JUNE 2023

LOCKBIT 2.0 RANSOMWARE

Malware Analysis Report

Theodoros Vergos “cde”

Table of Contents

Executive Report

Malware Family	LOCKBIT 2.0 Ransomware
MD5	96de05212b30ec85d4cf03386c1b84af
SHA1	dbe5243c6ea5cc4cfb3edf042bd94a59cf9a0e64
SHA256	00260c390ffab5734208a7199df0e4229a76261c3f5b7264c4515acb8eb9c2f8
Architecture	x86

The malware sample was identified to be of the LOCKBIT 2.0 Ransomware strain. This is an extremely destructive malware that rapidly encrypts user files, bar executables and system files. The typical usage of this malware is as a final payload after the threat actors have exfiltrated user files via usage of applications, specifically legitimate cloud storage services. Then they attempt to contact the victim in order to extort them and demand ransom for not publishing and deleting any extracted files. If their demands are not fulfilled, they leak the files on their deep web blog.
The malware is confirmed to have the capability to delete shadow copies, making the recovery of files that were not backed up impossible.
During the analysis no workaround was detected in order to stall or block the malware execution.
At the moment that this report is being written, the malware is being detected by 62/70 vendors according to Virustotal.com.

The author has written a YARA rule to aid in the detection of the malware. This YARA rule can be found in the next section of this document.

Yara Rule

```
rule cde_lockbit2_detection_rule {

    meta:
        last_updated = "2023-06-26"
        author = "cde"
       description = "This is a Yara rule for detecting LockBit 2.0 ransomware."

    strings:
        // Fill out identifying strings and other criteria
        $PE_byte = "MZ"
        $string1 = "\\Registry\\Machine\\Software\\Classes\\.lockbit" wide
        $string2 = "LockBit_Ransomware.hta" wide
        $string3 = "C:\\Windows\\system32\\mshta.exe" wide
        $string4 = "3E5FC7F9-9A51-4367-9063-A120244FBEC7" wide
                                                        $string5             =
"3085B89A0C515D2FB124D645906F5D3DA5CB97CEBEA975959AE4F95302A04E1D709C3C4AE9B7"
wide
        $hex_string1 = {0C 08 04 04 52 95 C7 C7 65 46 23 23 5E 9D C3 C3}
        $hex_string2 = {09 89 89 80 1A 0D 0D 17}
        $hex_string3 = {03 3B C9 01 2A 62 2E 00 DB 25 23 FF A9 C3 26 00 5E 7C
C1 01 9C 2B DF 00 5F 48 85 FE 78 CE BF 00}
        $hex_string4 = {00 E7 5B ED FF BF AA CC FF 98 07 DE 00 6D C0 31 00 C1
A6 92 FF E8 13 B5 FF 69 8E 34 FF 67 10 1B 01 FD C8 A5 00}
        $hex_string5 = {B2 CD 7F B2 75 9F EA 75 09 1B 12 09 83 9E 1D 83 2C 74
58 2C 1A 2E 34 1A 1B 2D 36 1B 6E B2 DC 6E 5A EE B4 5A A0 FB 5B A0 52 F6 A4 52
3B 4D 76 3B D6 61 B7 D6 B3 CE 7D B3}

    condition:
        // Fill out the conditions that must be met to identify the binary
        $PE_byte at 0 and
          ($string1 and $string2 and $string3 and $string4 and $string5 and
$hex_string1 and $hex_string2 and $hex_string3 and $hex_string4 and
$hex_string5)
}
```

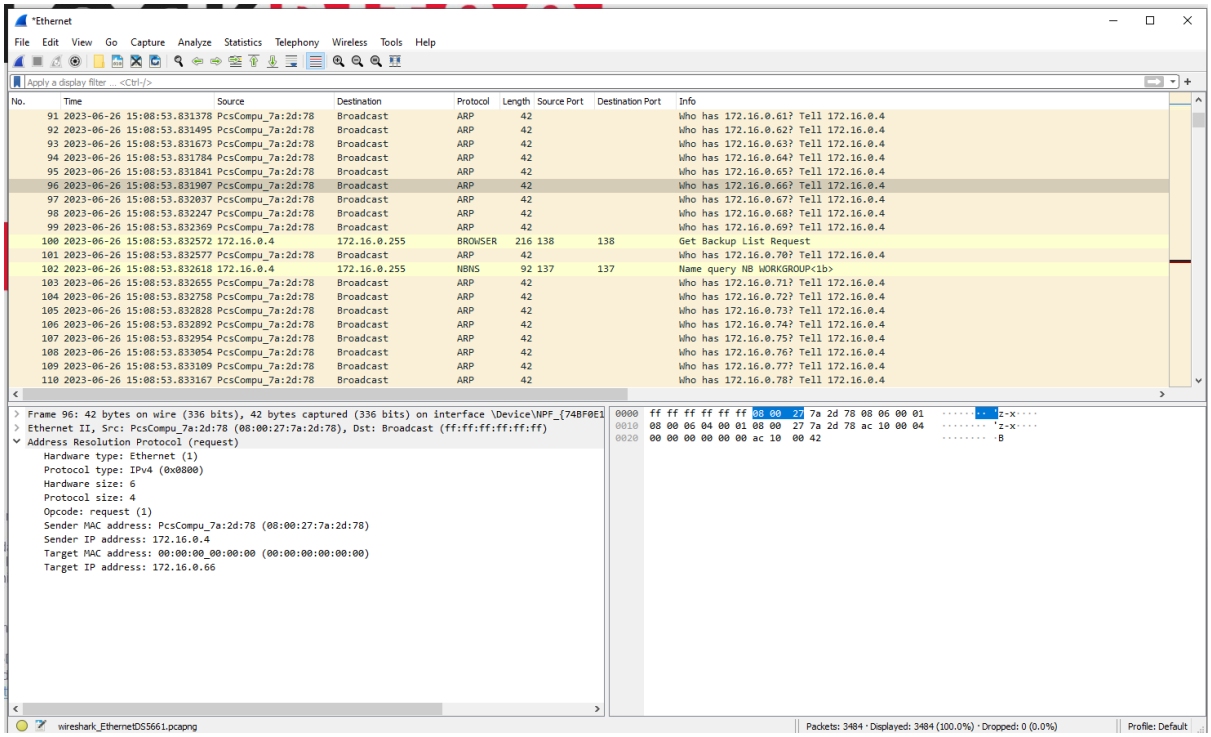
High Level Technical Summary

The malware was tested and analyzed under the following scenarios:

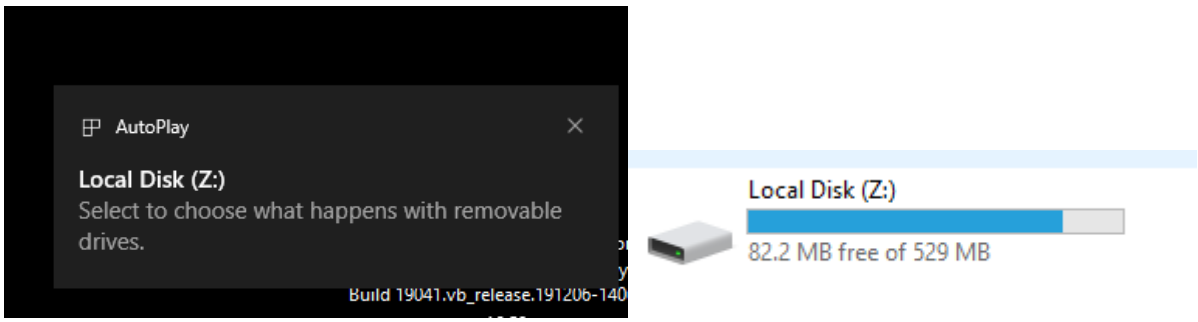
1. Execution with user privileges and no Internet access
2. Execution with user privileges and simulated Internet access with inetsim
3. Execution with administrative privileges and no Internet access
4. Execution with administrative privileges and simulated Internet access with inetsim

The malware was successfully executed under given scenarios with no notable differences in the outcome. For this reason, the reader may treat all scenarios as one.

Following the initial malware detonation, the system performs multiple ARP scans for network discovery. The reasons for those actions could not be determined due to the limited and restrictive nature of the lab environment.



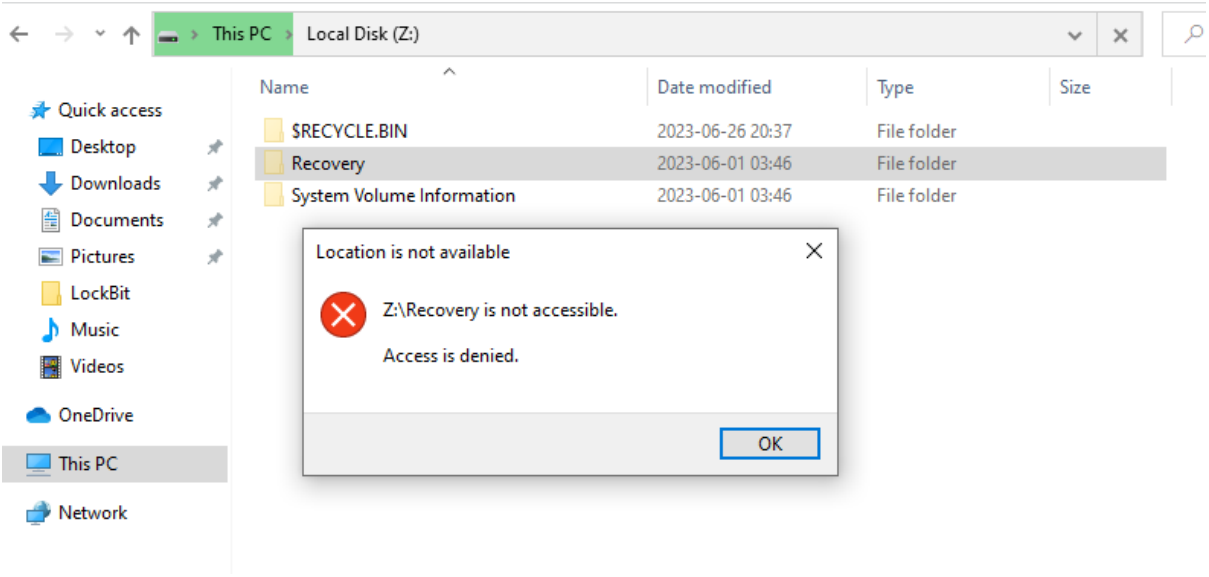
A few moments later an external media device is detected by the system and encryption of the user files begins.



The malware does not create any new users, but still is able to complete the execution sequence without the need for administrative rights.

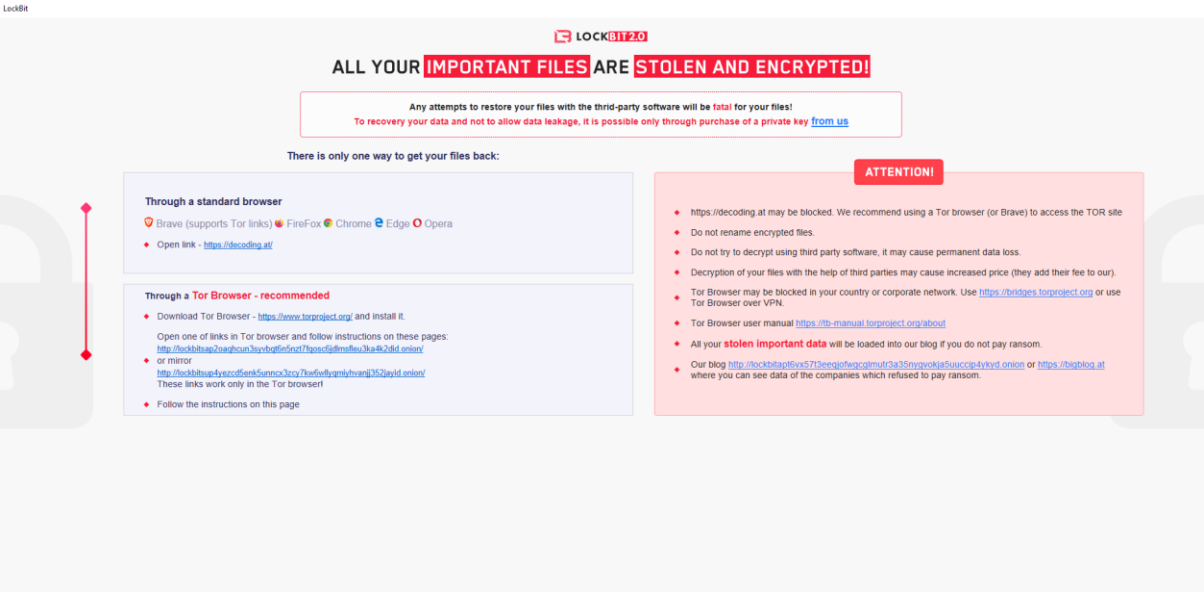
```
PS C:\Users\cde> gwmi win32_UserAccount | Select Name, FullName, Caption, Domain, SID | ft -AutoSize

Name               FullName Caption                            Domain          SID
----               -------- -------                            ------          ---
Administrator                DESKTOP-DEH1E4T\Administrator      DESKTOP-DEH1E4T S-1-5-21-2860406012-28484
cde                          DESKTOP-DEH1E4T\cde                DESKTOP-DEH1E4T S-1-5-21-2860406012-28484
DefaultAccount               DESKTOP-DEH1E4T\DefaultAccount     DESKTOP-DEH1E4T S-1-5-21-2860406012-28484
Guest                        DESKTOP-DEH1E4T\Guest              DESKTOP-DEH1E4T S-1-5-21-2860406012-28484
WDAGUtilityAccount           DESKTOP-DEH1E4T\WDAGUtilityAccount DESKTOP-DEH1E4T S-1-5-21-2860406012-28484
```

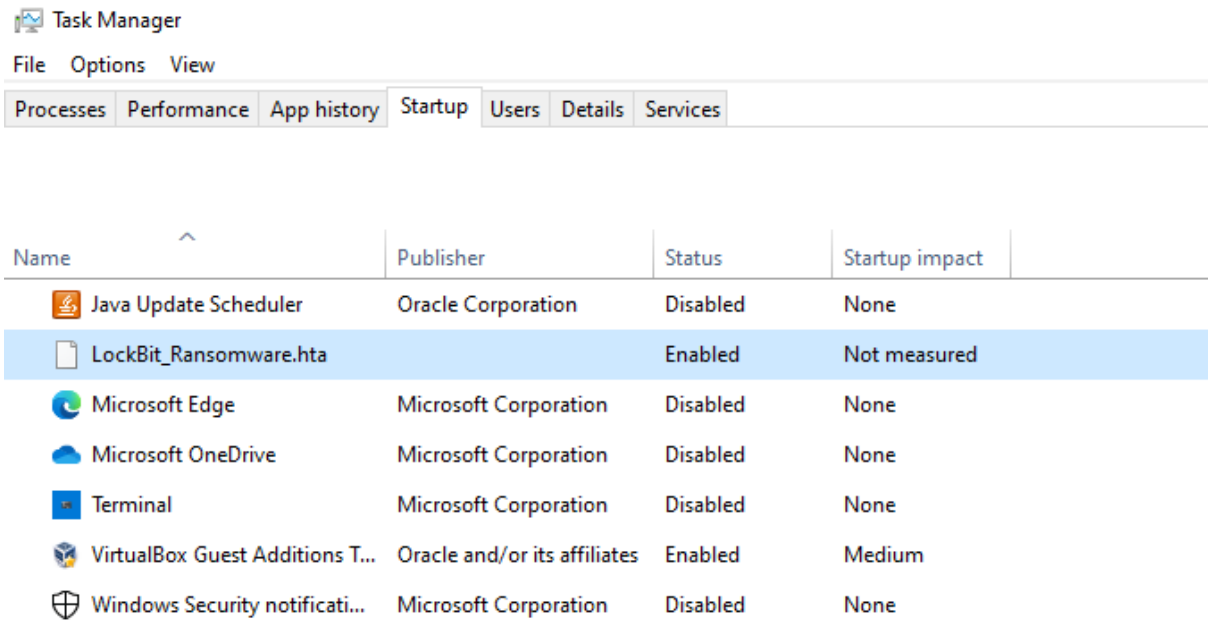
Following the malware execution and encryption some of the system files are not accessible to the user as seen below.



Then, the user wallpaper changes, all the encrypted files have the ".lockbit" suffix appended to them and on every folder that a file was encrypted a txt with the same instructions is generated.

Finally, a new file appears on the desktop (LockBit_Ransomware.hta), is executed by mshta.exe and the following persistent screen appears. This screen greets the user every time the system is restarted.





In the following pages of this report the reader will find a more in-depth analysis and detailed steps of the process followed by the researcher. Those findings can be used for further investigation and fine tuning of detection rules.

Static Analysis

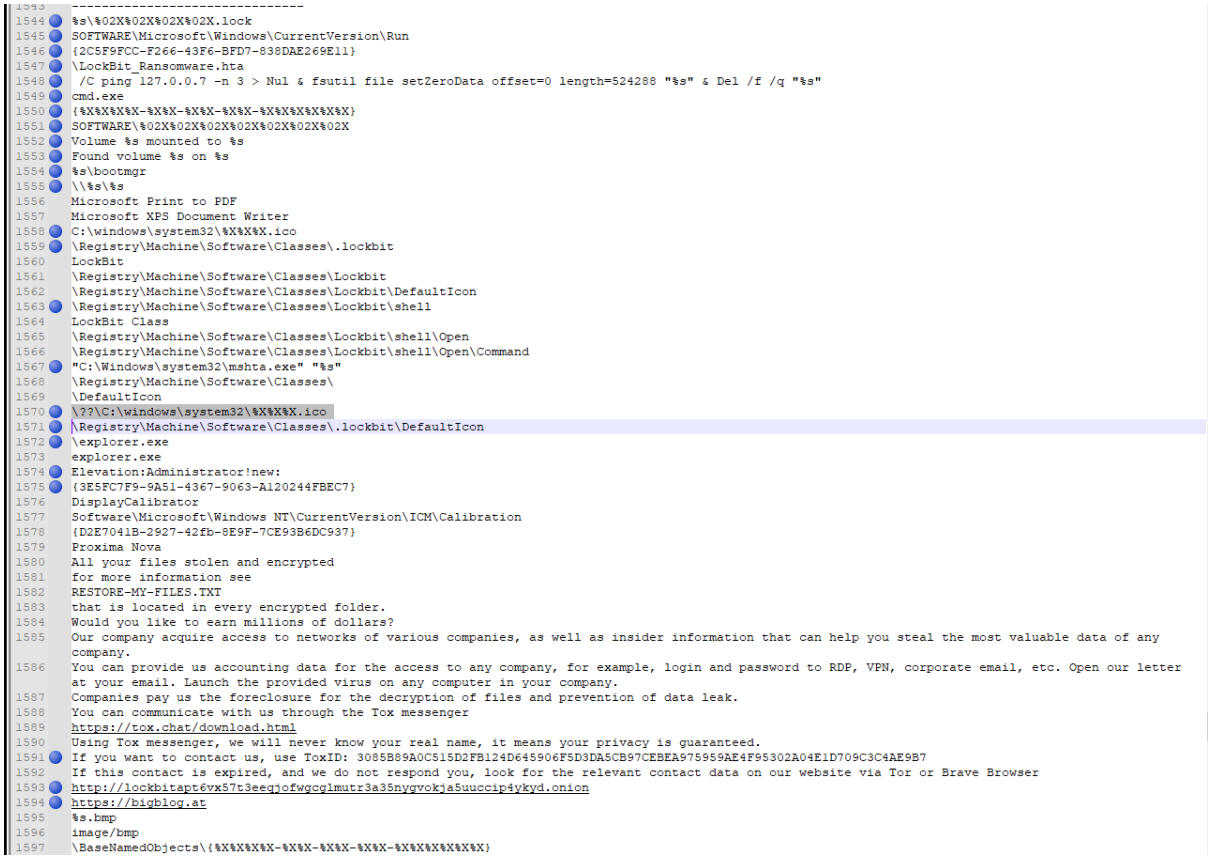
Basic Static Analysis

Using the command “floss -n 6 > output.txt” we get the following strings:

Wide ASCII strings (UTF-16):

1. %s\%02X%02X%02X%02X.lock
2. SOFTWARE\Microsoft\Windows\CurrentVersion\Run
3. {2C5F9FCC-F266-43F6-BFD7-838DAE269E11}
4. \LockBit_Ransomware.hta
5. /C ping 127.0.0.7 -n 3 > Nul & fsutil file setZeroData offset=0 length=524288 "%s" & Del /f /q "%s"
6. cmd.exe
7. {%X%X%X%X-%X%X-%X%X-%X%X-%X%X%X%X%X%X}
8. SOFTWARE\%02X%02X%02X%02X%02X%02X%02X
9. Volume %s mounted to %s
10. Found volume %s on %s
11. %s\bootmgr
12. \\%s\%s
13. Microsoft Print to PDF
14. Microsoft XPS Document Writer
15. C:\windows\system32\%X%X%X.ico
16. \Registry\Machine\Software\Classes\.lockbit
17. LockBit
18. \Registry\Machine\Software\Classes\Lockbit
19. \Registry\Machine\Software\Classes\Lockbit\DefaultIcon
20. \Registry\Machine\Software\Classes\Lockbit\shell
21. LockBit Class
22. \Registry\Machine\Software\Classes\Lockbit\shell\Open
23. \Registry\Machine\Software\Classes\Lockbit\shell\Open\Command
24. "C:\Windows\system32\mshta.exe" "%s"
25. \Registry\Machine\Software\Classes\
26. \DefaultIcon
27. \??\C:\windows\system32\%X%X%X.ico
28. \Registry\Machine\Software\Classes\.lockbit\DefaultIcon
29. \explorer.exe
30. explorer.exe
31. Elevation:Administrator!new:
32. {3E5FC7F9-9A51-4367-9063-A120244FBEC7}
33. DisplayCalibrator
34. Software\Microsoft\Windows NT\CurrentVersion\ICM\Calibration
35. {D2E7041B-2927-42fb-8E9F-7CE93B6DC937}
36. Proxima Nova
37. All your files stolen and encrypted
38. for more information see
39. RESTORE-MY-FILES.TXT
40. that is located in every encrypted folder.
41. Would you like to earn millions of dollars?

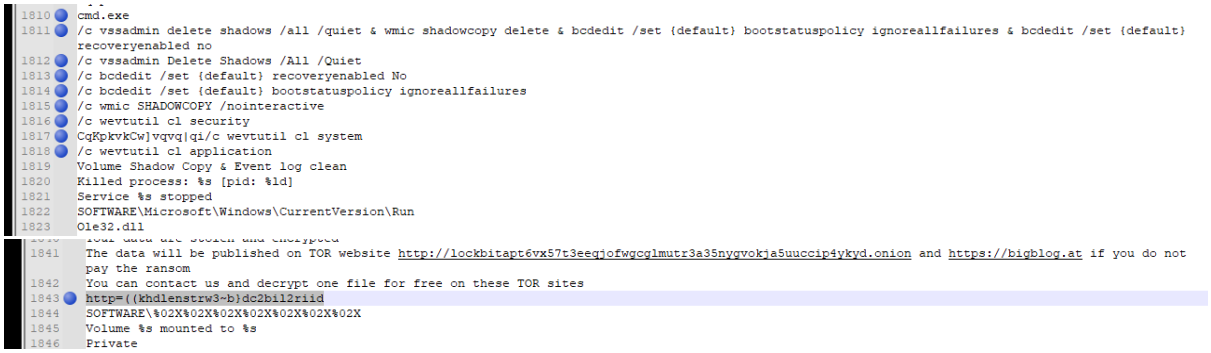
42. Our company acquire access to networks of various companies, as well as insider information that can help you steal the most valuable data of any company.
43. You can provide us accounting data for the access to any company, for example, login and password to RDP, VPN, corporate email, etc. Open our letter at your email. Launch the provided virus on any computer in your company.
44. Companies pay us the foreclosure for the decryption of files and prevention of data leak.
45. You can communicate with us through the Tox messenger
46. hxxps[:]//tox[.]chat/download[.]html
47. Using Tox messenger, we will never know your real name, it means your privacy is guaranteed.
48. If you want to contact us, use ToxID: 3085B89A0C515D2FB124D645906F5D3DA5CB97CEBEA975959AE4F95302A04E1D709C3C4AE9B7
49. If this contact is expired, and we do not respond you, look for the relevant contact data on our website via Tor or Brave Browser
50. hxxp[:]//lockbitapt6vx57t3eeqjofwgcglmutr3a35nygvokja5uuccip4ykyd[.]onion
51. hxxps[:]//bigblog[.]at



Tight/encoded strings:

1. cmd.exe
2. /c vssadmin delete shadows /all /quiet & wmic shadowcopy delete & bcdedit /set {default} bootstatuspolicy ignoreallfailures & bcdedit /set {default} recoveryenabled no
3. /c vssadmin Delete Shadows /All /Quiet
4. /c bcdedit /set {default} recoveryenabled No
5. /c bcdedit /set {default} bootstatuspolicy ignoreallfailures
6. /c wmic SHADOWCOPY /nointeractive

7. /c wevtutil cl security
8. CqKpkvkCw]vqvq|qi/c wevtutil cl system
9. /c wevtutil cl application
10. Volume Shadow Copy & Event log clean
11. Killed process: %s [pid: %ld]
12. http=((khdlenstrw3~b}dc2bil2riid



Advanced Static Analysis

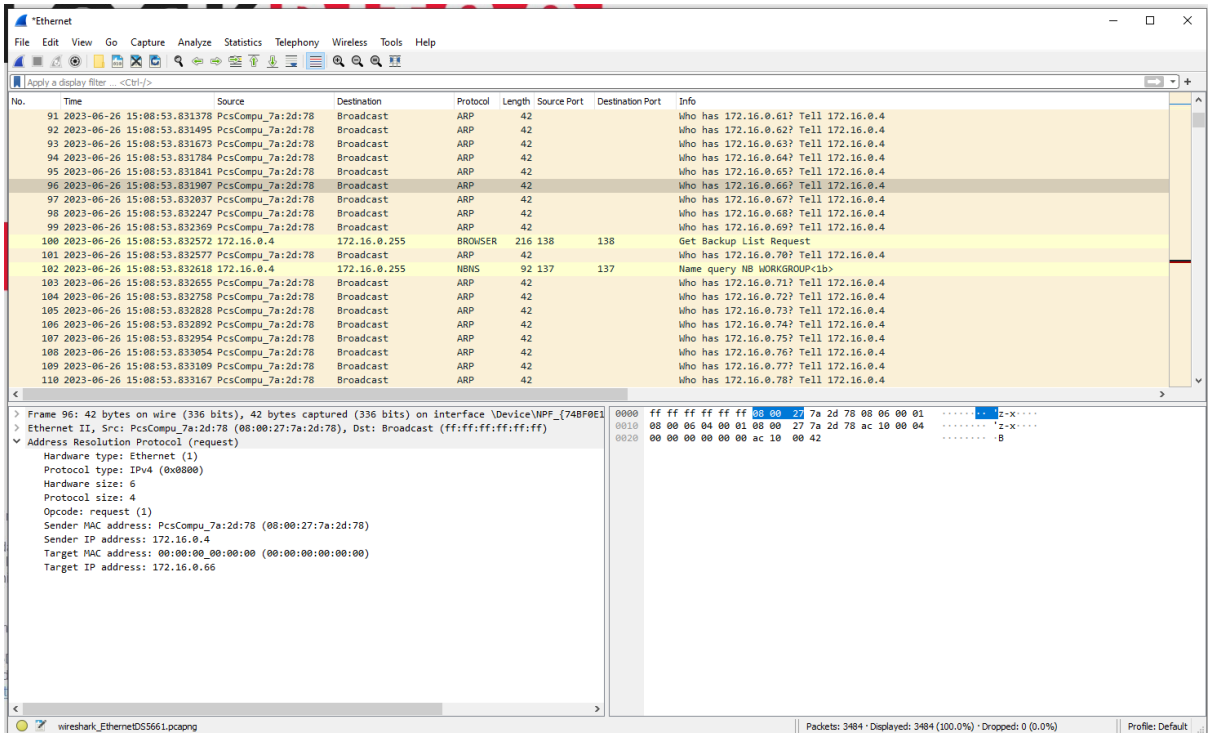
The malware is very complex and highly obfuscated leaving the researcher little opportunities for statically analyzing the malware code. The findings regarding to the malware code come from the dynamic analysis section of the investigation.

Dynamic Analysis

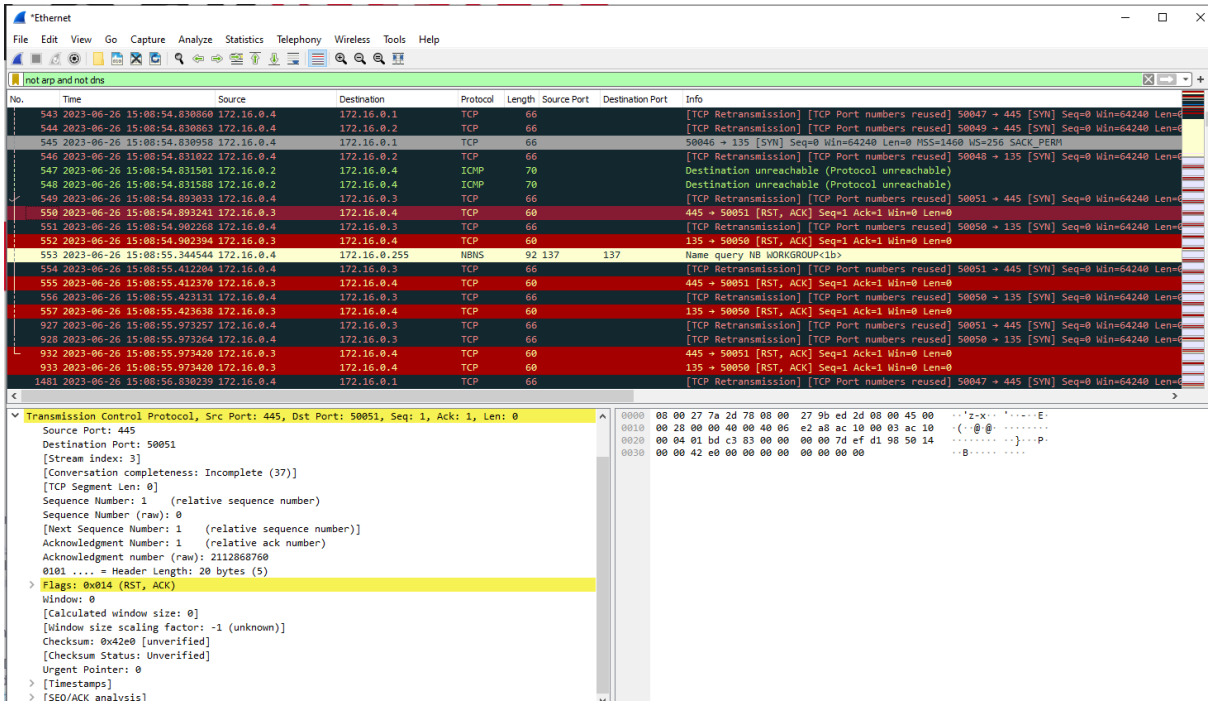
Basic Dynamic Analysis

In the following screenshots is shown the network traffic during and after the malware execution.

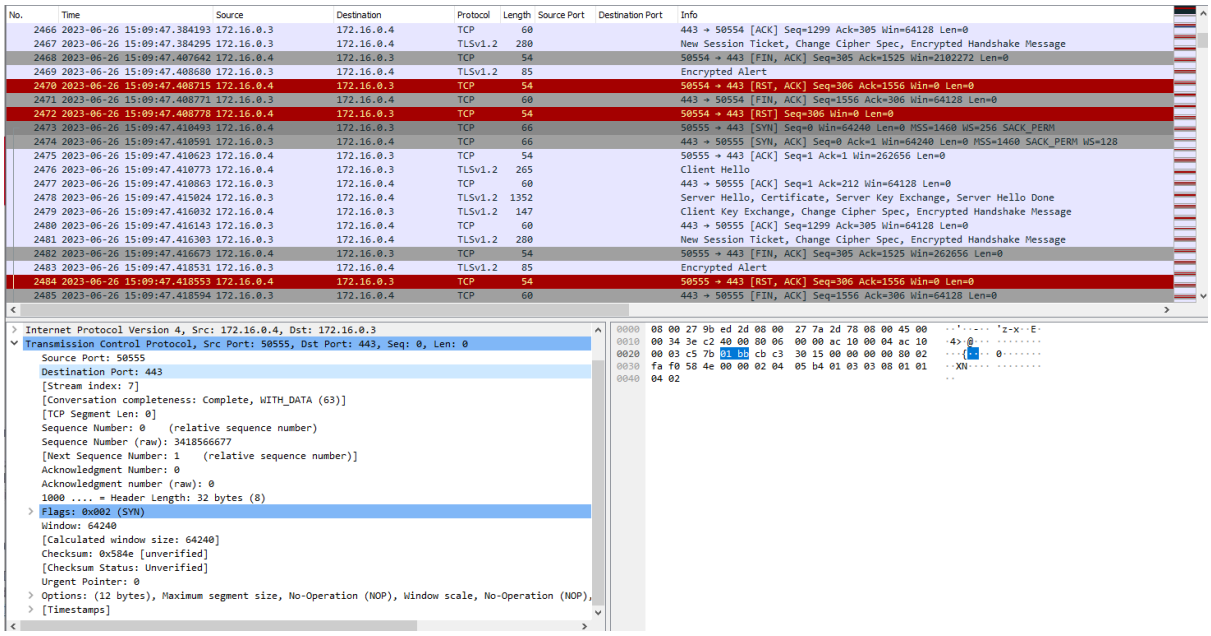
First the ARP scan as mentioned in the High-Level Technical analysis.



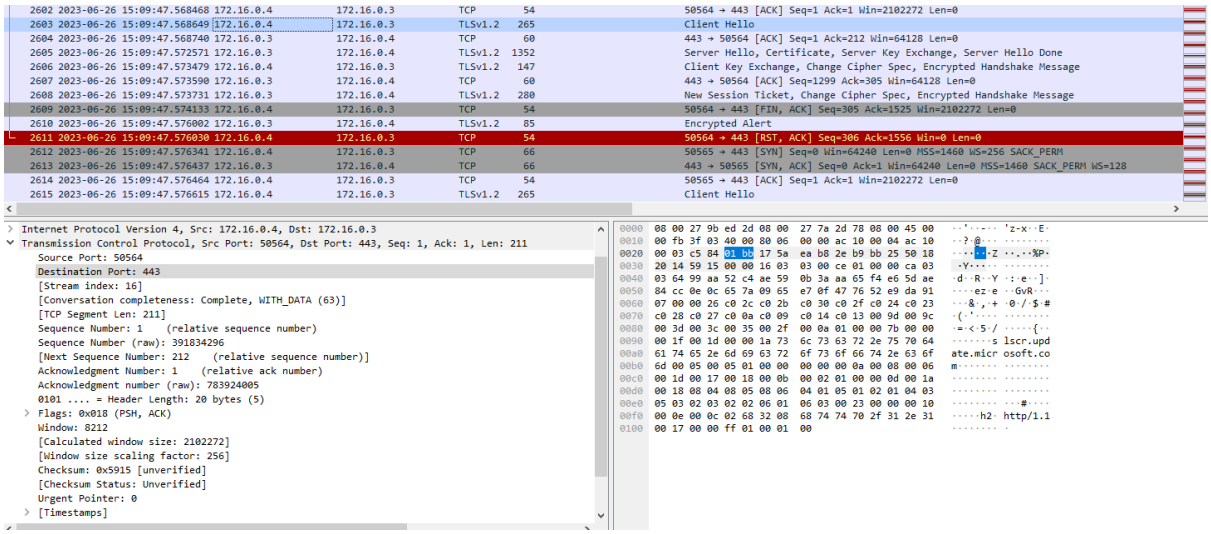
Then, the system attempts connect to port 445 (SMB) on the systems that were detected.



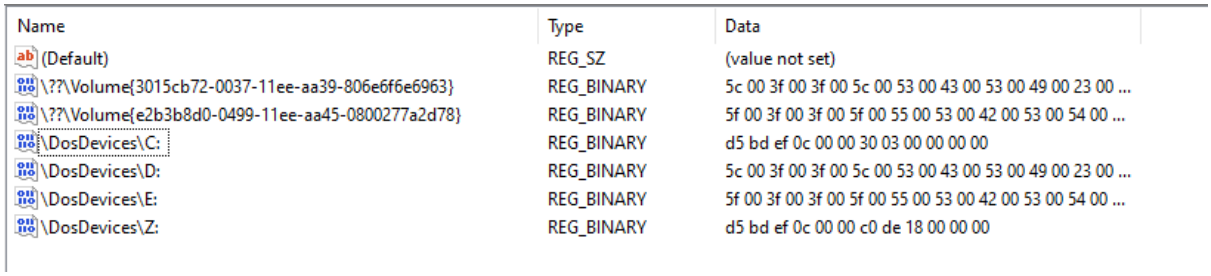
Since no connection could be established with those systems, the host attempts to connect to port 443 TCP and establish a TLSv1.2 tunnel.



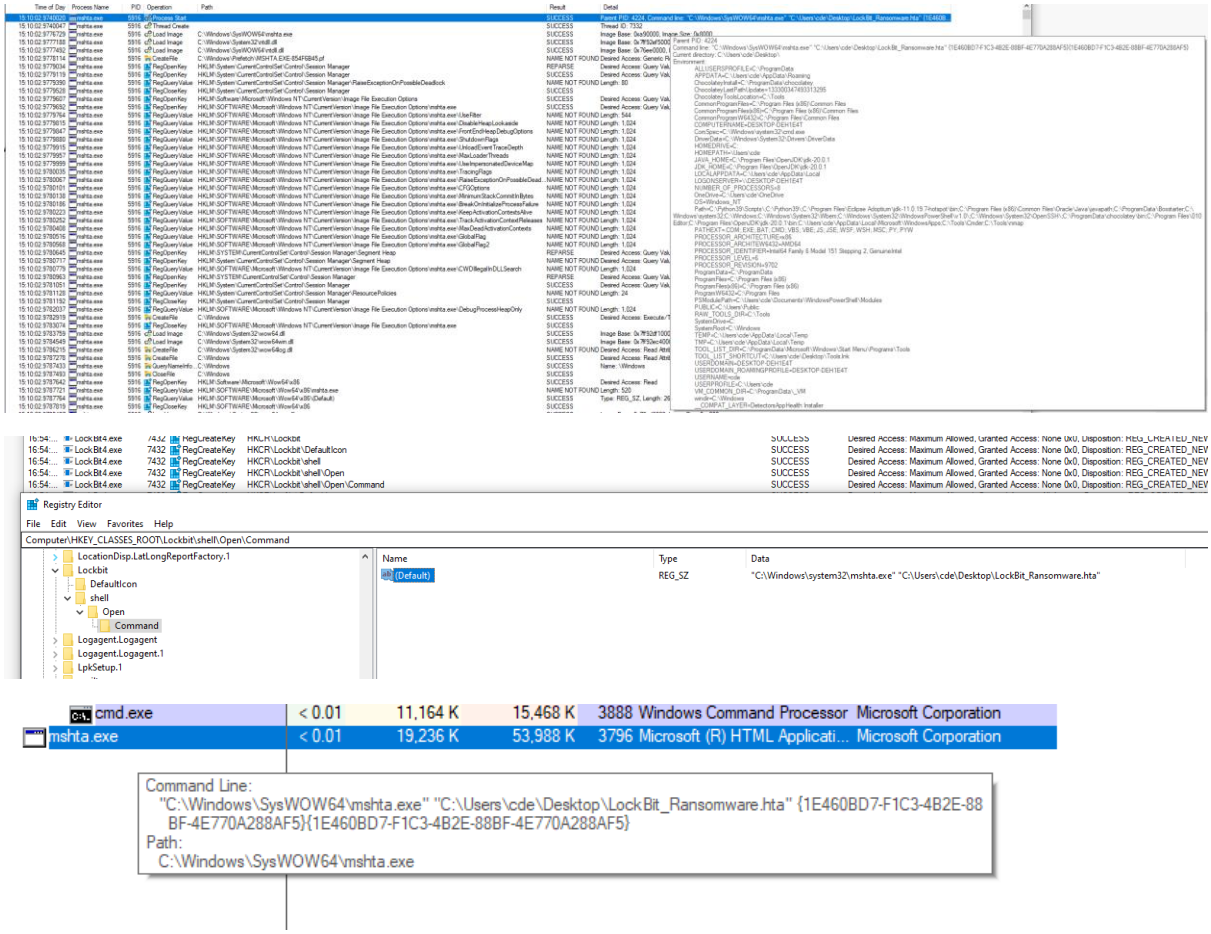
Since there was not an active listener to provide a callback the TLS tunnel is teared down after a while.



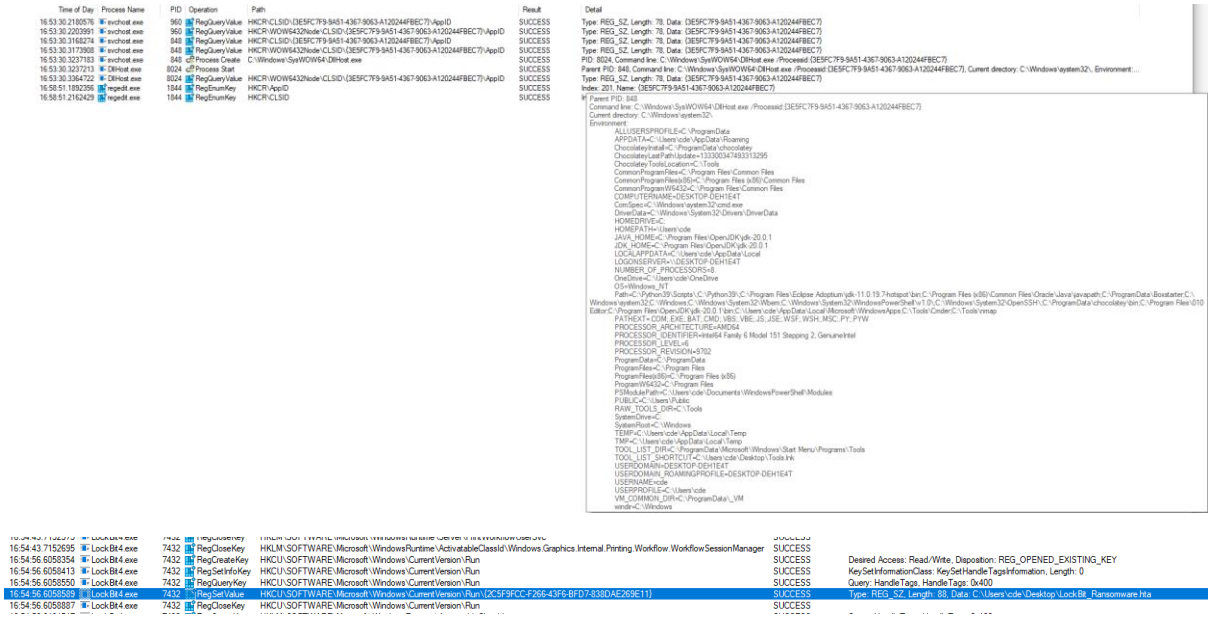
By documenting the storage devices' registry, we notice that the newly attached device Z:\ has the same Data value as our C:\ drive, but no other data could be found in order to support chasing this lead.



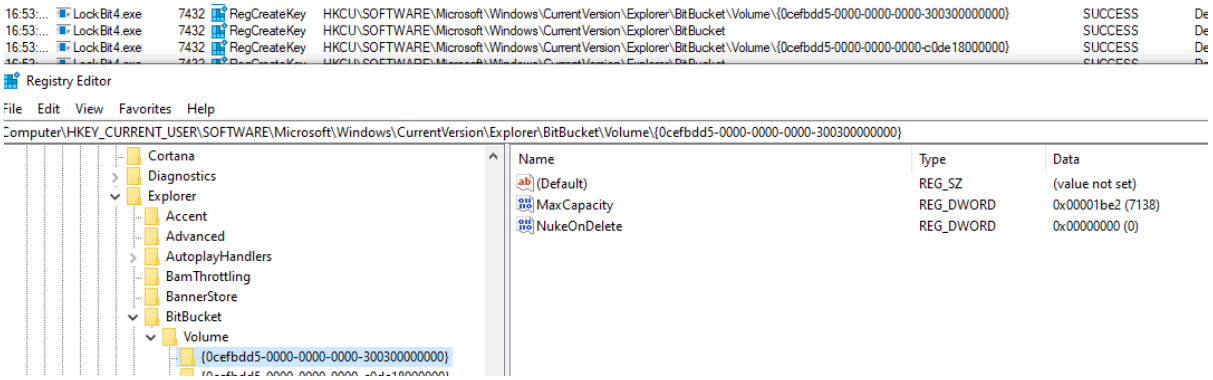
In the next screenshots the execution of the .hta file is documented.



During our static analysis, we notice that ASCII string number 32 looks like a processed. We search in procmon for "detail" containing this value and we get the following findings. It appears that the malware is performing a UAC bypass in order to execute with higher privileges. This is a vulnerability addressed in https://nvd.nist.gov/vuln/detail/cve-2016-0099 and https://learn.microsoft.com/en-us/security-updates/securitybulletins/2016/ms16-032.



Another interesting anti-forensics and anti-recovery technique used by the threat actors is changing the NukeOnDelete registry which enables file original locations to be deleted instead of going first to the Recycle Bin. This is documented in the following screenshot.



For the last part of the basic dynamic analysis, we check open connections with “netstat -a” before the malware is executed

```
C:\Users\cde\Desktop\test\LockBit
λ netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:445            DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:5040           DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:7680           DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49664          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49665          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49666          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49667          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49669          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49672          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49677          DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:135               DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:445               DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:7680              DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49664             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49665             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49666             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49667             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49669             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49672             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49677             DESKTOP-DEH1E4T:0      LISTENING
  UDP    0.0.0.0:123            *:*
  UDP    0.0.0.0:500            *:*
  UDP    0.0.0.0:4500           *:*
  UDP    0.0.0.0:5050           *:*
  UDP    127.0.0.1:1900         *:*
  UDP    127.0.0.1:63328        *:*
  UDP    127.0.0.1:64141        *:*
  UDP    [::]:123               *:*
  UDP    [::]:500               *:*
  UDP    [::]:4500              *:*
  UDP    [::1]:1900             *:*
  UDP    [::1]:64140            *:*

C:\Users\cde\Desktop\test\LockBit
λ
```

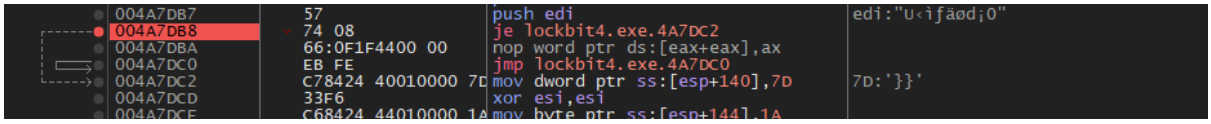
and compare it immediately after the execution has completed and note the changes.

```
C:\Users\cde\Desktop\test\LockBit
λ netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:445            DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:5040           DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:7680           DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49664          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49665          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49666          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49667          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49669          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49672          DESKTOP-DEH1E4T:0      LISTENING
  TCP    0.0.0.0:49677          DESKTOP-DEH1E4T:0      LISTENING
  TCP    172.16.0.4:139         DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:135               DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:445               DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:7680              DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49664             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49665             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49666             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49667             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49669             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49672             DESKTOP-DEH1E4T:0      LISTENING
  TCP    [::]:49677             DESKTOP-DEH1E4T:0      LISTENING
  UDP    0.0.0.0:123            *:*
  UDP    0.0.0.0:500            *:*
  UDP    0.0.0.0:4500           *:*
  UDP    0.0.0.0:5050           *:*
  UDP    0.0.0.0:5353           *:*
  UDP    0.0.0.0:5355           *:*
  UDP    127.0.0.1:1900         *:*
  UDP    127.0.0.1:57978        *:*
  UDP    127.0.0.1:63328        *:*
  UDP    172.16.0.4:137         *:*
  UDP    172.16.0.4:138         *:*
  UDP    172.16.0.4:1900        *:*
  UDP    172.16.0.4:57977       *:*
  UDP    [::]:123               *:*
  UDP    [::]:500               *:*
  UDP    [::]:4500              *:*
  UDP    [::]:5353              *:*
  UDP    [::]:5355              *:*
  UDP    [::1]:1900             *:*
  UDP    [::1]:57976            *:*
  UDP    [fe80::b4a0:8973:90e6:82de%4]:546  *:*
  UDP    [fe80::b4a0:8973:90e6:82de%4]:1900  *:*
  UDP    [fe80::b4a0:8973:90e6:82de%4]:57975  *:*

C:\Users\cde\Desktop\test\LockBit
λ
```

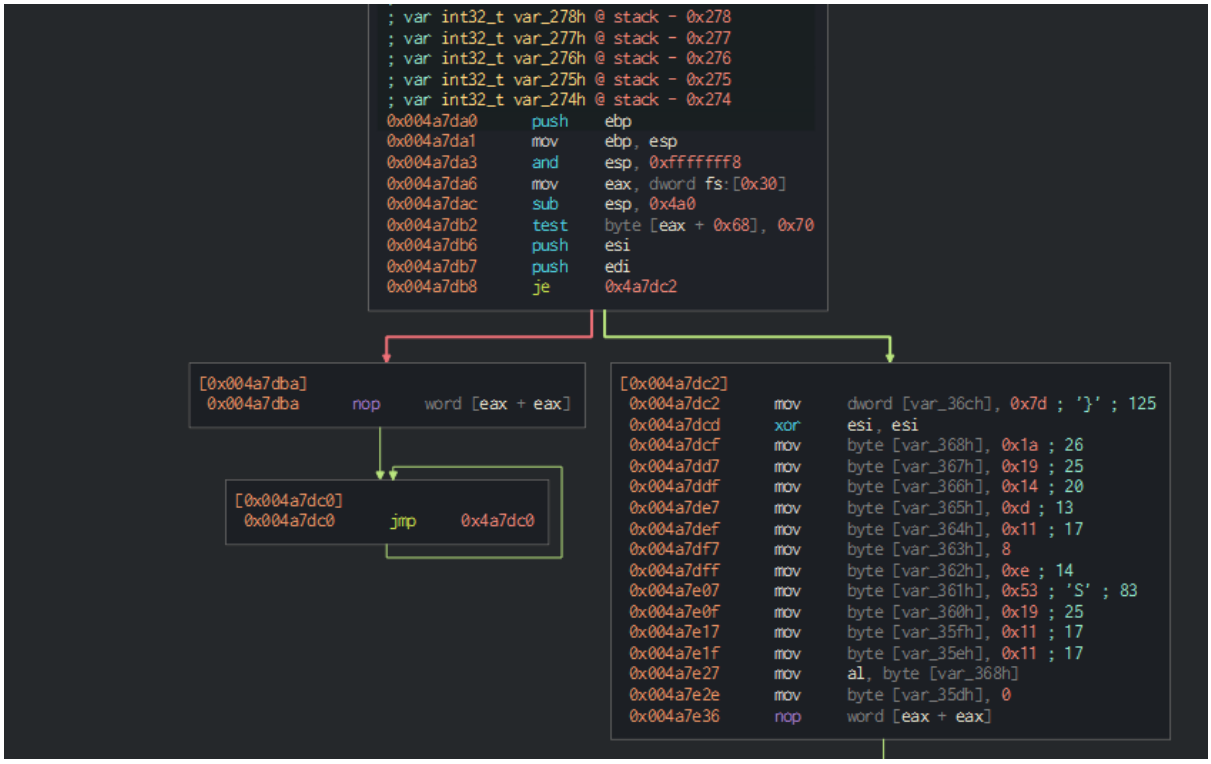
Advanced Dynamic Analysis:

By putting the malware through a debugger (x64dbg) the following could be unraveled about the malware code.

The malware check for the presence of a debugger and if it is true, it traps the execution in an endless loop. This can be escaped by modifying the ZF and proceeding with the execution.

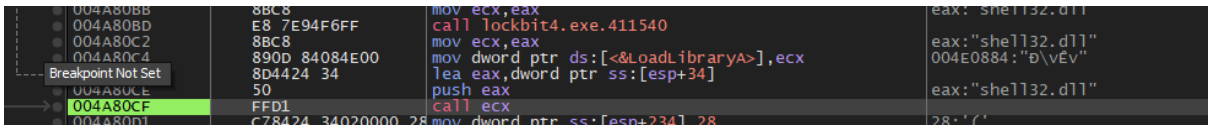


Which is verified on the main function - entry0

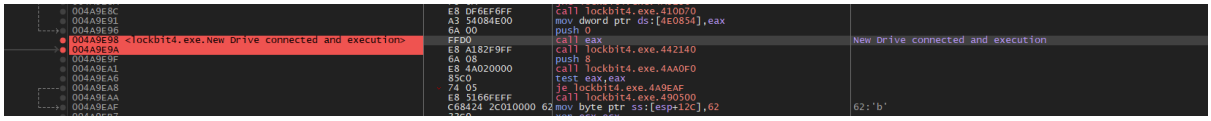


The malware pushes shell32.dll in the stack, which is used when opening web pages and files.

Thenby calling ecx the malware uses the LoadLibraryA API which allows it to call shell32.dll



When we reach location 004A9E98, the ransomware function is called attaching the drive Z:\ to the system and proceeding with encrypting the user's files.



The functions Fcn.00485fa0 and Fcn.00442140 is responsible for encrypting the files after enumerating them. For the encryption it is understood that a XOR cipher is used as seen in the snippet below.


0x00485fd5 mov edi, 2
0x00485fda setne byte [0x4e0920]
0x00485fe1 test eax, eax
0x00485fe3 je 0x4860e6
[0x00485fe9]
0x00485fe9 mov dword [var_40h], 0x56 ; 'V' ; 86
0x00485ff0 mov eax, dword [var_40h]
0x00485ff3 xor eax, 0x5b ; 91
0x00485ff6 mov byte [var_3ch], al
0x00485ff9 mov eax, dword [var_40h]
0x00485ffc inc al
0x00485ffe xor eax, 0x2b ; 43
0x00486001 mov byte [var_3bh], al
0x00486004 mov eax, dword [var_40h]
0x00486007 add al, 2
0x00486009 xor eax, 0x5d ; 93
0x0048600c mov byte [var_3ah], al
0x0048600f mov eax, dword [var_40h]
0x00486012 add al, 3
0x00486014 xor eax, 0x20 ; 32
0x00486017 mov byte [var_39h], al
0x0048601a mov eax, dword [var_40h]
0x0048601d add al, 4
0x0048601f xor eax, 0x41 ; 65
0x00486022 mov byte [var_38h], al
0x00486025 mov eax, dword [var_40h]
0x00486028 add al, 5
0x0048602a xor eax, 0x45 ; 69
0x0048602d mov byte [var_37h], al
0x00486030 mov eax, dword [var_40h]
0x00486033 add al, 6
0x00486035 xor eax, 0x53 ; 83
0x00486038 mov byte [var_36h], al
0x0048603b mov eax, dword [var_40h]
0x0048603e add al, 7
0x00486040 xor eax, 0x2d ; 45
0x00486043 mov byte [var_35h], al
0x00486046 mov eax, dword [var_40h]
0x00486049 add al, 8
0x0048604b xor eax, 0x4e ; 78
0x0048604e mov byte [var_34h], al
0x00486051 mov eax, dword [var_40h]
0x00486054 add al, 9
0x00486056 xor eax, 0x49 ; 73
0x00486059 mov byte [var_33h], al
0x0048605c mov eax, dword [var_40h]
0x0048605f add al, 0xa
0x00486061 xor eax, 0x20 ; 32
0x00486064 mov byte [var_32h], al
0x00486067 mov eax, dword [var_40h]
0x0048606a add al, 0xb ; 11
0x0048606c xor eax, 0x65 ; 101
0x0048606f mov byte [var_31h], al
0x00486072 mov eax, dword [var_40h]
0x00486075 add al, 0xc ; 12
0x00486077 xor eax, 0x6e ; 110
0x0048607a mov byte [var_30h], al
0x0048607d mov eax, dword [var_40h]
0x00486080 add al, 0xd ; 13
0x00486082 xor eax, 0x61 ; 97
0x00486085 mov byte [var_2fh], al
0x00486088 mov eax, dword [var_40h]
0x0048608b add al, 0xe ; 14
0x0048608d xor eax, 0x62 ; 98
0x00486090 mov byte [var_2eh], al
0x00486093 mov eax, dword [var_40h]
0x00486096 add al, 0xf ; 15
0x00486098 xor eax, 0x6c ; 108
0x0048609b mov byte [var_2dh], al
0x0048609e mov eax, dword [var_40h]
0x004860a1 add al, 0x10 ; 16
0x004860a3 xor eax, 0x65 ; 101
0x004860a6 mov byte [var_2ch], al
0x004860a9 mov eax, dword [var_40h]
0x004860ac add al, 0x11 ; 17
0x004860ae mov byte [var_2ah], 0
0x004860b2 xor eax, 0x64 ; 100
0x004860b5 xor edx, edx
0x004860b7 mov byte [var_2bh], al
0x004860ba mov al, byte [var_3ch]
0x004860bd nop dword [eax]
[0x004860e6]
0x004860e6 mov dword [var_1ch], 0x2d ; '-' ; 45
0x004860ed mov eax, dword [var_1ch]
0x004860f0 xor eax, 0x5b ; 91
0x004860f3 mov byte [var_18h], al
0x004860f6 mov eax, dword [var_1ch]
0x004860f9 inc al
0x004860fb xor eax, 0x2d ; 45
0x004860fe mov byte [var_17h], al
0x00486101 mov eax, dword [var_1ch]
0x00486104 add al, 2
0x00486106 xor eax, 0x5d ; 93
0x00486109 mov byte [var_16h], al
0x0048610c mov eax, dword [var_1ch]
0x0048610f add al, 3
0x00486111 xor eax, 0x20 ; 32
0x00486114 mov byte [var_15h], al
0x00486117 mov eax, dword [var_1ch]
0x0048611a add al, 4
0x0048611c xor eax, 0x41 ; 65
0x0048611f mov byte [var_14h], al
0x00486122 mov eax, dword [var_1ch]
0x00486125 add al, 5
0x00486127 xor eax, 0x45 ; 69
0x0048612a mov byte [var_13h], al
0x0048612d mov eax, dword [var_1ch]
0x00486130 add al, 6
0x00486132 xor eax, 0x53 ; 83
0x00486135 mov byte [var_12h], al
0x00486138 mov eax, dword [var_1ch]
0x0048613b add al, 7
0x0048613d xor eax, 0x2d ; 45
0x00486140 mov byte [var_11h], al
0x00486143 mov eax, dword [var_1ch]
0x00486146 add al, 8
0x00486148 xor eax, 0x4e ; 78
0x0048614b mov byte [var_10h], al
0x0048614e mov eax, dword [var_1ch]
0x00486151 add al, 9
0x00486153 xor eax, 0x49 ; 73
0x00486156 mov byte [var_fh], al
0x00486159 mov eax, dword [var_1ch]
0x0048615c add al, 0xa
0x0048615e xor eax, 0x20 ; 32
0x00486161 mov byte [var_eh], al
0x00486164 mov eax, dword [var_1ch]
0x00486167 add al, 0xb ; 11
0x00486169 xor eax, 0x64 ; 100
0x0048616c mov byte [var_dh], al
0x0048616f mov eax, dword [var_1ch]
0x00486172 add al, 0xc ; 12
0x00486174 xor eax, 0x69 ; 105
0x00486177 mov byte [var_ch], al
0x0048617a mov eax, dword [var_1ch]
0x0048617d add al, 0xd ; 13
0x0048617f xor eax, 0x73 ; 115
0x00486182 mov byte [var_bh], al
0x00486185 mov eax, dword [var_1ch]
0x00486188 add al, 0xe ; 14
0x0048618a xor eax, 0x61 ; 97
0x0048618d mov byte [var_ah], al
0x00486190 mov eax, dword [var_1ch]
0x00486193 add al, 0xf ; 15
0x00486195 xor eax, 0x62 ; 98
0x00486198 mov byte [var_9h], al
0x0048619b mov eax, dword [var_1ch]
0x0048619e add al, 0x10 ; 16
0x004861a0 xor eax, 0x6c ; 108
0x004861a3 mov byte [var_8h], al
0x004861a6 mov eax, dword [var_1ch]
0x004861a9 add al, 0x11 ; 17
0x004861ab xor eax, 0x65 ; 101
0x004861ae mov byte [var_7h], al
0x004861b1 mov eax, dword [var_1ch]
0x004861b4 add al, 0x12 ; 18
0x004861b6 mov byte [var_5h], 0
0x004861ba xor eax, 0x64 ; 100
0x004861bd xor edx, edx
0x004861bf mov byte [var_6h], al
0x004861c2 mov al, byte [var_18h]