

# A Deep Dive into Medusa Ransomware

**Prepared by:** Vlad Pasca, Senior Malware &  
Threat Analyst



[SecurityScorecard.com](https://www.SecurityScorecard.com)  
[info@securityscorecard.com](mailto:info@securityscorecard.com)

Tower 49  
12 E 49<sup>th</sup> Street  
Suite 15-001  
New York, NY 10017  
[1.800.682.1707](tel:18006821707)

## Table of contents

|                          |    |
|--------------------------|----|
| Table of contents        | 1  |
| Executive summary        | 2  |
| Analysis and findings    | 2  |
| Indicators of Compromise | 18 |
| Appendix                 | 19 |

## Executive summary

Medusa ransomware appeared in June 2021, and it became more active this year by launching the “Medusa Blog” containing data leaked from victims that didn’t pay the ransom. The malware stops a list of services and processes decrypted at runtime and deletes the Volume Shadow Copies.

The files are encrypted using the AES256 algorithm, with the key being encrypted using an RSA public key. The ransomware deletes itself after the file encryption is complete. The extension of the encrypted files is changed to “.MEDUSA”.

## Analysis and findings

We will analyze a ransomware sample that our Professional Services team found in a Medusa Ransomware engagement. We can’t share the malware hash to protect the client's confidentiality.

The ransomware can run with one of the following parameters: “-d”, “-f”, “-i”, “-k”, “-n”, “-p”, “-s”, “-t”, “-v”, “-w”, and “-V”. If the “-v” parameter is not specified, the process calls the ShowWindow function to hide the current window (0x0 = **SW\_HIDE**):



Figure 1

When running with the “-v” parameter, the malware displays multiple strings in the command line window using WriteFile, as shown in Figure 2.



Figure 2

The malicious process creates multiple anonymous pipes via a function call to CreatePipe:

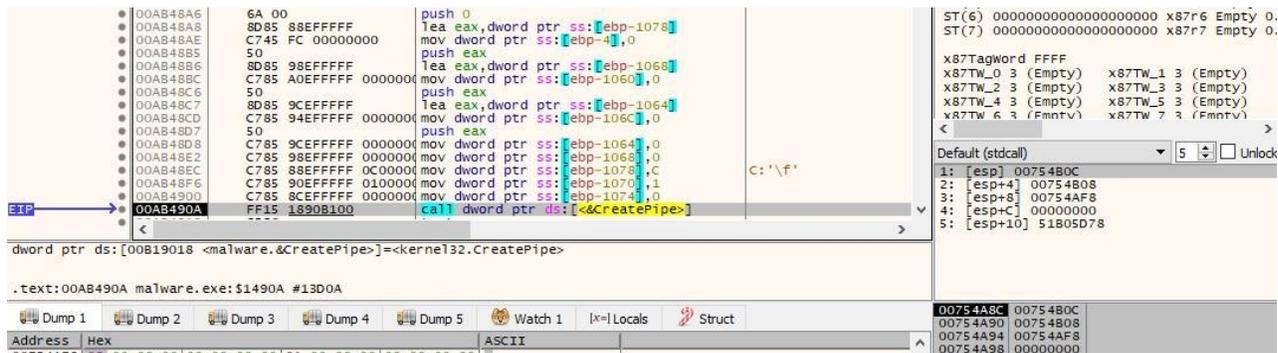


Figure 3

The SetHandleInformation routine is used to make the pipes inheritable by child processes (0x1 = **HANDLE\_FLAG\_INHERIT**):

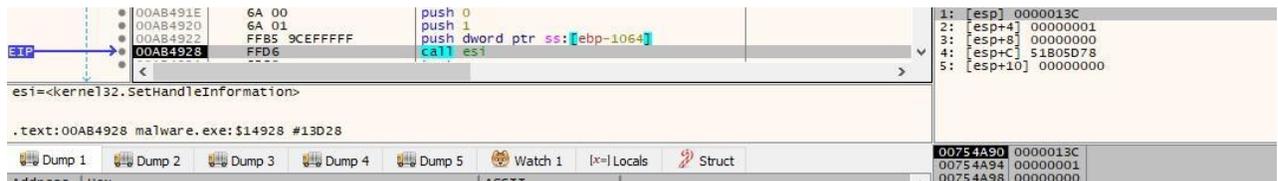


Figure 4

The malware creates a PowerShell process using the CreateProcessA API (0x08000000 = **CREATE\_NO\_WINDOW**):

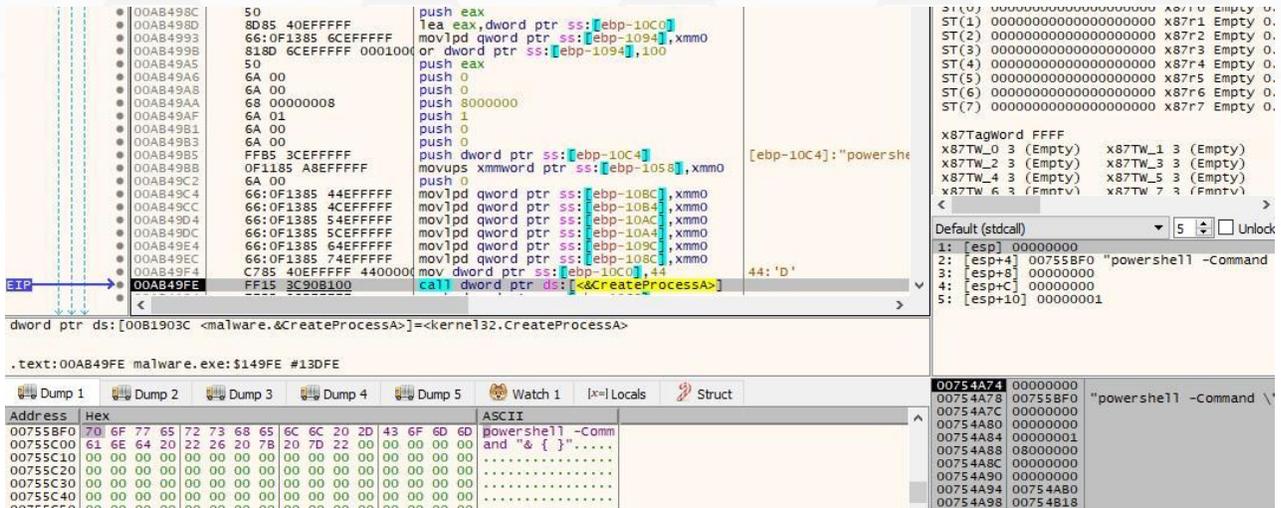


Figure 5

The malicious process reads data from the pipe containing the above process output using ReadFile:

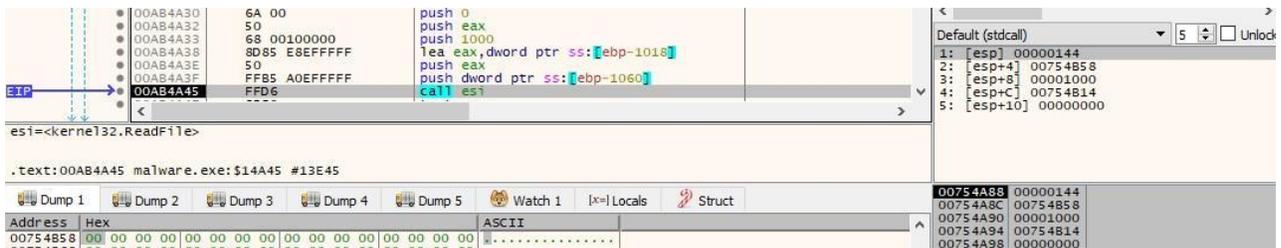


Figure 6

The sample retrieves the firmware table from the raw SMBIOS firmware table provider using the GetSystemFirmwareTable routine (see Figure 7).

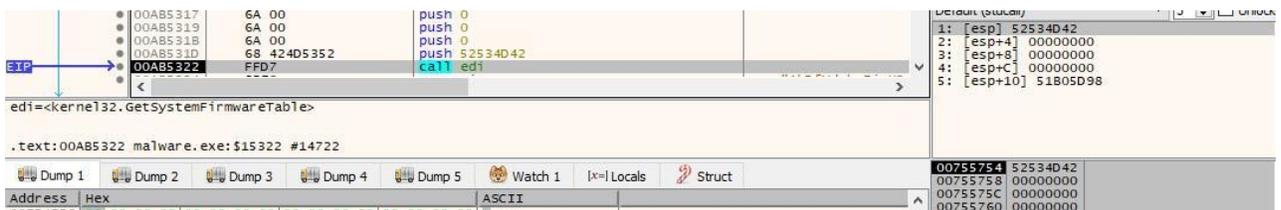


Figure 7

CryptStringToBinaryA is used to decode the RSA public key from Base64 (0x7 = **CRYPT\_STRING\_ANY**):

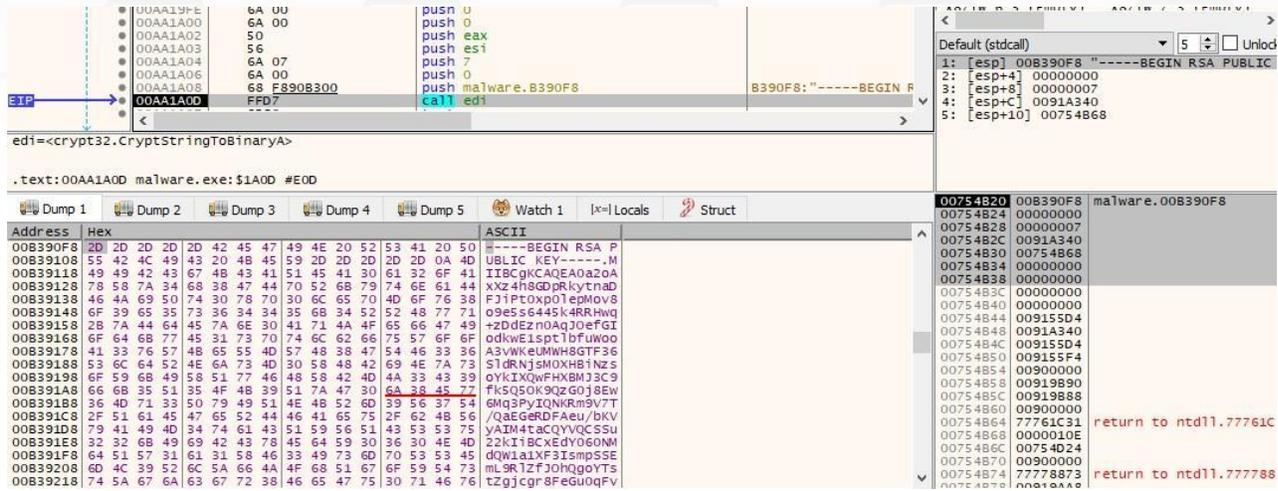


Figure 8

Medusa ransomware decodes a structure of the **RSA\_CSP\_PUBLICKEYBLOB** type by calling the CryptDecodeObjectEx function (0x10001 = **X509\_ASN\_ENCODING | PKCS\_7\_ASN\_ENCODING**, 0x13 = **RSA\_CSP\_PUBLICKEYBLOB**):

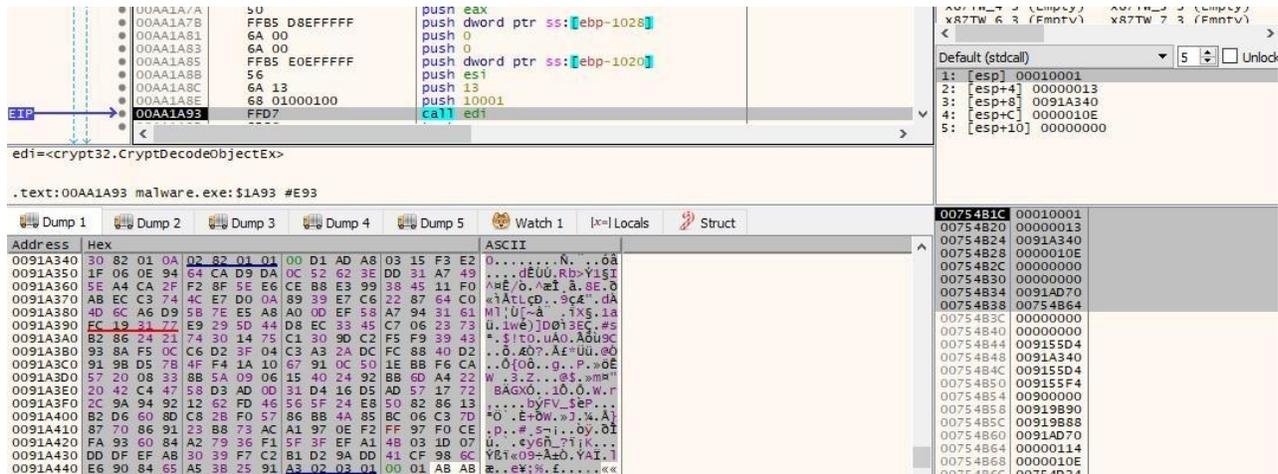


Figure 9

The process imports the RSA public key from a key BLOB using BCryptImportKeyPair:



The entire list of processes and services to terminate can be found in the Appendix.

The malware obtains the number of milliseconds that have elapsed since the system was started:

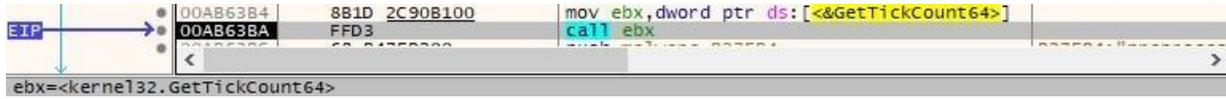


Figure 13

The ransomware stops the target services using the “net stop” command and the target processes using the “taskkill” command:

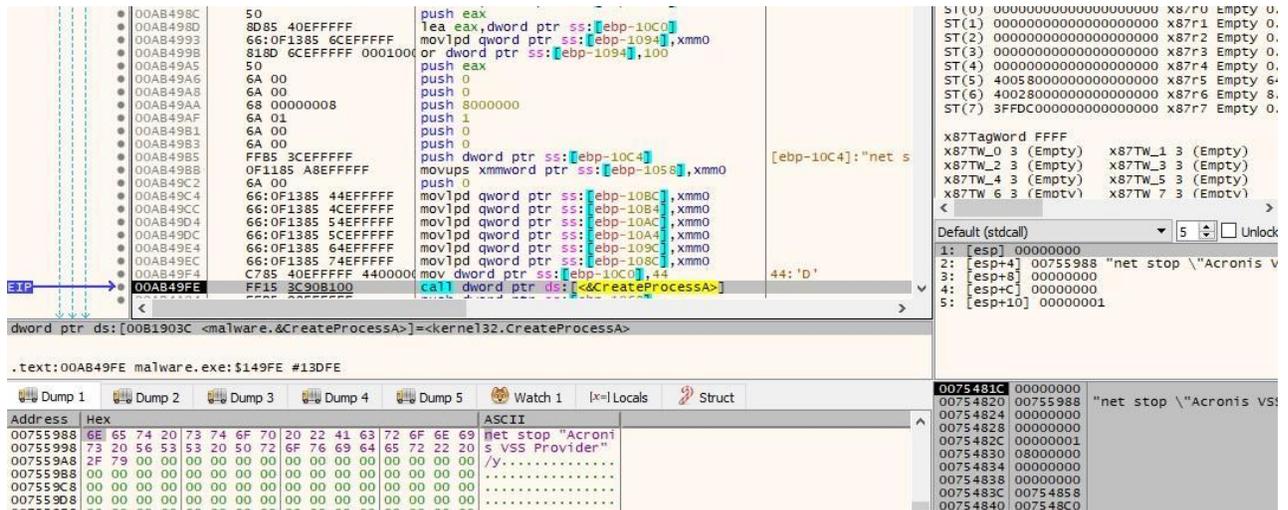


Figure 14

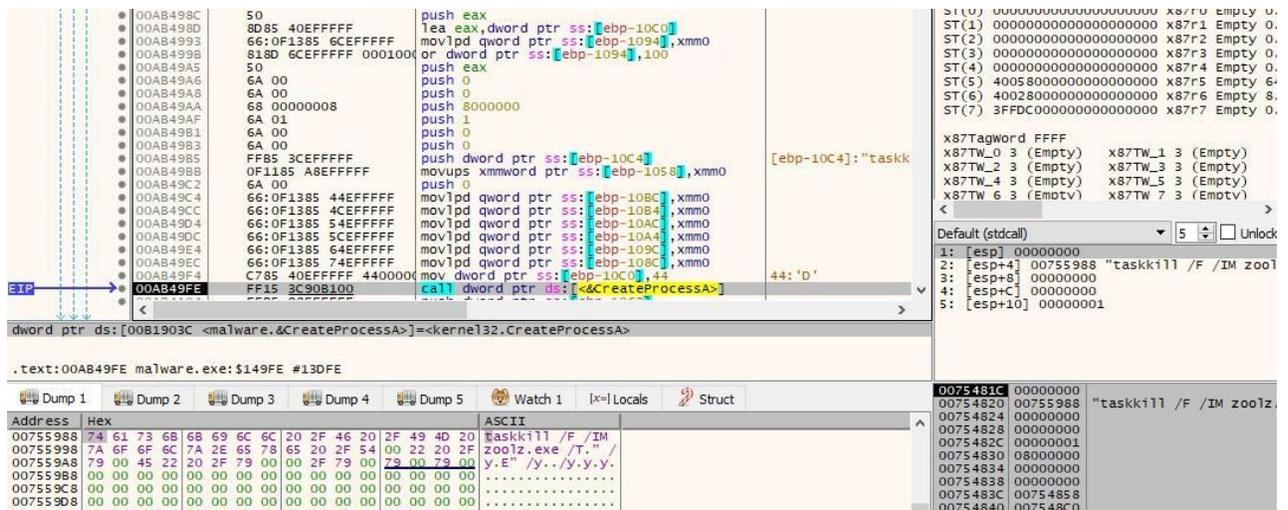


Figure 15

The process deletes the Volume Shadow Copies using the vssadmin command, as highlighted below:

Figure 16

GetLogicalDriveStringsW is utilized to extract the valid drives in the system (Figure 17):

Figure 17

The executable retrieves the drive type via a function call to GetDriveTypeW:

Figure 18

It extracts the amount of space that is available on the disk volume using the GetDiskFreeSpaceExW routine, as shown below:

Figure 19

The sample spawns two processes in order to resize the maximum amount of storage space used for shadow copy storage:

The screenshot shows a debugger window with assembly code on the left and a memory dump on the right. The assembly code includes instructions like `push eax`, `lea eax, dword ptr ss:[ebp-10C0]`, `movlpd qword ptr ss:[ebp-1094], xmm0`, and `call dword ptr ds:[<<CreateProcessA>]`. The memory dump shows the output of the `call` instruction, displaying the command `shadowstorage /f or=C:/on=C:/maxsize=401MB.....`.

Figure 20

The screenshot shows a debugger window with assembly code on the left and a memory dump on the right. The assembly code is similar to Figure 20 but includes `or=C:/on=C:/maxsize=unbounded.` in the memory dump output. The `call` instruction in the assembly code is `call dword ptr ds:[<<CreateProcessA>]`.

Figure 21

The ransomware enumerates the files on the drives using the `FindFirstFileExW` and `FindNextFileW` APIs:

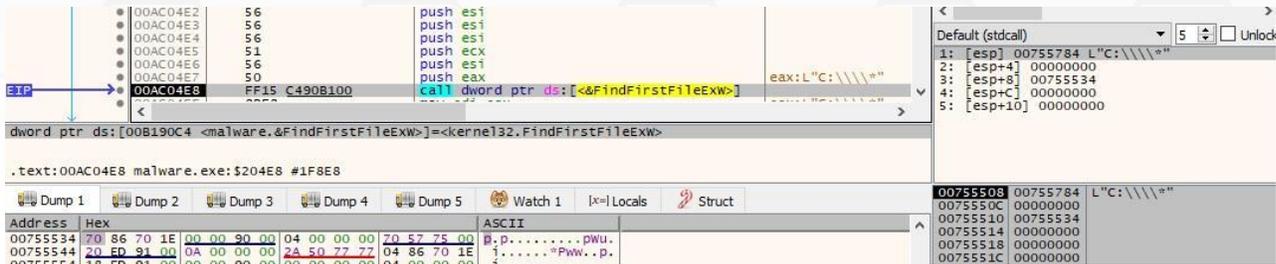


Figure 22

The following files and directories will be skipped from encryption:

```

.rdata:00B27B88 aGskpDir: ; DATA XREF: sub_AB14A0+817fo
.rdata:00B27B88 ; sub_AB14A0+83Efo
.rdata:00B27B88 text "UTF-16LE", 'g_skp_dir',0
.rdata:00B27B9C aDesktopIni: ; DATA XREF: sub_AB14A0+97Ffo
.rdata:00B27B9C text "UTF-16LE", 'desktop.ini',0
.rdata:00B27B84 aThumbsDb: ; DATA XREF: sub_AB14A0+995fo
.rdata:00B27B84 text "UTF-16LE", 'Thumbs.db',0
.rdata:00B27BC8 aWindows: ; DATA XREF: sub_AB27F0+294fo
.rdata:00B27BC8 text "UTF-16LE", 'Windows',0
.rdata:00B27BD8 aWindowsOld: ; DATA XREF: sub_AB27F0+456fo
.rdata:00B27BD8 text "UTF-16LE", 'Windows.old',0
.rdata:00B27BF0 aPerflogs: ; DATA XREF: sub_AB27F0+626fo
.rdata:00B27BF0 text "UTF-16LE", 'PerfLogs',0
.rdata:00B27C02 align 4
.rdata:00B27C04 aMsocache: ; DATA XREF: sub_AB27F0+7F6fo
.rdata:00B27C04 text "UTF-16LE", 'MSOCache',0
.rdata:00B27C16 align 4
.rdata:00B27C18 aProgramFiles: ; DATA XREF: sub_AB27F0+9EEfo
.rdata:00B27C18 text "UTF-16LE", 'Program Files',0
.rdata:00B27C34 aProgramFilesX86: ; DATA XREF: sub_AB27F0+BBDFo
.rdata:00B27C34 text "UTF-16LE", 'Program Files (x86)',0
.rdata:00B27C5C aProgramdata: ; DATA XREF: sub_AB27F0+D8DFo
.rdata:00B27C5C text "UTF-16LE", 'ProgramData',0

```

Figure 23

The GetFileExInfoStandard API is utilized to obtain attributes for a file or directory (0x0 = **GetFileExInfoStandard**):

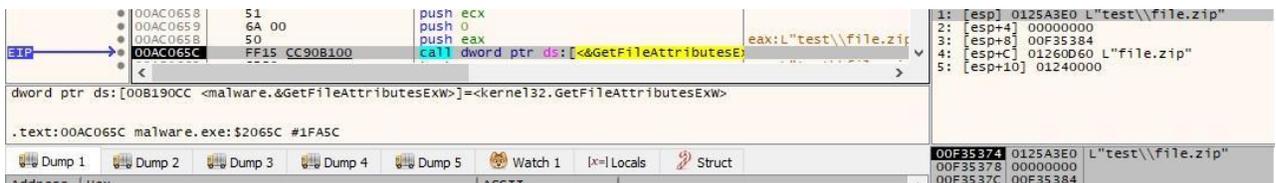


Figure 24

The ransom note called "!!!READ\_ME\_MEDUSA!!!.txt" is created in every traversed directory. It contains the victim's name and a 32-byte hash that should be used during the communication with the threat actor:

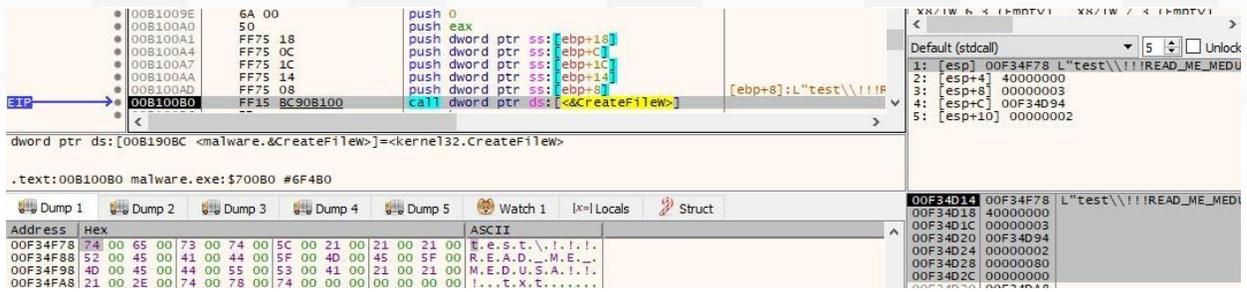


Figure 25

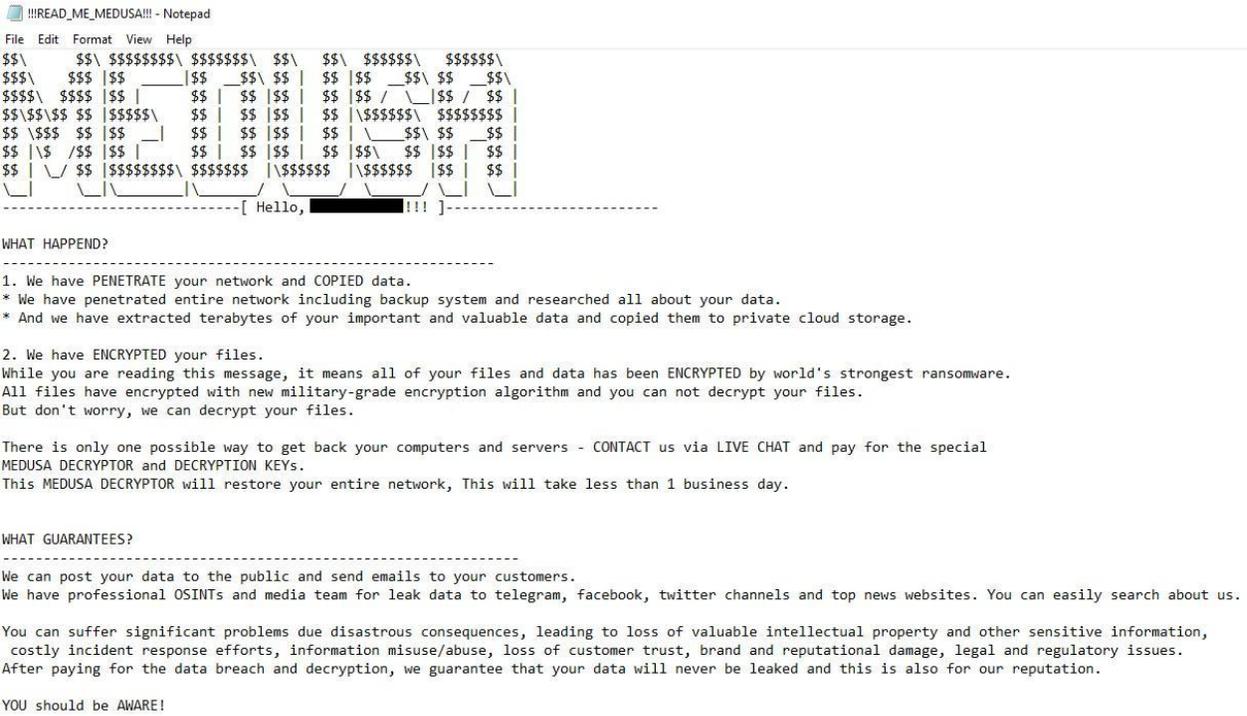


Figure 26

The sample opens a target file by calling the CreateFileW API (0xC0000000 = **GENERIC\_READ** | **GENERIC\_WRITE**, 0x3 = **FILE\_SHARE\_READ** | **FILE\_SHARE\_WRITE**, 0x3 = **OPEN\_EXISTING**, 0x80 = **FILE\_ATTRIBUTE\_NORMAL**):

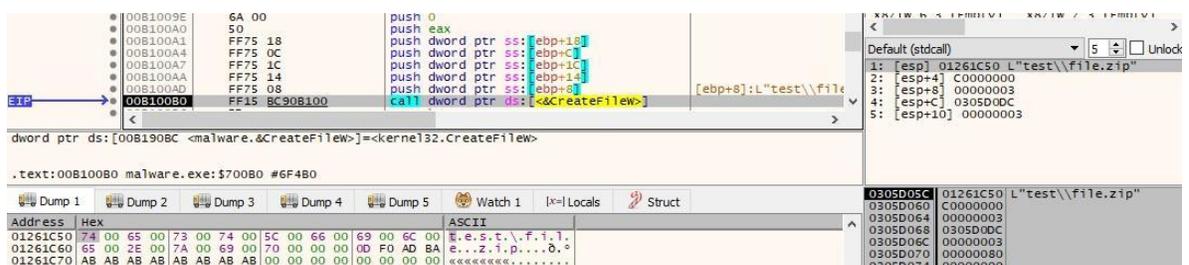


Figure 27

GetFileType is utilized to retrieve the file type, as highlighted below:

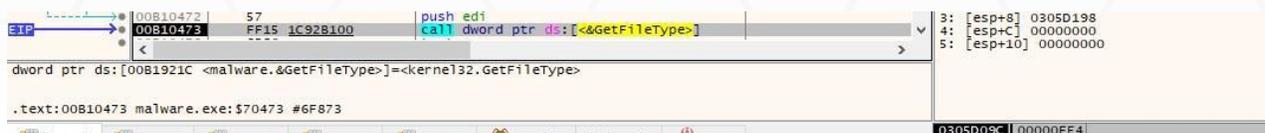


Figure 28

The malicious process moves the file pointer of the target file via a function call to SetFilePointerEx (see Figure 29).

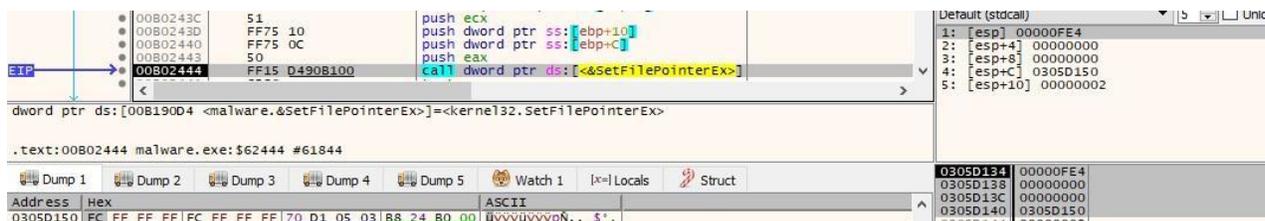


Figure 29

Each file is read by calling the ReadFile function:

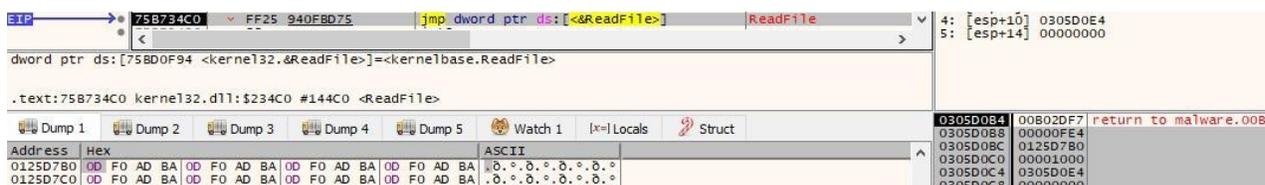


Figure 30

The BCryptGetProperty API is used to obtain the values of the “ObjectLength” and “BlockLength” properties for the CNG object:

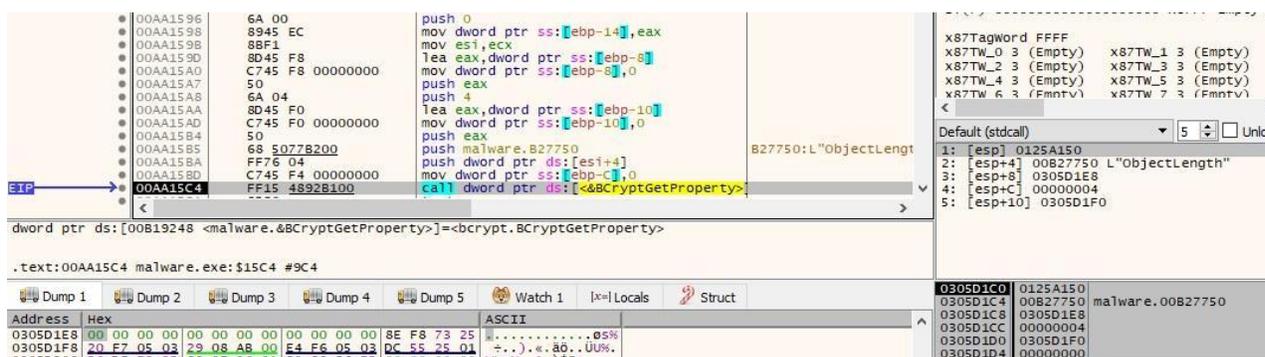


Figure 31

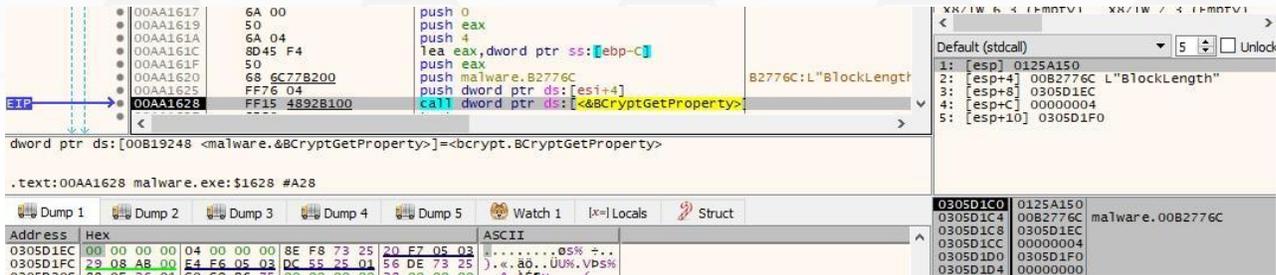


Figure 32

The following 16 bytes represent the IV (initialization vector) that is the same for all files to be encrypted:

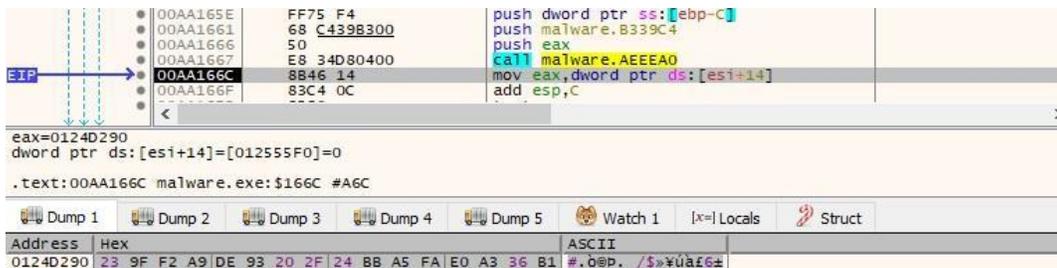


Figure 33

The AES chaining mode is set to cipher block chaining using the BCryptSetProperty routine:



Figure 34

The malware creates a key object based on 32 bytes that were generated, which represent the AES256 key that is changing between iterations:

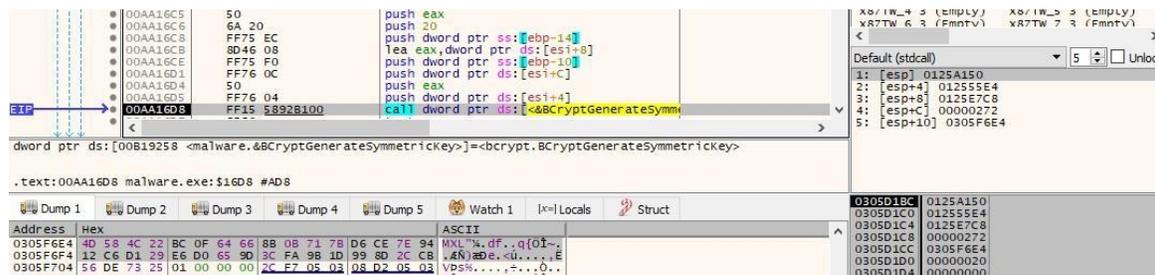


Figure 35

| Address  | Hex   | ASCII               |
|----------|---|---------------------|
| 0125E7C8 | 00 00 00 00 00 00 00 00 14 00 00 00 52 55 55 55 | .....RUUU           |
| 0125E7D8 | 50 A1 25 01 F0 E7 25 01 00 00 00 00 00 00 00 00 | Pi%.0c%.....        |
| 0125E7E8 | 00 00 00 00 00 00 00 00 4A 02 00 00 48 53 53 4D | .....J...KSSM       |
| 0125E7F8 | 02 00 01 00 00 00 00 00 10 00 00 00 10 00 00 00 | .....               |
| 0125E808 | 00 01 00 00 00 00 00 00 38 A2 25 01 00 00 00 00 | .....8e%.....       |
| 0125E818 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....               |
| 0125E828 | 20 00 00 00 4D 58 4C 22 BC 0F 64 66 88 08 71 78 | ...MXL"%4.df..q{    |
| 0125E838 | D6 CE 7E 94 12 C6 D1 29 E6 D0 65 9D 3C FA 98 1D | Öi~.~.ÄN)æDe.<ü..   |
| 0125E848 | 99 8D 2C CB 00 00 00 00 4D 58 4C 22 BC 0F 64 66 | ...É...MXL"%4.df    |
| 0125E858 | 88 08 71 78 D6 CE 7E 94 12 C6 D1 29 E6 D0 65 9D | ..q{0i~.~.ÄN)æDe.   |
| 0125E868 | 3C FA 98 1D 99 8D 2C CB 11 29 53 CC AD 26 37 AA | <ü...É.}S1.&7ª      |
| 0125E878 | 26 2D 46 D1 F0 E3 38 45 9E D7 D6 47 78 07 B3 DA | &-FN0ª8E.X0GX.*U    |
| 0125E888 | 44 FD 28 C7 DD 70 04 0C 42 DB AD 0D EF FD 9A A7 | Dy(CYp..BÜ..ÿ.ÿ     |
| 0125E898 | C9 D0 DC 76 39 33 E4 33 8C 14 BF 84 F4 13 0C 5E | EDÜv93ª3...¿.0..^   |
| 0125E8A8 | 80 EE 24 99 6D 9E 20 95 4D 6C 87 31 A2 91 1D 96 | °i\$.m..M1.1e...    |
| 0125E8B8 | 68 41 C1 E0 52 72 25 D3 8C 54 80 E2 78 47 8C BC | kAAªr%0.T.ªxG.%ª    |
| 0125E8C8 | C8 A9 A8 25 A5 37 88 B0 DF A8 60 37 7D 39 7D A1 | Eª %ª7.ªB`ª7ª}i     |
| 0125E8D8 | 16 78 BC 41 44 0A 99 92 97 33 6E AD EF 74 E2 11 | .Xª4AD...3n.ªiª.    |
| 0125E8E8 | 27 DD 4A 34 82 EA C2 84 48 8D 3F 24 35 B4 42 85 | 'YJª.ªA.H.ª?ª\$ª'B. |

Figure 36

Firstly, the AES key is encrypted using the RSA public key via a call to BCryptEncrypt:

The screenshot shows a debugger window with assembly code on the left and a stack dump on the right. The assembly code at address 00A808BF is highlighted, showing a call to BCryptEncrypt. The stack dump shows various memory locations, including one containing '&L\"SHA1\"'.

Figure 37

| Address  | Hex   | ASCII                          |
|----------|---|--------------------------------|
| 0305F530 | C4 C4 0E 59 06 12 C0 AE 65 16 95 60 65 09 7F 8C | AA.Y..Aªe...e...               |
| 0305F540 | 1F 00 45 82 6D E7 2C 59 BE 18 49 56 4E BC E6 05 | ..EªmªcªYª.IVNªæ.              |
| 0305F550 | 5A 61 1A 12 BA 48 56 53 3B 71 6C 33 1E FD A2 0D | Zª..ªHVªS;qª13.Yªe.            |
| 0305F560 | 45 6C 42 64 61 78 98 C6 37 CA EF 54 07 2B 59 C5 | E1ªdªa{.ª7ªEªIªT..+Yª          |
| 0305F570 | F1 9A 4E 3D 52 E8 5A B9 09 16 4E 8C 31 2B 84 67 | ªn.N=ReªZ`ª..N.ªi+ªg           |
| 0305F580 | 96 CB CA A7 9E F7 0C E2 5D 5A BE 90 C6 71 B9 28 | .ªEªÿ.ª.ªàª}ªZª%.ªqª'+         |
| 0305F590 | 01 DF 8D 46 11 AE E9 6C 5A 77 56 90 43 83 5F 48 | .ªBª.Fª.ªeªiªZªwªVª.Cª_K       |
| 0305F5A0 | BB C9 8C D8 63 30 94 8E BA B3 2A EB 0C F5 42 CA | ª»ªEª.ª0ªcª0...ªªªªeª.ª0ªBªEª  |
| 0305F5B0 | CC 98 CA 86 BA BE DD ED 31 38 15 08 3B 33 7D 03 | Iª.ªEª.ªªyªiª18...ª}ª.         |
| 0305F5C0 | 09 EC F1 87 2F 20 C3 BD 4B 88 F4 A7 82 7C 34 EE | .ªiªn./ªAªyªKª.ª0ªÿª.ª}ªiª     |
| 0305F5D0 | AF A6 CB 85 D3 FC D3 D8 50 86 2C 2E F3 CE C5 F2 | .ªiªEª.ª0ªUª0ªPª...ª.ª0ªiªAª0ª |
| 0305F5E0 | 22 A6 87 F8 CD B1 6A 69 70 FC E9 93 BC 87 A0 C6 | "ª.ª.ª0ªiªªjªiªpªiªeª.%ª.ªª    |
| 0305F5F0 | 1E 0D 17 D2 88 D7 E0 BA 6C DC 3C 34 0E 41 B6 DA | ..ª.ª0ª.xªª0ªiªÜª<ªAªªÿªÜ      |
| 0305F600 | 54 83 A7 1A 95 9D 5A 13 2B 1D 40 23 D9 27 E2 8B | Tªªsª..ª.ª.ª+ª@ª#ªUªªª.        |
| 0305F610 | 8A B0 09 2A AA 2ª C 7E 9D 75 C9 90 69 96 41 E6  | .ª.ªªªªªªªª.ªUªEª.ª1ªAªª       |
| 0305F620 | 92 AF 32 04 67 B7 68 7A 80 43 E0 20 7D A7 30 02 | .ª.ª2ª.gª.hªzª.Cªª}ª0ª.        |

Figure 38

The file content is encrypted using the AES256 algorithm, as highlighted in Figure 39.

```

00AA1746 6A 00      push 0
00AA1748 57        push edi
00AA1749 50        push eax
00AA174A FF75 F4   push dword ptr ss:[ebp-C]
00AA174D 51        push ecx
00AA174E 52        push edx
00AA174F 50        push 0
00AA1751 50        push eax
00AA1752 53        push ebx
00AA1753 FF75 F8   push dword ptr ss:[ebp-8]
00AA1756 FF15 5092B100 call dword ptr ds:[<&BCryptEncrypt>]

```

Register Window:

- Default (stdcall)
- 1: [esp] 0125E7D0
- 2: [esp+4] 0305E278
- 3: [esp+8] 00001000
- 4: [esp+C] 00000000
- 5: [esp+10] 0124D290

Dump Window:

| Address  | Hex   | ASCII            |
|----------|---|------------------|
| 0305E278 | 50 48 03 04 14 00 00 00 08 00 CA 8B 12 53 9C 22 | PK.....É..S..    |
| 0305E288 | B8 13 ED AB 13 00 78 53 2B 00 0B 00 00 00 70 72 | ..i«.XS+....pr   |
| 0305E298 | 6F 63 65 78 70 2E 65 78 65 E4 8D 78 7C 94 C5 F5 | ocexp.exe&{ .A0  |
| 0305E2A8 | 3F BE 49 16 58 64 61 83 06 45 45 89 82 D5 1A 54 | ?AI.xda..EE..O.T |
| 0305E2B8 | 14 50 69 C0 72 5B 40 61 61 43 CC 2E 5E 02 2A 90 | PIAr@aaCI.A.*    |
| 0305E2C8 | AE 11 15 61 17 F1 12 05 37 0B 59 1F 56 A3 A2 C3 | ..a..ñ..7.Y.V&A  |

Figure 39

The encrypted data is written back to the file by calling the WriteFile function:

```

00B05AA2 6A 00      push 0
00B05AA4 AB        stosd
00B05AA5 AB        stosd
00B05AA6 8D45 DC   lea eax,dword ptr ss:[ebp-24]
00B05AA9 50        push eax
00B05AAA FF75 F0   push dword ptr ss:[ebp-10]
00B05AAD 53        push ebx
00B05AAE 51        push ecx
00B05AAF FF15 0892B100 call dword ptr ds:[<&writeFile>]

```

Register Window:

- Default (stdcall)
- 1: [esp] 00000FE4
- 2: [esp+4] 0305D278
- 3: [esp+8] 00001000
- 4: [esp+C] 0305D0C0
- 5: [esp+10] 00000000

Dump Window:

| Address  | Hex   | ASCII          |
|----------|---|----------------|
| 0305D278 | 47 B5 9F 8B 0A 1B 92 35 9A F0 5D AB DE 18 79 87 | µ.....5.ð<P.y. |

Figure 40

The ransomware appends the “.MEDUSA” extension to all encrypted files (see Figure 41).

```

00AB1164 50        push eax
00AB1165 57        push edi
00AB1166 FF15 4090B100 call dword ptr ds:[<&MoveFilew>]

```

Register Window:

- Default (stdcall)
- 1: [esp+4] 0305F278 L"test\\file.zip.MEDU
- 2: [esp+8] 25730E56
- 3: [esp+C] 01260E88
- 4: [esp+10] 75B8C9C0 <kernel32.GetTickCou

Dump Window:

| Address  | Hex      | ASCII                    |
|----------|----------|--------------------------|
| 0305D200 | 01261C50 | L"test\\file.zip"        |
| 0305D204 | 0305F278 | L"test\\file.zip.MEDUSA" |

Figure 41

An encrypted file has the following structure: Encrypted file content + “MEDUSA” string + file length + Encrypted AES key with RSA + “Company identification hash” (Figure 42).

```

28:7290h: 3E A1 80 99 DE 3D F1 98 2D A2 AB 4D 64 0B D3 6A > ;€™p=ñ"-C«Md.0j
28:72A0h: 25 52 05 68 89 EC 9A F7 D4 22 CB B8 9D 75 74 0E %R.h%is÷0"E.ut.
28:72B0h: 70 9B AD BB 1D 50 98 FF DB 7D 42 5D 48 56 05 C9 p>-».P"yÜ}B]HV.É
28:72C0h: 4D 45 44 55 53 41 00 00 BA 72 28 00 00 00 00 00 MEDUSA..°r(....
28:72D0h: 00 02 00 00 00 00 00 00 C4 C4 0E 59 06 12 C0 AE .....AA.Y..À@
28:72E0h: 65 16 95 60 65 09 7F 8C 1F 00 45 B2 6D E7 2C 59 e..'e..E.E²mc,Y
28:72F0h: BE 18 49 56 4E BC E6 05 5A 61 1A 12 BA 48 56 53 %I.VN%æ.Za..°HVS
28:7300h: 3B 71 6C 33 1E FD A2 0D 45 6C 42 64 61 7B 98 C6 ;q13.yc.ElBda{~Æ
28:7310h: 37 CA EF 54 07 2B 59 C5 F1 9A 4E 3D 52 E8 5A B9 7ÉiT.+YÅñ$N=RèZ'
28:7320h: 09 16 4E 8C 31 2B 84 67 96 CB CA A7 9E F7 0C E2 ..NÆ1+,,g-ÉÈ$ž÷.â
28:7330h: 5D 5A BE 90 C6 71 B9 2B 01 DF 8D 46 11 AE E9 6C ]Z%Æq'+.β.F.®Él
28:7340h: 5A 77 56 90 43 B3 5F 4B BB C9 8C D8 63 30 94 8E ZwV.C³_K»ÉE0c0"Z
28:7350h: BA B3 2A EB 0C F5 42 CA CC 98 CA 86 BA BE DD ED °³*è.òBÈI"É†°%Yi
28:7360h: 31 38 15 08 3B 33 7D 03 09 EC F1 87 2F 20 C3 BD 18.;}3)..iñ†/ Åñ
28:7370h: 4B 88 F4 A7 82 7C 34 EE AF A6 CB 85 D3 FC D3 D8 K'òš,|4i~|E.0ú00
28:7380h: 50 86 2C 2E F3 CE C5 F2 22 A6 87 F8 CD B1 6A 69 P†,.óIÀò"†±jji
28:7390h: 70 FC E9 93 BC 87 A0 C6 1E 0D 17 D2 B8 D7 E0 BA pué"¼†Æ...0,xà°
28:73A0h: 6C DC 3C 34 0E 41 B6 DA 54 B3 A7 1A 95 9D 5A 13 lÜ<4.A¶UT³š,.Z.
28:73B0h: 2B 1D 40 23 D9 27 E2 8B 8A B0 09 2A AA 2A C7 E6 +.®#Ù'â<š°. *a*Çæ
28:73C0h: 9D 75 C9 90 69 96 41 E6 92 AF 32 04 67 B7 68 7A .uÉ.i-Aæ'2.g.hz
28:73D0h: 80 43 E0 20 7D A7 30 02 €Cà }š0.

```

Figure 42

Finally, if the “-d” parameter is not specified, the malware deletes itself:

```

00AB498C 50          push eax
00AB498D 8D85 40EFFFFF lea eax,dword ptr ss:[ebp-10C0]
00AB4993 66:0F1385 6CEFFFFF movlpd qword ptr ss:[ebp-1094],xmm0
00AB499B 818D 6CEFFFFF 00010000 or dword ptr ss:[ebp-1094],100
00AB49A5 50          push eax
00AB49A6 6A 00       push 0
00AB49A8 6A 00       push 0
00AB49AA 68 00000008 push 80000000
00AB49AF 6A 01       push 1
00AB49B1 6A 00       push 0
00AB49B3 6A 00       push 0
00AB49B5 FF85 3CEFFFFF push dword ptr ss:[ebp-10C4]
00AB49BB 0F1185 A8EFFFFF movups xmmword ptr ss:[ebp-1058],xmm0
00AB49C2 6A 00       push 0
00AB49C4 66:0F1385 44EFFFFF movlpd qword ptr ss:[ebp-108C],xmm0
00AB49CC 66:0F1385 4CEFFFFF movlpd qword ptr ss:[ebp-1084],xmm0
00AB49D4 66:0F1385 54EFFFFF movlpd qword ptr ss:[ebp-10A4],xmm0
00AB49DC 66:0F1385 5CEFFFFF movlpd qword ptr ss:[ebp-10A4],xmm0
00AB49E4 66:0F1385 64EFFFFF movlpd qword ptr ss:[ebp-109C],xmm0
00AB49EC 66:0F1385 74EFFFFF movlpd qword ptr ss:[ebp-108C],xmm0
00AB49F4 C785 40EFFFFF 44000000 mov dword ptr ds:[ebp-10C0],44
00AB49FE FF15 2C90B100 call dword ptr ds:[<CreateProcessA>]

```

```

dword ptr ds:[00B1903C <malware.&CreateProcessA>]=<kernel32.CreateProcessA>

```

```

.txt:00AB49FE malware.exe:$149FE #13DFE

```

```

cmd /c ping localhost

```

Figure 43

Medusa ransomware excludes the System folder from encryption by running with the “-f” parameter:

```
Command Prompt
C:\Users\User\Desktop>malware.exe -f -v
--start--
:exclude systemfolder
:initial run powershell from predefined variable.

default key:0
preprocess
kill_services processes
kill_services Acronis VSS Provider
kill_services Enterprise Client Service
kill_services Sophos Agent
kill_services Sophos AutoUpdate Service
kill_services Sophos Clean Service
kill_services Sophos Device Control Service
kill_services Sophos File Scanner Service
kill_services Sophos Health Service
kill_services Sophos MCS Agent
kill_services Sophos MCS Client
kill_services Sophos Message Router
kill_services Sophos Safestore Service
kill_services Sophos System Protection Service
kill_services Sophos Web Control Service
kill_services SQLsafe Backup Service
kill_services SQLsafe Filter Service
kill_services Symantec System Recovery
kill_services Veeam Backup Catalog Data Service
```

Figure 44

The malware can encrypt a specific folder using the “-i” parameter and load the RSA public key from a file mentioned in the “-k” parameter. The ransom note can be changed with a file mentioned in the “-t” parameter.

The C drive is not encrypted if it runs with the “-s” parameter, and the sample doesn’t stop the target processes/services and doesn’t delete the Volume Shadow Copies if the “-p” parameter is specified:

```
Command Prompt
C:\Users\User\Desktop>malware.exe -s -p -v
--start--
:exclude systemdrive
:do not use preprocess
:initial run powershell from predefined variable.

default key:0
preprocess
:System
encrypt system
C:\
Total time:0
```

Figure 45

The ransomware can execute a PowerShell script using the “-w” parameter, as highlighted in the figure below.

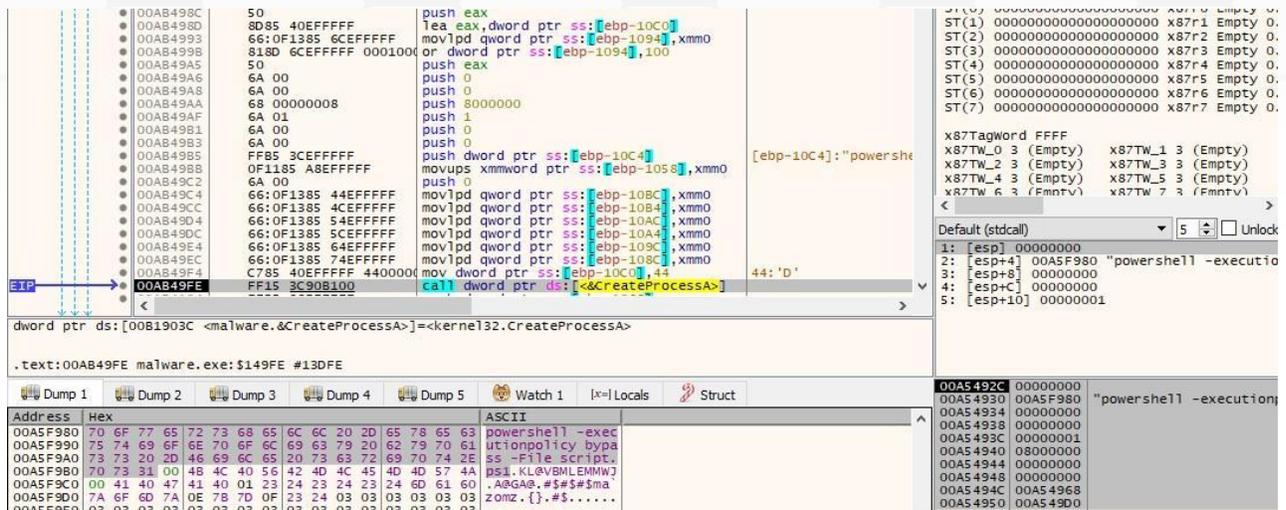


Figure 46

The last parameter, “-V,” displays the Medusa ransomware version:

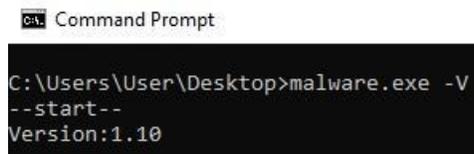


Figure 47

## Indicators of Compromise

### Medusa Ransom Note

!!!READ\_ME\_MEDUSA!!!.txt

### Processes spawned

powershell -Command "& {}"

net stop <service name> /y

taskkill /F /IM <process name> /T

vssadmin Delete Shadows /all /quiet

vssadmin resize shadowstorage /for=C: /on=C: /maxsize=401MB

vssadmin resize shadowstorage /for=C: /on=C: /maxsize=unbounded

cmd /c ping localhost -n 3 > nul & del <Executable>

# Appendix

## List of services

"Acronis VSS Provider" "Enterprise Client Service" "Sophos Agent" "Sophos AutoUpdate Service" "Sophos Clean Service" "Sophos Device Control Service" "Sophos File Scanner Service" "Sophos Health Service" "Sophos MCS Agent" "Sophos MCS Client" "Sophos Message Router" "Sophos Safestore Service" "Sophos System Protection Service" "Sophos Web Control Service" "SQLsafe Backup Service" "SQLsafe Filter Service" "Symantec System Recovery" "Veeam Backup Catalog Data Service" "AcronisAgent" "AcrSch2Svc" "Antivirus" "ARSM" "BackupExecAgentAccelerator" "BackupExecAgentBrowser" "BackupExecDeviceMediaService" "BackupExecJobEngine" "BackupExecManagementService" "BackupExecRPCService" "BackupExecVSSProvider" "bedbg" "DCAgent" "EPSecurityService" "EPUupdateService" "EraserSvc11710" "EsgShKernel" "FA\_Scheduler" "IISAdmin" "IMAP4Svc" "macmnsvc" "masvc" "MBAMService" "MBEndpointAgent" "McAfeeEngineService" "McAfeeFramework" "McAfeeFrameworkMcAfeeFramework" "McShield" "McTaskManager" "mfemms" "mfevtp" "MMS" "mozyprobackup" "MsDtsServer" "MsDtsServer100" "MsDtsServer110" "MSEExchangeES" "MSEExchangeIS" "MSEExchangeMGMT" "MSEExchangeMTA" "MSEExchangeSA" "MSEExchangeSRS" "MSOLAP\$SQL\_2008" "MSOLAP\$SYSTEM\_BGC" "MSOLAP\$TPS" "MSOLAP\$TPSAMA" "MSSQL\$BKUPEXEC" "MSSQL\$ECWDB2" "MSSQL\$PRACTICEMGT" "MSSQL\$PRACTICEBGC" "MSSQL\$PROFXENGAGEMENT" "MSSQL\$SBSMONITORING" "MSSQL\$SHAREPOINT" "MSSQL\$SQL\_2008" "MSSQL\$SYSTEM\_BGC" "MSSQL\$TPS" "MSSQL\$TPSAMA" "MSSQL\$VEEAMSQL2008R2" "MSSQL\$VEEAMSQL2012" "MSSQLFDLauncher" "MSSQLFDLauncher\$PROFXENGAGEMENT" "MSSQLFDLauncher\$SBSMONITORING" "MSSQLFDLauncher\$SHAREPOINT" "MSSQLFDLauncher\$SQL\_2008" "MSSQLFDLauncher\$SYSTEM\_BGC" "MSSQLFDLauncher\$TPS" "MSSQLFDLauncher\$TPSAMA" "MSSQLSERVER" "MSSQLServerADHelper100" "MSSQLServerOLAPService" "MySQL80" "MySQL57" "ntrtscan" "OracleClientCache80" "PDVFSservice" "POP3Svc" "ReportServer" "ReportServer\$SQL\_2008" "ReportServer\$SYSTEM\_BGC" "ReportServer\$TPS" "ReportServer\$TPSAMA" "RESvc" "sacsvr" "SamSs" "SAVAdminService" "SAVService" "SDRSVC" "SepMasterService" "ShMonitor" "Smcinst" "SmcService" "SMTPSvc" "SNAC" "SntpService" "sophossp" "SQLAgent\$BKUPEXEC" "SQLAgent\$ECWDB2" "SQLAgent\$PRACTICEBGC" "SQLAgent\$PRACTICEMGT" "SQLAgent\$PROFXENGAGEMENT" "SQLAgent\$SBSMONITORING" "SQLAgent\$SHAREPOINT" "SQLAgent\$SQL\_2008" "SQLAgent\$SYSTEM\_BGC" "SQLAgent\$TPS" "SQLAgent\$TPSAMA" "SQLAgent\$VEEAMSQL2008R2" "SQLAgent\$VEEAMSQL2012" "SQLBrowser" "SQLSafeOLRService" "SQLSERVERAGENT" "SQLTELEMETRY" "SQLTELEMETRY\$ECWDB2" "SQLWriter" "SstpSvc" "svcGenericHost" "swi\_filter" "swi\_service" "swi\_update\_64" "TmCCSF" "tmlisten" "TrueKey" "TrueKeyScheduler" "TrueKeyServiceHelper" "UI0Detect" "VeeamBackupSvc" "VeeamBrokerSvc" "VeeamCatalogSvc" "VeeamCloudSvc" "VeeamDeploymentService" "VeeamDeploySvc" "VeeamEnterpriseManagerSvc" "VeeamMountSvc" "VeeamNFSSvc" "VeeamRESTSvc" "VeeamTransportSvc" "W3Svc" "wbengine" "WRSVC" "MSSQL\$VEEAMSQL2008R2" "SQLAgent\$VEEAMSQL2008R2" "VeeamHvIntegrationSvc" "swi\_update" "SQLAgent\$CXDB" "SQLAgent\$CITRIX\_METAFRAME" "SQL Backups" "MSSQL\$PROD" "Zoolz 2 Service" "MSSQLServerADHelper" "SQLAgent\$PROD" "msftesql\$PROD" "NetMsmqActivator" "EhttpSrv" "ekrn" "ESHASRV" "MSSQL\$SOPHOS" "SQLAgent\$SOPHOS" "AVP" "klnagent" "MSSQL\$SQLEXPRESS" "SQLAgent\$SQLEXPRESS" "wbengine" "kavfssl" "KAVFSGT" "KAVFS" "mfefire"

## List of processes

"zoolz.exe" "agntsvc.exe" "dbeng50.exe" "dbsnmp.exe" "encsvc.exe" "excel.exe" "firefoxconfig.exe" "infopath.exe" "isqlplussvc.exe" "msaccess.exe" "msftesql.exe" "mspub.exe" "mydesktopqos.exe" "mydesktopservice.exe" "mysqld.exe" "mysqld-nt.exe" "mysqld-opt.exe" "ocautoupds.exe" "ocomm.exe" "ocssd.exe" "onenote.exe" "oracle.exe" "outlook.exe" "powerpnt.exe" "sqbcoreservice.exe" "sqlagent.exe" "sqlbrowser.exe" "sqlservr.exe" "sqlwriter.exe" "steam.exe" "synctime.exe" "tbirdconfig.exe" "thebat.exe" "thebat64.exe" "thunderbird.exe" "visio.exe" "winword.exe" "wordpad.exe" "xfssvcon.exe" "tmlisten.exe" "PccNTMon.exe" "CNTAoSMgr.exe" "Ntrtscan.exe" "mbamtray.exe"