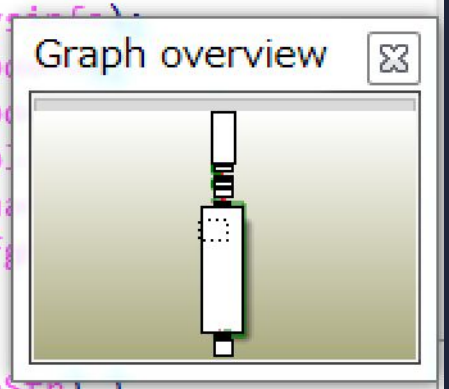
Command ID	
Sysinfo	Collect system information
Download	Upload file to C2
UploadFileOk	Download file from C2 and execute
RemoteRun	Execute file
Computer New (FileManager)	Collect drive information and list of files
Shell	Remote shell
cd	Change directory
dir or ls	Collect list of files in the folder
del	Delete file
Exit	Terminate C2 session

EntryShell: Junk code

Like Xiangoop loader, junk code is also sometimes applied to some subroutine

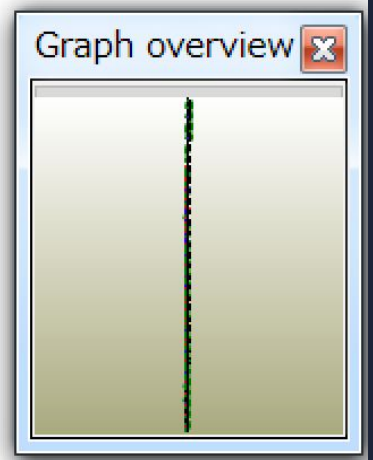
2022

```
56    My_AES_Round(a2d7394d9317a1e_0, aComputer);
57    My_AES_Round(a7fa2b5f545a273_0, aLoginOk);
58    My_AES_Round(aAdd24d415fec26_0, aOnline);
59    My_AES_Round(a56a1a6951da311_0, aPwError);
60    My_AES_Round(aBfd14534c01e51_0, aPwOk);
61    My_AES_Round(a26d61f18ce7b1e_0, aRefresh);
62    My_AES_Round(a78ac92a56b2fec_0, aRemoterun);
63    My_AES_Round(a91f49683ec3467_0, aShell);
64    My_AES_Round(aF06b09a64eb34c_0, aSy
65    My_AES_Round(aD040c8dbf7a479_0, aUp
66    My_AES_Round(a0b23f7a0ef08f7_0, aUp
67    My_AES_Round(a7e02395fc09491_0, aUp
68    My_AES_Round(aBcc7a6993a71d4_0, aCh
69    My_AES_Round(aA4cd25265631f3_0, aCf
70    mmsi(MultiByteStr, 0, 0x400u);
71    My_AES_0(aD64c1c26b4a279_0, v13);
72    if ( !My_str_chck_0(&v15, MultiByteStr) )
73      DllEntry(v11, v8, ebx0, 0);
74  }
75  return 1;
```



2023

```
2341        }
2342        else
2343        {
2344          v277 = v276 - 48;
2345        }
2346        if ( v277 < 0 )
2347          break;
2348        v273 += 2;
2349        *v272++ = v277 + 16 * v275;
2350        v274 = *v273;
2351      }
2352      while ( *v273 );
2353    }
2354    *v272 = 0;
2355  }
2356  My_AES_Round(qword_18005C9B0, &v281);
2357  v278 = v281;
2358  for ( i19 = MultiByteStr; v278; v278 = i19[&v281 - MultiB
2359    *i19++ = v278;
2360  *i19 = 0;
2361  return My_Subst(MultiByteStr, &Cfg_Error);
2362}
```



EntryShell: Wrap up (KeyBoy update)

Version Identifier	Key Changes
aafiegkafeb	• Added static string obfuscation by using AES-128-ECB with key: ‘afkngaikfaf’ • Config encoding update and proxy related values are added • Config is embedded and dynamic update by ‘ChangeCfg’ request from C2 server • C2 traffic data is encrypted with AES-128-ECB • Junk code is applied to some subroutines

CrowDoor


vb 2023 LONDON
Virus Bulletin International Conference
4 -6 October 2023
RED ROOM
VB2023
LONDON W6
VIRUSBULLETIN.COM
SOPHOSLABS

What is CrowDoor ?

‘CrowDoor’ is named after ‘SparrowDoor’

Found CrowDoor sample from the same victim during the Tropic Trooper incident response in June 2023

Similarities with SparrowDoor

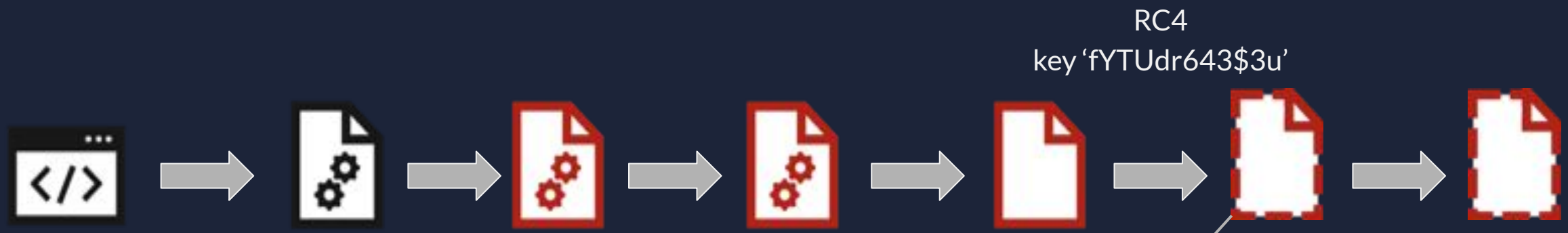
- Complete code overwraps at loader shellcode
- Actions by command line arguments
- Implementation of Command ID

Dissimilarities

- Config
- C2 Traffic

https://www.ncsc.gov.uk/files/NCSC-MAR-SparrowDoor.pdf
https://www.welivesecurity.com/2021/09/23/famoussparrow-suspicious-hotel-guest/

CrowDoor: Loader




```
$ yara ~/SparrowDoor.yar ./Shellcode_Payload.bin  -m -s
SparrowDoor_shellcode [author="NCSC",description="Targets code features of the reflective
loader for SparrowDoor.
Targeting in
memory.",date="2022-02-28",hash1="c1890a6447c991880467b86a013dbeaa66cc615f"]
0x5a:$peb: 8B 48 08 89 4D FC 8B 51 3C 8B 54 0A 78 8B 74 0A 20 03 D1 03 F1 B3 64
0x71:$getp_match: 8B 06 03 C1 80 38 47 75 34 80 78 01 65 75 2E 80 78 02 74 75 28 80 78 03 50
75 22 80 78 04 72 75 ...
0x36:$k_check: 8B 48 20 8A 09 80 F9 6B 74 05 80 F9 4B 75 05
0xda:$resolve_load_lib: C7 45 C4 4C 6F 61 64 C7 45 C8 4C 69 62 72 C7 45 CC 61 72 79 41 C7 45
D0 00 00 00 00 FF 75 FC FF ...
```

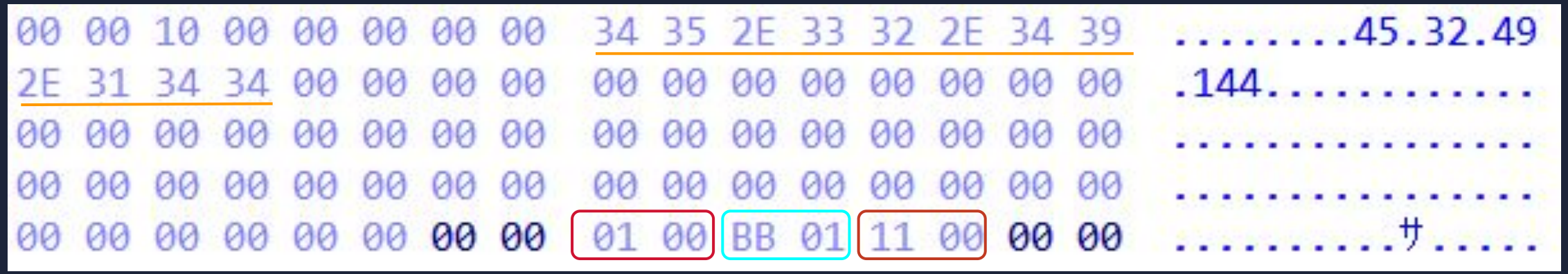
CrowDoor: Actions performed by command line argument

4 patterns are very similar to SparrowDoor's -i -k -d switch

Argument or flag (Value after parsing command line argument)	Action
No argument	Persistence is set through the registry Run key or a service and the backdoor is restarted
0	Persistence is set through the registry Run key or a service and the backdoor is restarted
1	The backdoor is restarted by injecting to 'colorcpl.exe'
2	The backdoor interpreter is called

CrowDoor: Config

Config is hardcoded without encoded



45_32_49_144: C2 IP address

0x01: Connection method

- 0x1 : TCP socket connection (Hardcoded at found sample)
- 0x2 : TCP socket connection, do communication thread with lowest priority
- 0x3 : TCP socket connection and set header string like below, do communication thread with lowest priority

'User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) Chrome/107.0'

0x1BB: Port Number (443)

0x11: Number of retries for C2 connection

CrowDoor: C2 traffic

C2 Communication is done with TCP socket connection and encrypted with RC4. Initial 16 bytes are encrypted with hardcoded key ‘fYTUdr643$3u’ and additional data with a randomly generated key.

```
00000000  a9 7c 46 50 b8 a6 cf 27  8a 9f 37 1d 20 21 07 62   .|FP...' ..7. !.b
00000010  ea b4 67 db b7 9d 8a                              ..g....
```

Decrypt initial 16 bytes with key ‘fYTUdr643$3u’

random key | Command id | Length

19 28 bb 9a | 35 71 34 02 | 07 00 00 00 | 00 00 00 00

Little Endian

‘ 9abb2819’ ‘0x2347135’ ‘0x7’

```
00000010  ea b4 67 db b7 9d 8a                              ..g....
```

Decrypt additional data with key ‘9abb28199abb2819’

2E 6E E2 AE D6 38 78

CrowDoor: Command ID

Command ID	Action
0x2347135	Initial connection to C2
0x2347136	Collect ComputerName, UserName, OS version and hostnet or IP address information
0x2347137	Remote Shell
0x234713B	Delete malware files, persistence and exit
0x2347140	File related operation
0x2347141	Read file
0x2347142	Write file
0x2347144	Collect drive information
0x2347145	Search file
0x2347148	Create directory
0x2347149	Rename file or directory
0x234714A	Delete file or directory
0x234714B	Communication with C2

CrowDoor: Similarities with SparrowDoor

SparrowDoor

```
case 0x1A6B561Au:
  v43 = 0;
  memset(&v44, 0, 0x206u);
  MultiByteToWideChar_0(0xFDE9u, 0, v16, v15, &v43, 260);
  CreateDirectoryW(&v43, 0);
  break;
case 0x18695638u:
  My_RENAME(v16);
  break;
case 0x196A5629u:
  My_DeleteFile(v16, v
  break;
```

```
v11 = 0;
v12 = 0;
v13 = 0;
v6 = 0;
v10 = 28;
v7 = FO_RENAME;
v8 = &v18;
v9 = &v20;
return SHFileOperationW(&v6);
```

```
v6 = 0;
v7 = 20;
v4 = FO_DELETE;
v5 = &v11;
return SHFileOperationW(&v3);
```

CrowDoor

```
case 0x2347148:                         // Create Directory
  memset(&v42, 0, 0x208u);
  MultiByteToWideChar_0(0xFDE9u, 0, (v4 + 16), *(v4 + 8), &v42, 260);
  if ( !CreateDirectoryW(&v42, 0) )
    GetLastError_0();
  continue;
case 0x2347149:                         // Rename File or Directory
  memset(&v46, 0, 0x104u);
  memset(&v47, 0, 0x104u);
  v21 = *(v4 + 16);
  mm(&v46, v4 + 17, v21);
  mm(&v47, v4 + v21 + 18, *(v21 + v4 + 17));
  memset(&v41, 0, 0x208u);
  memset(&v40, 0, 0x208u);
  MultiByteToWideChar = *MultiByteToWideChar_0;
  MultiByteToWideChar_0(0xFDE9u, 0, &v46, -1, &v41, 260);
  MultiByteToWideChar(65001, 0, &v47, -1, &v40, 260);
  v34 = 0;
  LOWORD(v32) = 28;
  *(&v32 + 2) = 0i64;
  v30 = &v41;
  v31 = &v40;
  v28 = 0;
  v29 = FO_RENAME;
  v9 = &v51;
  if ( SHFileOperationW(&v28) )
    GetLastError_0();
  continue;
case 0x234714A:                         // // Delete File or Directory
  memset(&v39, 0, 0x208u);
  MultiByteToWideChar_0(0xFDE9u, 0, (v4 + 16), *(v4 + 8), &v39, 260);
```

Attribution

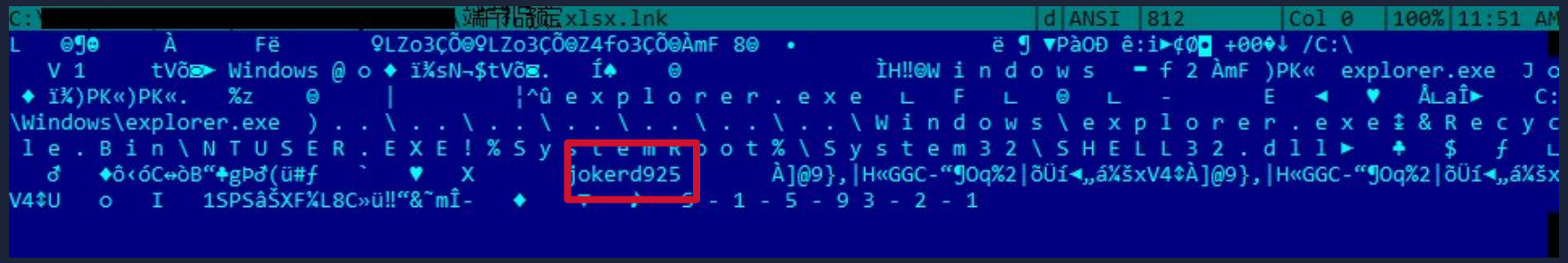
Attribution -Relationship between Tropic Trooper and FamousSparrow

- One event in this Tropic Trooper attack campaign confirmed the presence of CrowDoor used by FamousSparrow.
- From these series of activities, we first discovered that Tropic Trooper and FamousSparrow may be linked and operationally closely related.

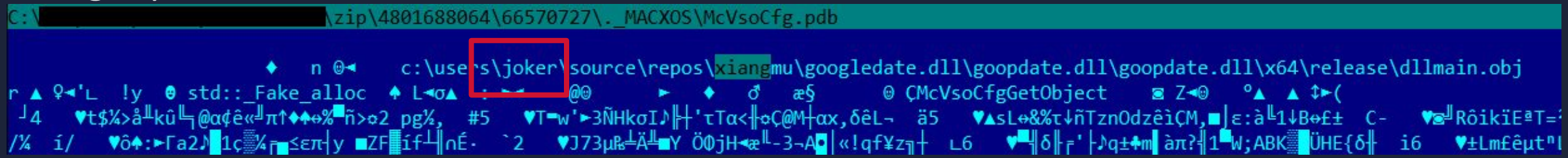
Attribution -Developer's name

The common developer name "**joker**" was left in the PDB files and attached .lnk files!!

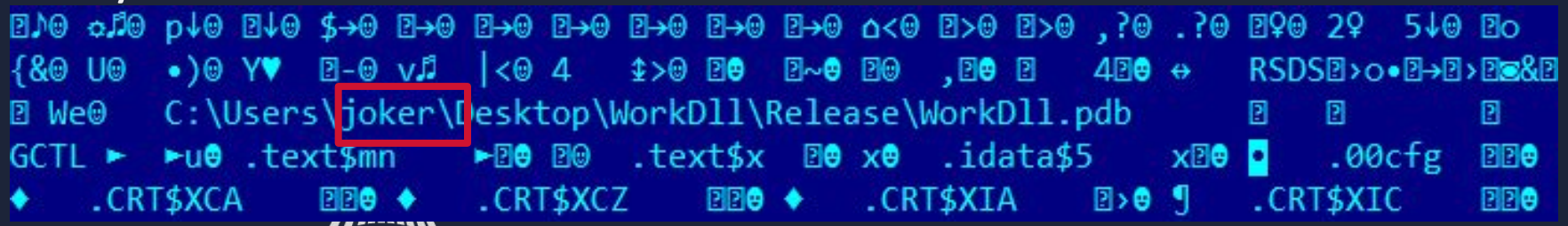
Attached .lnk file



Xiangoop Loader



Entry Shell



Conclusions

Conclusions

Tropic Trooper attack campaign from 2022 to 2023.

- Beware of physical penetration rather than malware infection by spear-phishing!!
- New malware, “Xiangoop Loader” and its transition to become anti-analysis, with the variety of encryption algorithm, junk code and CFF
- A variant of KeyBoy, "EntryShell”, updating command ID and its similar anti-analysis features such as obfuscated command ID and junk code as well
- New malware, “CrowDoor”, which is associated with FamousSparrow, and shared a detailed analysis of these features.

Finally...

A new version of Xiangoop Loader has been observed last week!

Prepare for TropicTroper attacks!


Thank you very much!
Enjoy VB2023

Appendix


vb 2023 LONDON
Virus Bulletin International Conference
4 -6 October 2023
RED ROOM
VB2023 LONDON W6
VIRUSBULLETIN.COM

IoCs: malware types and md5 hashes

Xiangoop Loader A	06d84cd9721bed541b5f59736b39d3a3
Xiangoop Loader A	1ffd397619bc1a5b2ff25a5067312077
Xiangoop Loader AM	53e402edb9196fa30bd1d0bb8e66bde6
Xiangoop Loader AM	c801c30657a15d398de36519f2072713
Xiangoop Loader AM module	28310ac912df6cd5f7d3ac27a6b5fafc
Xiangoop Loader AM module	29e9f8d8a1609c128e3088515714bb45
Xiangoop Loader AM module	4a168c51a6dcc0ac6df273c394133b68
Xiangoop Loader AM module	600229a19d26964fdbd2d0caf4a6dc84
Xiangoop Loader AM module	61b35baeb8dce981e3d692aedbaac6ae
Xiangoop Loader AM module	76d2f0e0f101cdec483d27c08a56ef87
Xiangoop Loader AM module	7d4ab23dbad10ab728c87666b6513e87
Xiangoop Loader AM module	bc322adb53f476cea28d9bd4992d9f02
Xiangoop Loader AM module	c9021f1801c19d1a0198d27ff2453b75
Xiangoop Loader AM module	ce7a7238e4ff2d28a3876777f12fbbd2
Xiangoop Loader AM module	e4e8f18c571ec28bb4fe7eba511926fe
Xiangoop Loader AM module	e687a1f959365a64e5ed7f5748f6a790
Xiangoop Loader AM module	fc3730e18dd09249d8a19b39f6a1ae80
Xiangoop Loader AwMJ	ccb5aa2057a157261606c043ce7d45e8
Xiangoop Loader AwMJ module	074b41bbc7018a139212f63331bb0a14
Xiangoop Loader AwMJ module	12be7a86877a561677d2cd63c2b7c19f
Xiangoop Loader AwMJ module	36a24ebf972ec1012eedee41863488a6
Xiangoop Loader AwMJ module	46048c962243c2999796b3a3fe525631
Xiangoop Loader AwMJ module	5f134ca309cacac5f6651c60f1eb0a78
Xiangoop Loader AwMJ module	e98be352638418900fd5378de14956be
Xiangoop Loader SxI	4006dcb60b94f22e313138d836f6692f
Xiangoop Loader SxI	bb01bc33b0475fb2624d906760ebe290

Xiangoop Loader AJ +BLOB	64294f4f6d9a91a7df44d36fa5c88651
Xiangoop Loader SxJC+BLOB	22879c3aabceed8968edcadce38fa2c6
Xiangoop Loader SxJC+BLOB	2f91d9ad3a03a4a8f99776f79830dc00
BLOB	47c7091a4eedc310b928b57347c57616
BLOB	84b6a4044b6a505c1d24f4cceba294d0
BLOB	998ae2fab40c911d321a398c911687e1
BLOB	b150e9258273e22eec5d49053147b956
BLOB	cccc4cf8267815cf7ae1f924ef2d9b83
BLOB	e66eab6fd531377392950d150da8061a
Cobalt Strike beacon	60343197c88fc483072b4875be70a9cb
Cobalt Strike beacon	cea56b3f18618c25eab43c4df5cae00c
EntryShell	7f2029336efc798486a8e35fe6a2c54a
msi file	924e3153b6788f012b7f8626410e1155
msi file	aa91030187a7fe89cf6f88018996813a
cab file	b97f02f0f7fd80b0265899e1b64cb09a
cab file	bd5b82e0f5bc53447f024a6cdc584b30
zip file	4ecde4df3fcba436e9e5bdf8bc2f1248
zip file	d9cfd1bed982779eb17ca45b4b31ea8e
rar file	e360559825976478cfc7abe24b2699b7
rar file	f75f6e731048c01bae2d75b53a057ad5
rar file	567f8cab5d1dd8f42934209b91fe15d3
SparrowDoor Loader	8a900f742d0e3cd3898f37dbc3d6e054
SparrowDoor Loader	a213873eb55dc092ddf3adbeb242bd44
BLOB	90afb6d2dfd161ce7752226b8a52e609

References

https://documents.trendmicro.com/assets/wp/wp-operation-tropic-trooper.pdf
https://citizenlab.ca/2016/11/parliament-keyboy/
https://www.macnica.co.jp/business/security/security-reports/pdf/cyberespionage_report_2022.pdf
https://www.ncsc.gov.uk/files/NCSC-MAR-SparrowDoor.pdf
https://www.welivesecurity.com/2021/09/23/famoussparrow-suspicious-hotel-guest/
https://www.trendmicro.com/en_fi/research/23/h/earth-estries-targets-government-tech-for-cyberespionage.html
https://www.eset.com/int/about/newsroom/press-releases/research/eset-research-discovers-famousparrow-apt-group-spying-on-hotels-governments-and-private-companies/
https://blog-en.itochuci.co.jp/entry/2023/09/28/171001