A Deep Dive into Medusa Ransomware

Prepared by: Vlad Pasca, Senior Malware & Threat Analyst

SecurityScorecard

SecurityScorecard.com
info@securityscorecard.com

Tower 49
12 E 49th Street
Suite 15-001
New York, NY 10017
1.800.682.1707

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**):

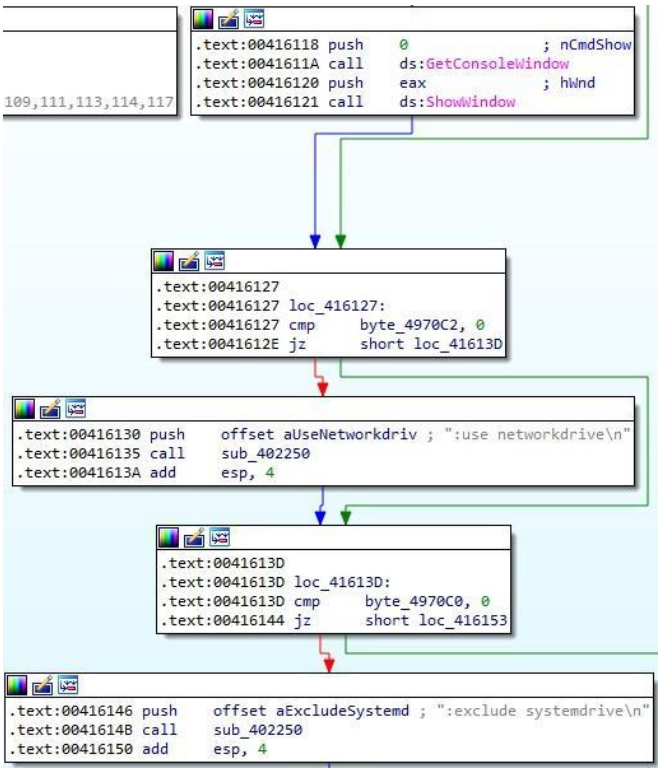


Figure 1

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

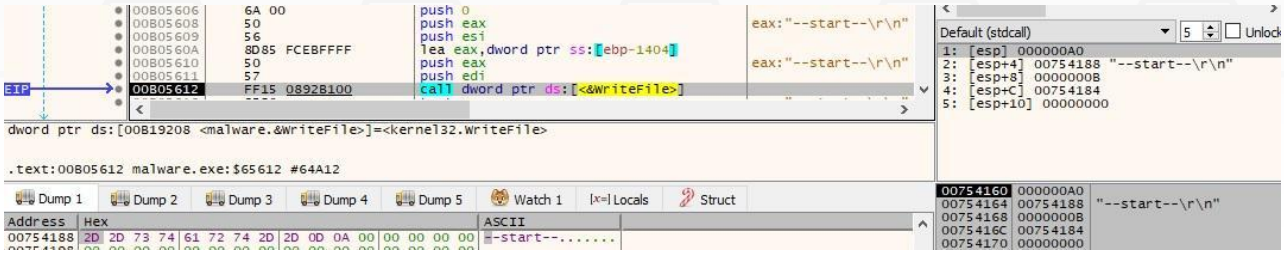


Figure 2

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

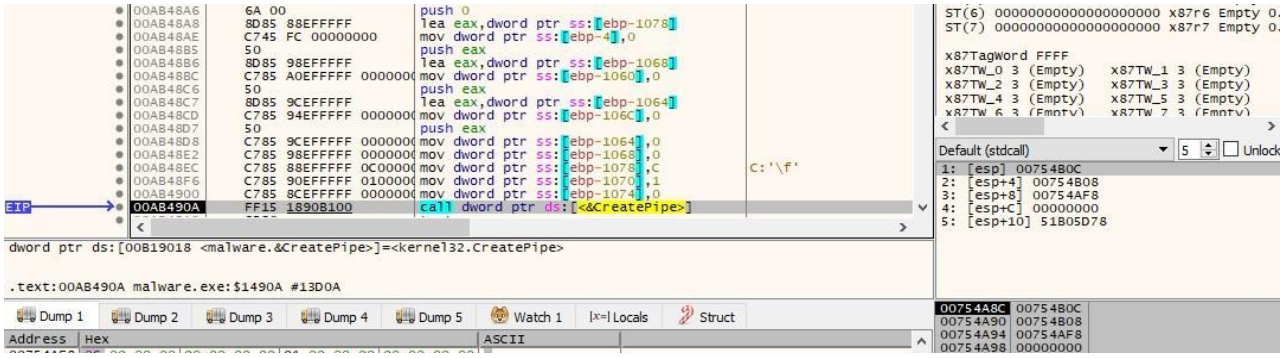


Figure 3

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

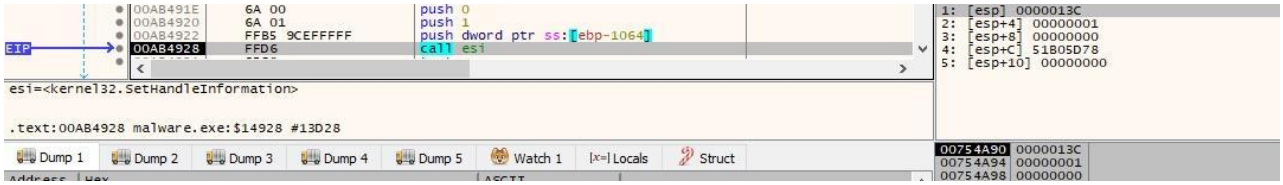


Figure 4

The malware creates a PowerShell process using the CreateProcessA API (0x08000000 = **CREATE_NO_WINDOW**):

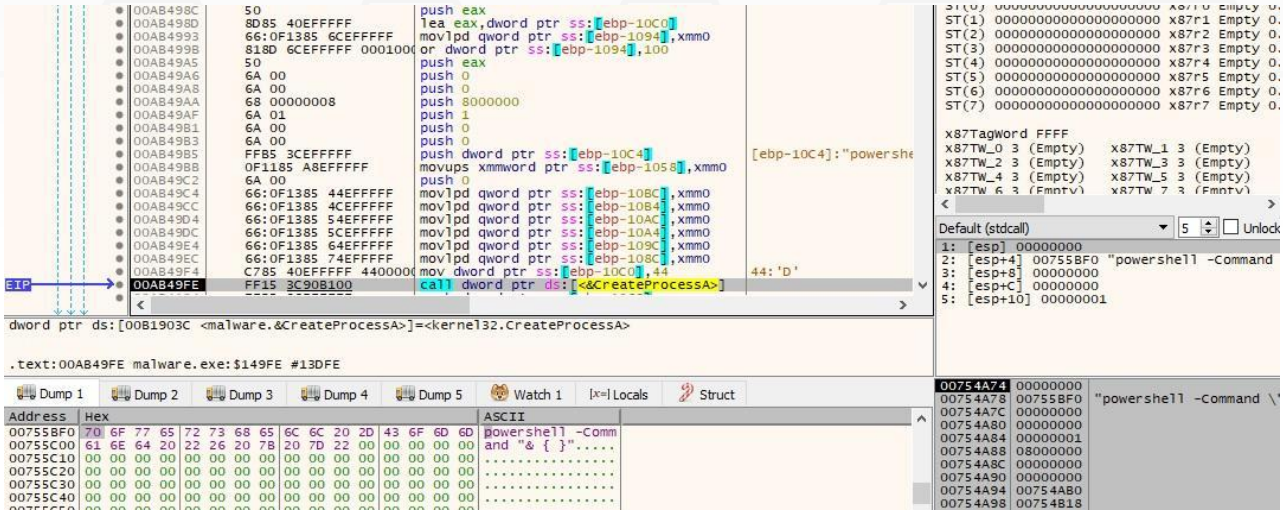


Figure 5

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

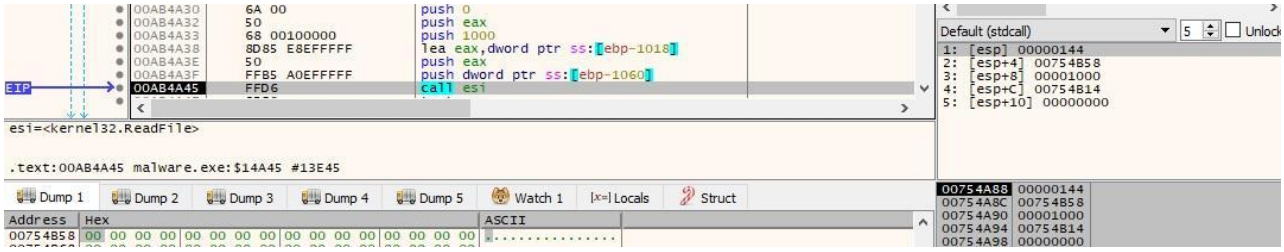


Figure 6

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

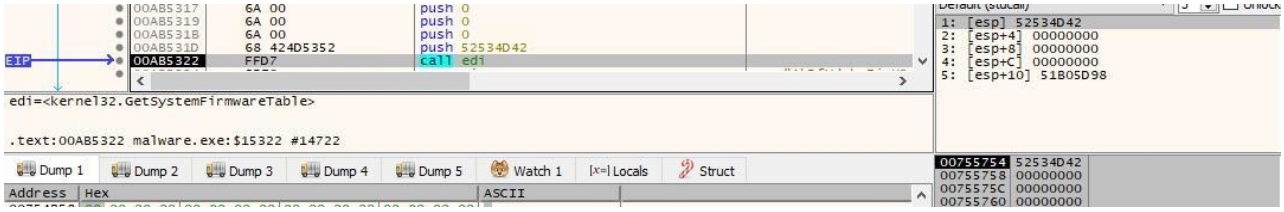


Figure 7

CryptStringToBinaryA is used to decode the RSA public key from Base64 (0x7 = **CRYPT_STRING_ANY**):

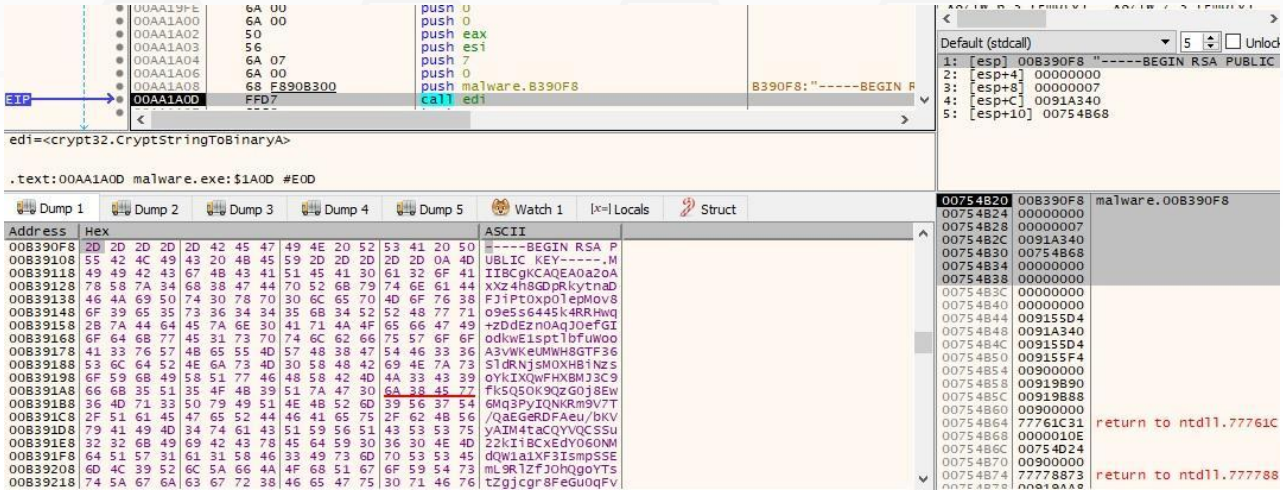


Figure 8

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**):

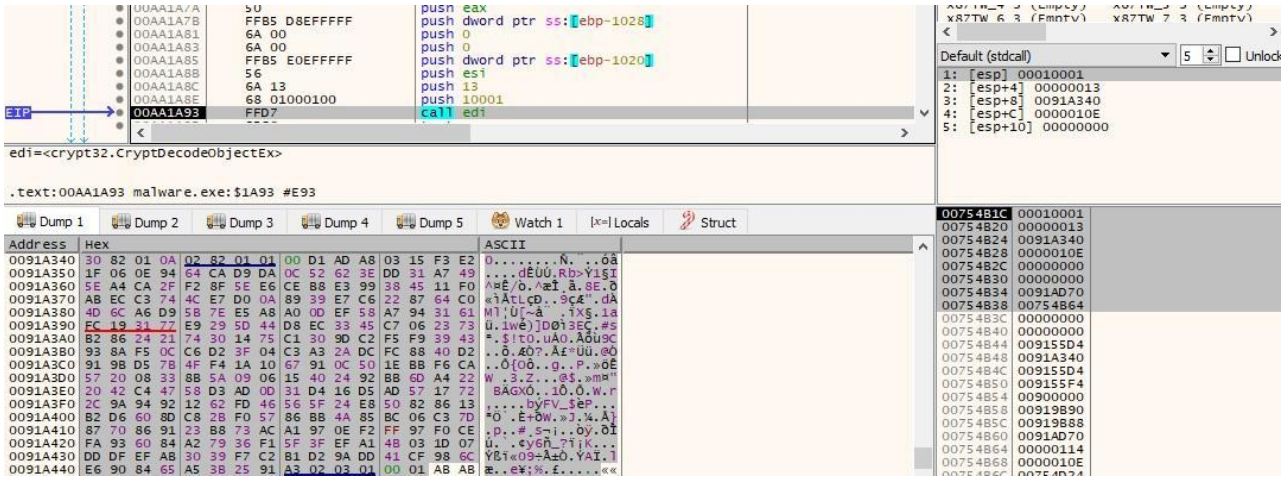


Figure 9

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

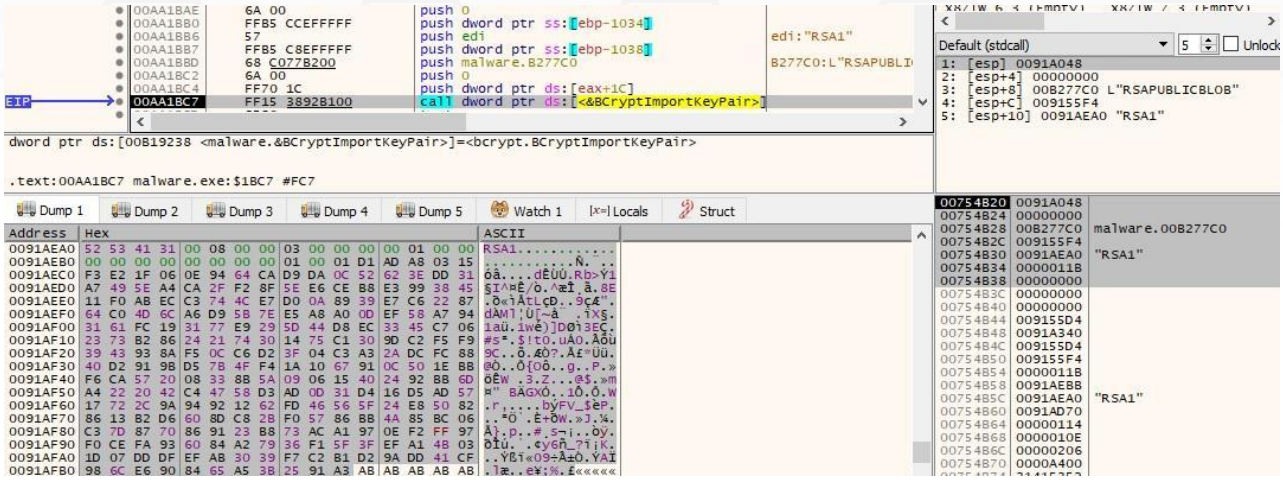


Figure 10

A list of file extensions that will be skipped is decrypted using the XOR operation with 0x2E: ".dll", ".exe", ".lnk", and ".MEDUSA". The ransomware also decrypts a list of services and processes that will be stopped:

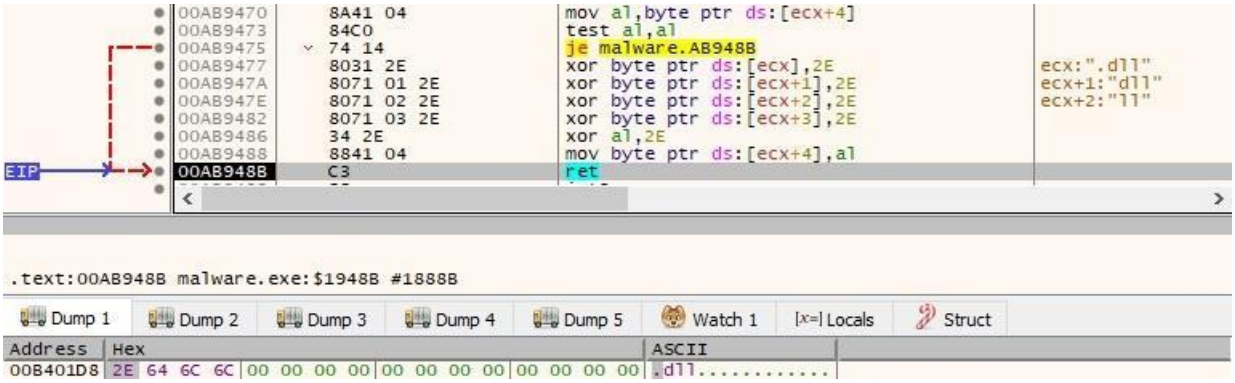


Figure 11

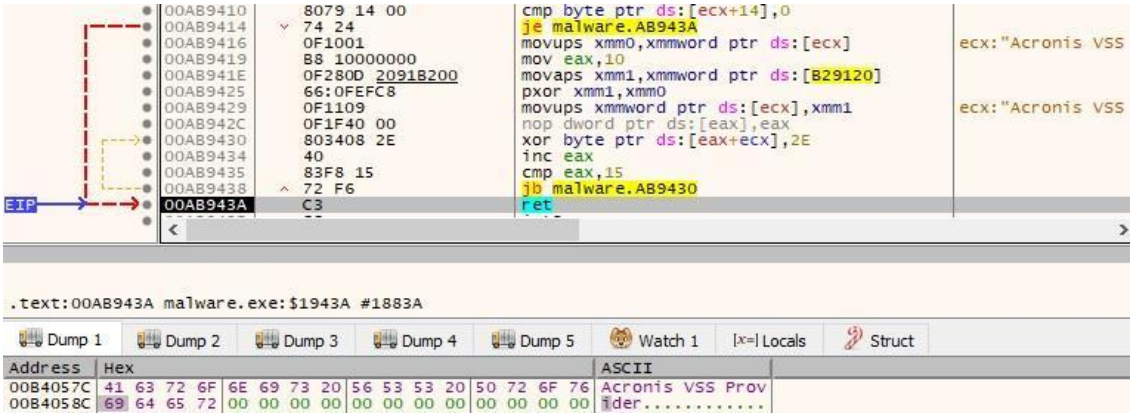


Figure 12

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:

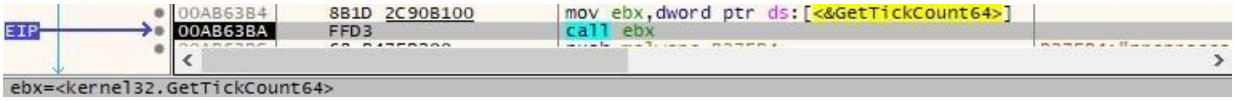


Figure 13

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

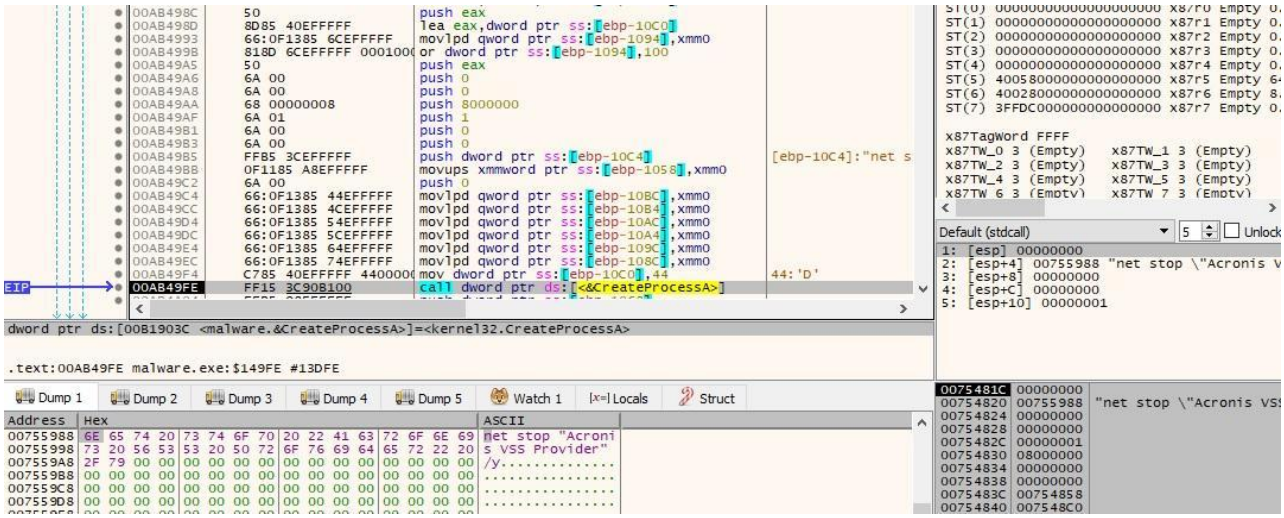


Figure 14

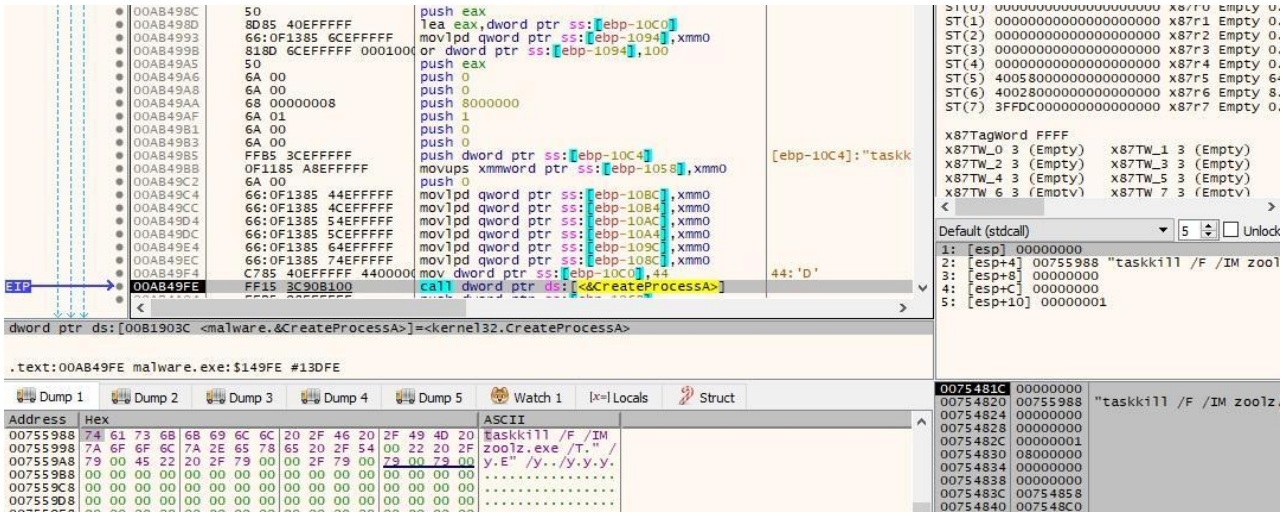


Figure 15

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

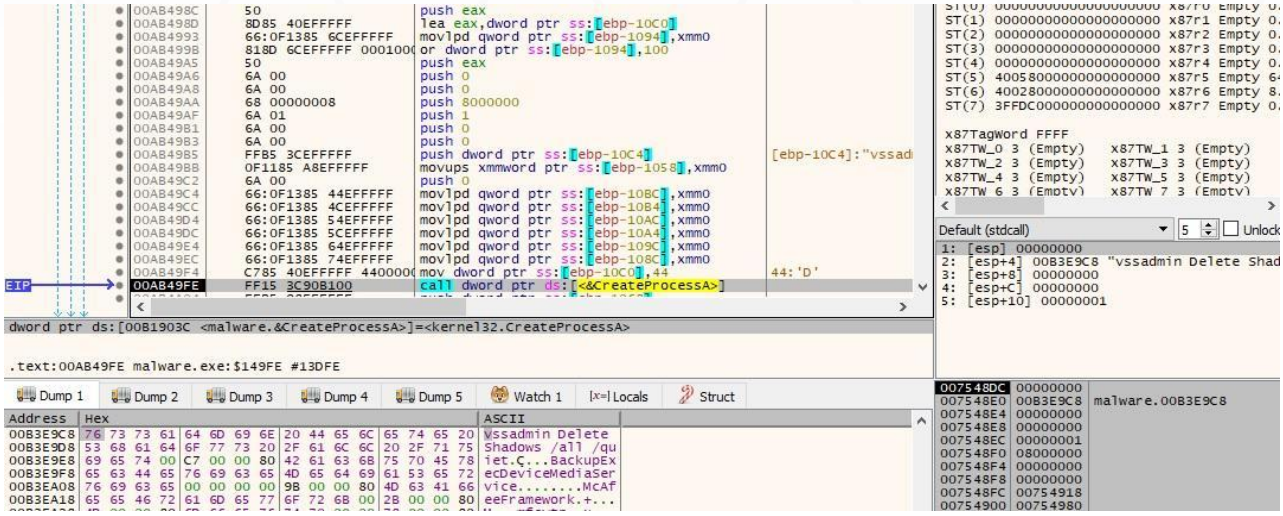


Figure 16

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:

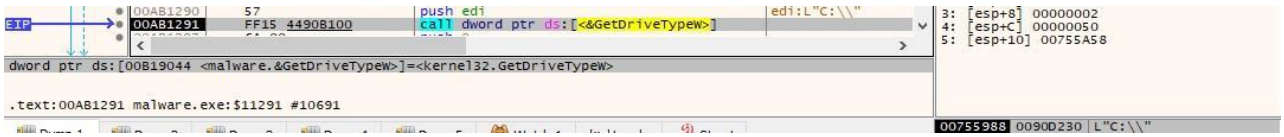


Figure 18

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:

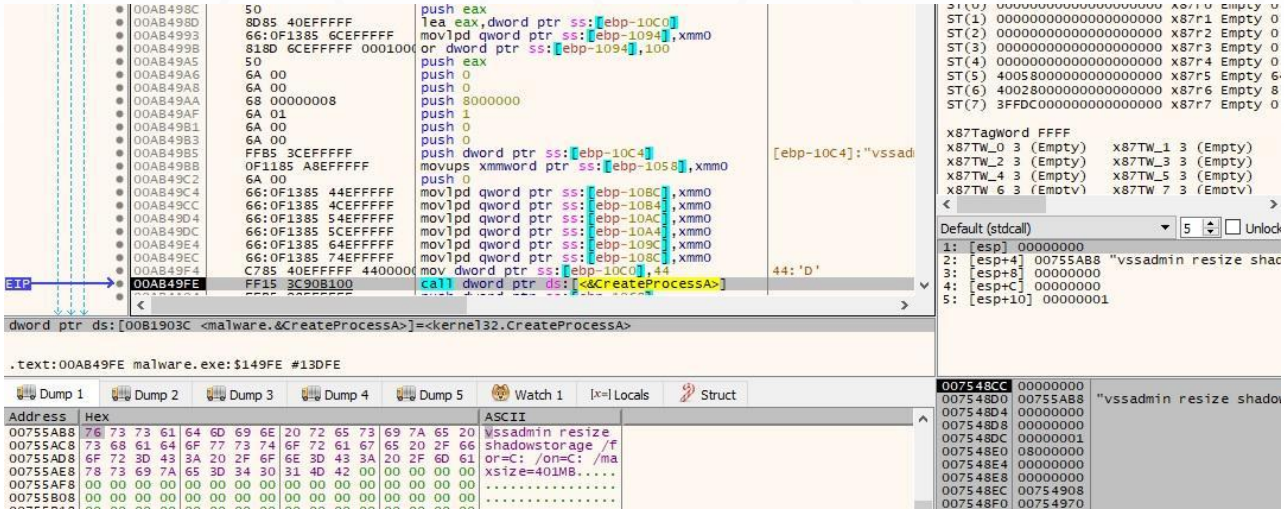


Figure 20

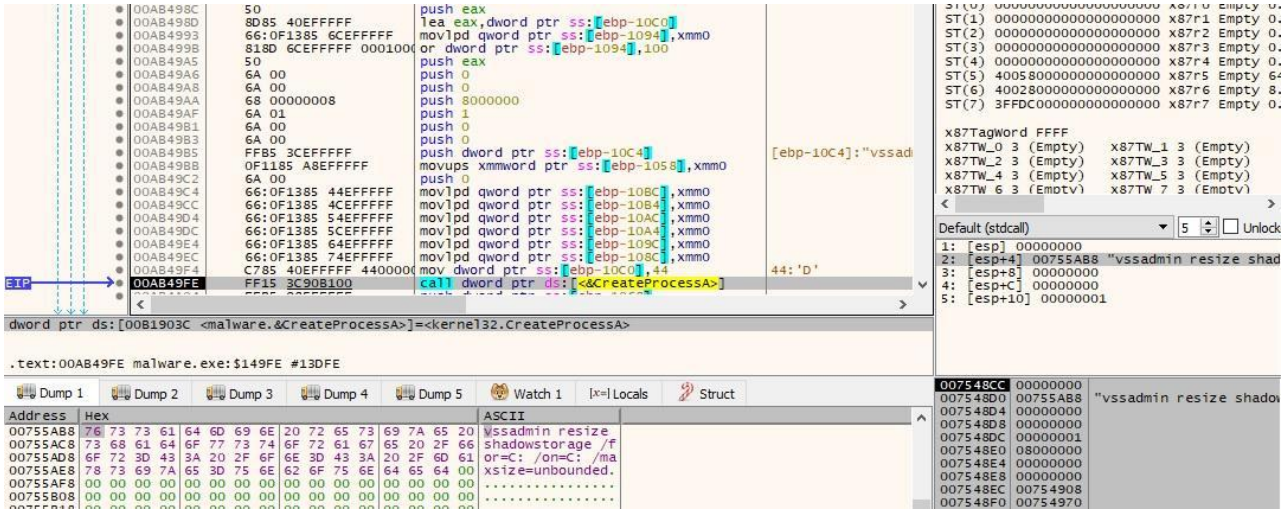


Figure 21

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

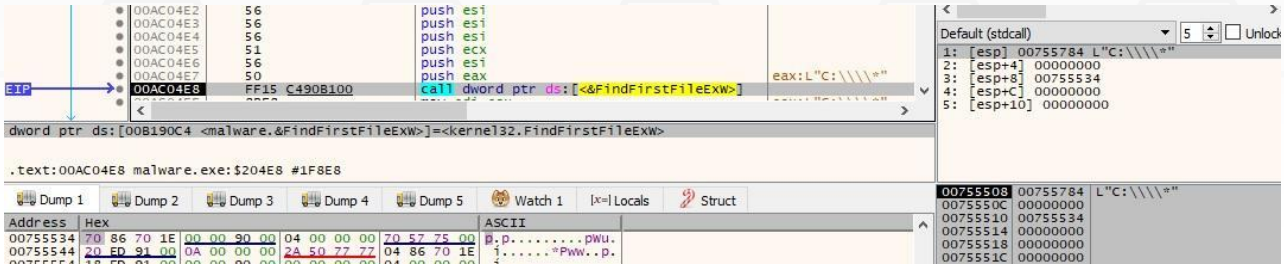


Figure 22

The following files and directories will be skipped from encryption:

```
.rdata:00B27B88 aGSkpDir:                               ; DATA XREF: sub_AB14A0+817↑o
.rdata:00B27B88                                         ; sub_AB14A0+83E↑o
.rdata:00B27B88                 text "UTF-16LE", 'g_skp_dir',0
.rdata:00B27B9C aDesktopIni:                            ; DATA XREF: sub_AB14A0+97F↑o
.rdata:00B27B9C                 text "UTF-16LE", 'desktop.ini',0
.rdata:00B27BB4 aThumbsDb:                              ; DATA XREF: sub_AB14A0+995↑o
.rdata:00B27BB4                 text "UTF-16LE", 'Thumbs.db',0
.rdata:00B27BC8 aWindows:                               ; DATA XREF: sub_AB27F0+294↑o
.rdata:00B27BC8                 text "UTF-16LE", 'Windows',0
.rdata:00B27BD8 aWindowsOld:                            ; DATA XREF: sub_AB27F0+456↑o
.rdata:00B27BD8                 text "UTF-16LE", 'Windows.old',0
.rdata:00B27BF0 aPerflogs:                              ; DATA XREF: sub_AB27F0+626↑o
.rdata:00B27BF0                 text "UTF-16LE", 'PerfLogs',0
.rdata:00B27C02                 align 4
.rdata:00B27C04 aMsocache:                              ; DATA XREF: sub_AB27F0+7F6↑o
.rdata:00B27C04                 text "UTF-16LE", 'MSOCache',0
.rdata:00B27C16                 align 4
.rdata:00B27C18 aProgramFiles:                          ; DATA XREF: sub_AB27F0+9EE↑o
.rdata:00B27C18                 text "UTF-16LE", 'Program Files',0
.rdata:00B27C34 aProgramFilesX8:                        ; DATA XREF: sub_AB27F0+BBD↑o
.rdata:00B27C34                 text "UTF-16LE", 'Program Files (x86)',0
.rdata:00B27C5C aProgramdata:                           ; DATA XREF: sub_AB27F0+D8D↑o
.rdata:00B27C5C                 text "UTF-16LE", 'ProgramData',0
```

Figure 23

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

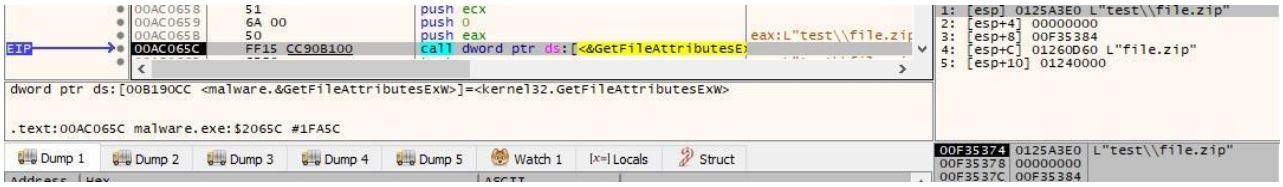


Figure 24

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

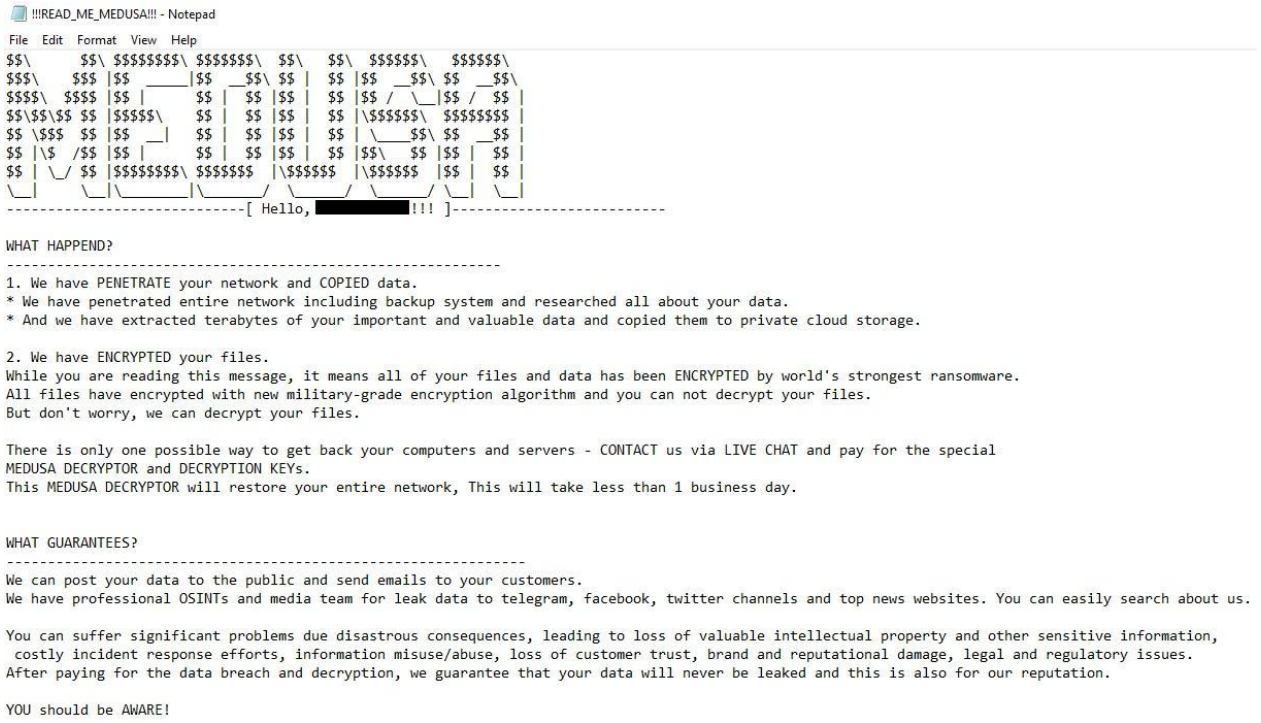


Figure 26

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**):

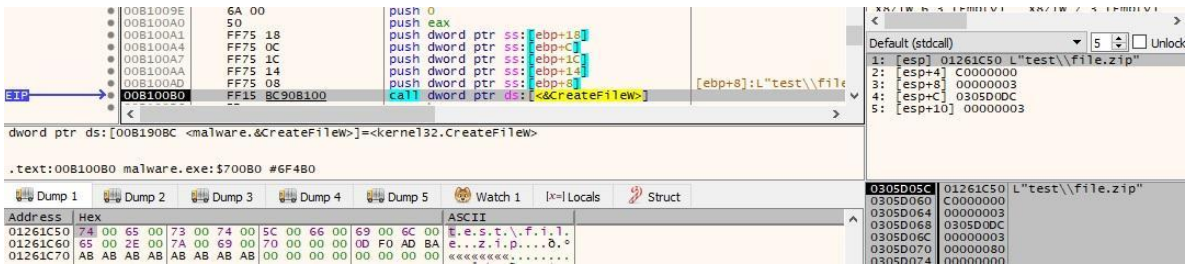


Figure 27

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

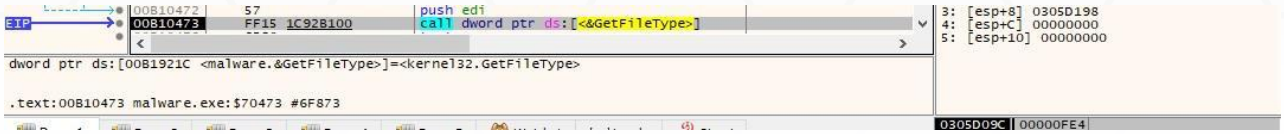


Figure 28

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

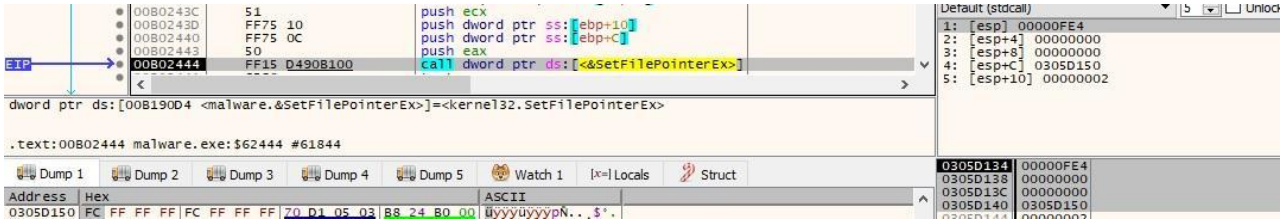


Figure 29

Each file is read by calling the ReadFile function:

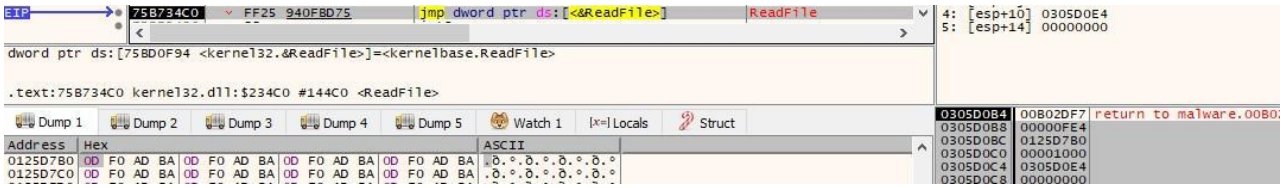


Figure 30

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

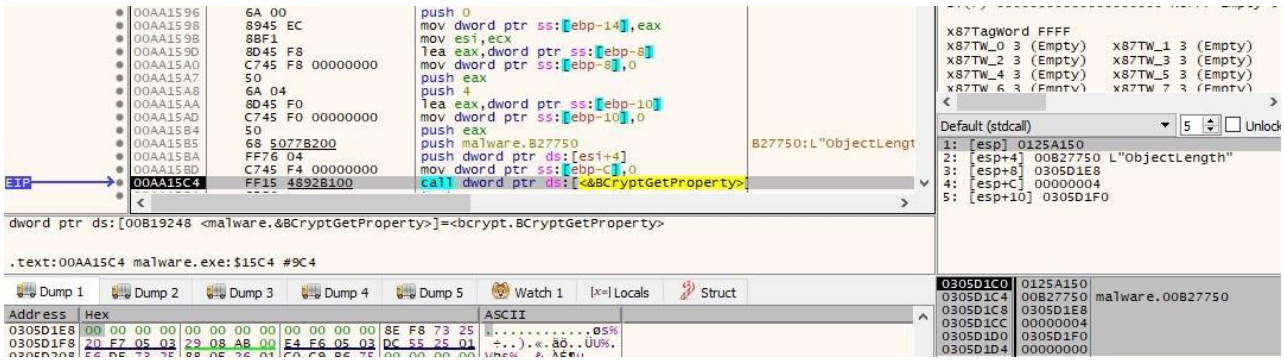


Figure 31

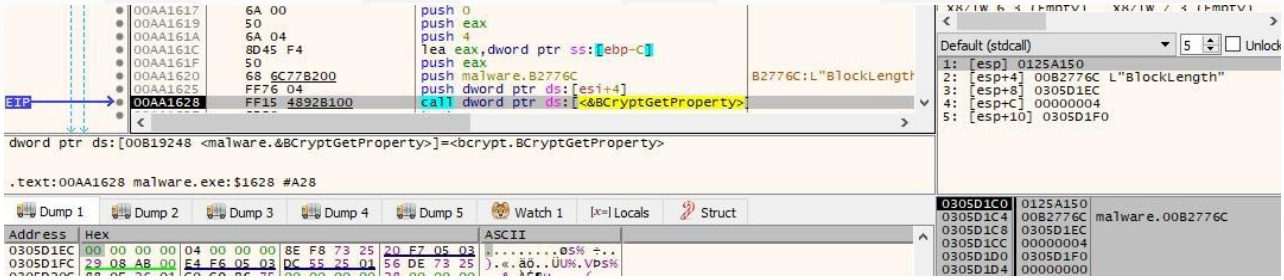


Figure 32

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

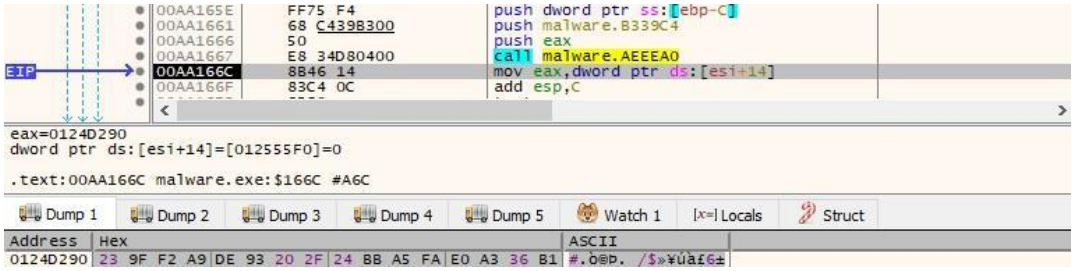


Figure 33

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

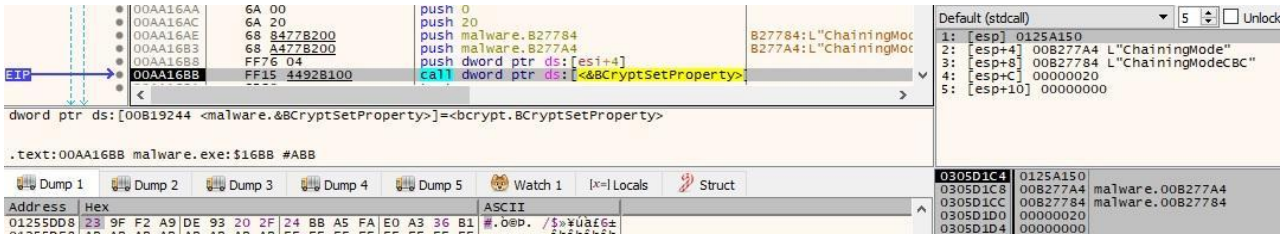


Figure 34

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

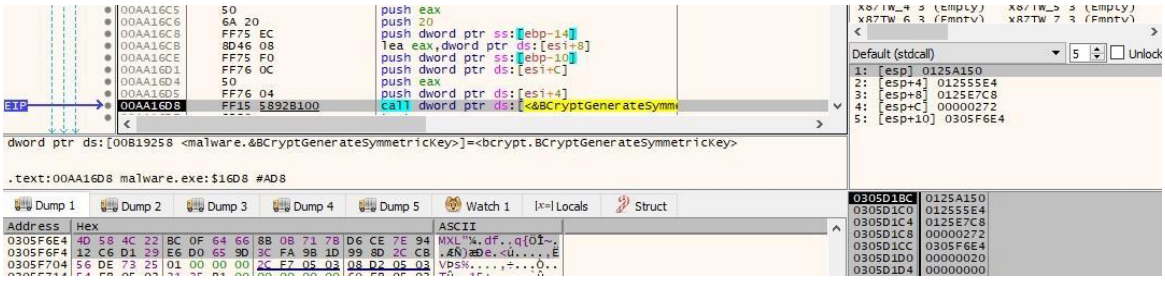
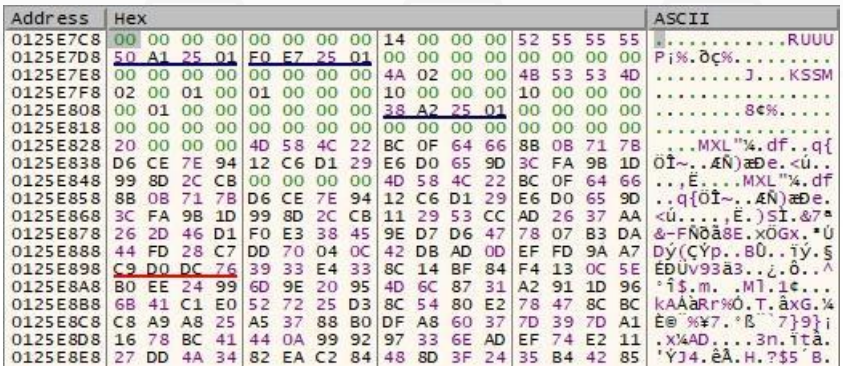


Figure 35



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	P¡%.ðç%.....
0125E7E8	00 00 00 00 00 00 00 00 4A 02 00 00 4B 53 53 4D	....J...KSSM
0125E7F8	02 00 01 00 01 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	....8¢%.....
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 8B 0B 71 7B	...MXL"¼.df..q{
0125E838	D6 CE 7E 94 12 C6 D1 29 E6 D0 65 9D 3C FA 9B 1D	ÖÎ~..ÆÑ)æÐe.<ú..
0125E848	99 8D 2C CB 00 00 00 00 4D 58 4C 22 BC 0F 64 66	..,Ë....MXL"¼.df
0125E858	8B 0B 71 7B D6 CE 7E 94 12 C6 D1 29 E6 D0 65 9D	..q{ÖÎ~..ÆÑ)æÐe.
0125E868	3C FA 9B 1D 99 8D 2C CB 11 29 53 CC AD 26 37 AA	<ú....,Ë.)SÌ.&7ª
0125E878	26 2D 46 D1 F0 E3 38 45 9E D7 D6 47 78 07 B3 DA	&-FÑðã8E.×ÖGx.³Ú
0125E888	44 FD 28 C7 DD 70 04 0C 42 DB AD 0D EF FD 9A A7	Dý(ÇÝp..BÛ..ïý.§
0125E898	C9 D0 DC 76 39 33 E4 33 8C 14 BF 84 F4 13 0C 5E	ÉÐÜv93ä3..¿.ô..^
0125E8A8	B0 EE 24 99 6D 9E 20 95 4D 6C 87 31 A2 91 1D 96	°î$.m. .Ml.1¢...
0125E8B8	6B 41 C1 E0 52 72 25 D3 8C 54 80 E2 78 47 8C BC	kAÁàRr%Ó.T.âxG.¼
0125E8C8	C8 A9 A8 25 A5 37 88 B0 DF A8 60 37 7D 39 7D A1	È©¨%¥7.°ß¨`7}9}¡
0125E8D8	16 78 BC 41 44 0A 99 92 97 33 6E AD EF 74 E2 11	.x¼AD....3n.ïtâ.
0125E8E8	27 DD 4A 34 82 EA C2 84 48 8D 3F 24 35 B4 42 85	'ÝJ4.êÂ.H.?$5´B.

Figure 36

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

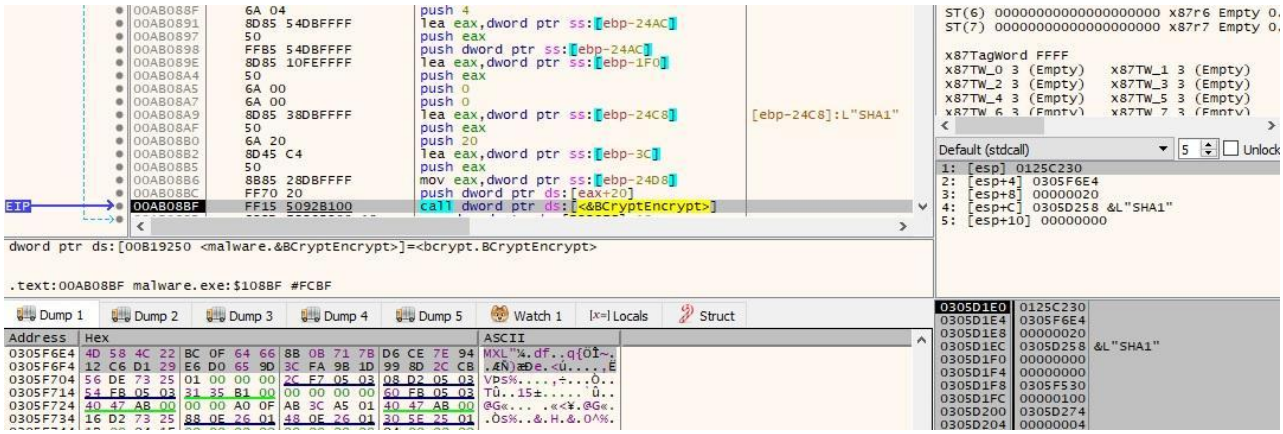


Figure 37

Address	Hex	ASCII
0305F530	C4 C4 0E 59 06 12 C0 AE 65 16 95 60 65 09 7F 8C	ÄÄ.Y..À®e..`e...
0305F540	1F 00 45 B2 6D E7 2C 59 BE 18 49 56 4E BC E6 05	..E²mç,Y¾.IVN¼æ.
0305F550	5A 61 1A 12 BA 48 56 53 3B 71 6C 33 1E FD A2 0D	Za..ºHVS;ql3.ý¢.
0305F560	45 6C 42 64 61 7B 98 C6 37 CA EF 54 07 2B 59 C5	ElBda{.Æ7ÊïT.+YÅ
0305F570	F1 9A 4E 3D 52 E8 5A B9 09 16 4E 8C 31 2B 84 67	ñ.N=RèZ'..N.1+.g
0305F580	96 CB CA A7 9E F7 0C E2 5D 5A BE 90 C6 71 B9 2B	.ËÊ§.÷.â]Z¾.Æq'+
0305F590	01 DF 8D 46 11 AE E9 6C 5A 77 56 90 43 B3 5F 4B	.ß.F.®élZwV.C³_K
0305F5A0	BB C9 8C D8 63 30 94 8E BA B3 2A EB 0C F5 42 CA	»É.Øc0..ºª*è.õBÊ
0305F5B0	CC 98 CA 86 BA BE DD ED 31 38 15 08 3B 33 7D 03	Ì.Ê.º¾Ýí18..;3}.
0305F5C0	09 EC F1 87 2F 20 C3 BD 4B 88 F4 A7 82 7C 34 EE	.ìñ./ Ã½K.ô§.|4î
0305F5D0	AF A6 CB 85 D3 FC D3 D8 50 86 2C 2E F3 CE C5 F2	¯¦Ë.ÓüÓØP.,.óÎÅò
0305F5E0	22 A6 87 F8 CD B1 6A 69 70 FC E9 93 BC 87 A0 C6	"¦.Íí±jipüé.¼. Æ
0305F5F0	1E 0D 17 D2 B8 D7 E0 BA 6C DC 3C 34 0E 41 B6 DA	...Ò¸×àºlÜ<4.A¶Ú
0305F600	54 B3 A7 1A 95 9D 5A 13 2B 1D 40 23 D9 27 E2 8B	T³§...Z.+.@#Ù'â.
0305F610	8A B0 09 2A AA 2A C7 E6 9D 75 C9 90 69 96 41 E6	.°.*ª*Çæ.uÉ.i.Aæ
0305F620	92 AF 32 04 67 B7 68 7A 80 43 E0 20 7D A7 30 02	.¯2.g·hz.Cà }§0.

Figure 38

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

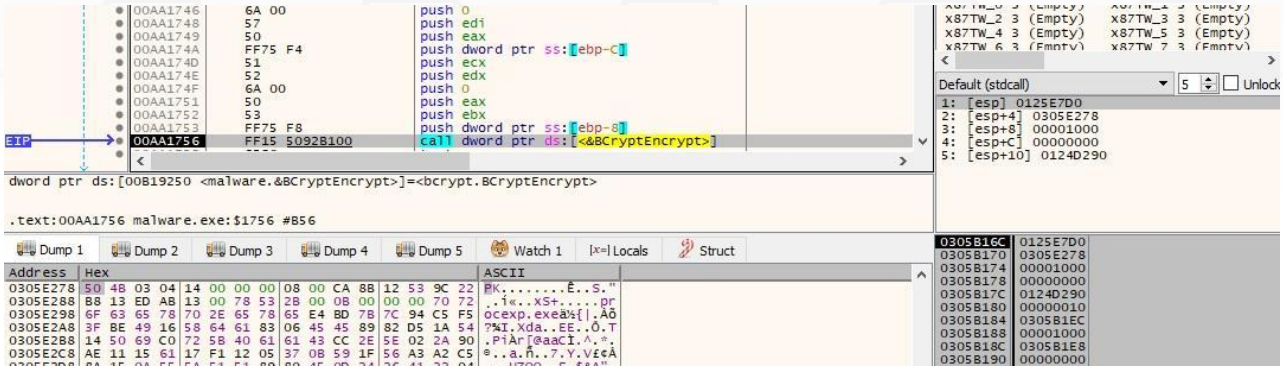


Figure 39

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

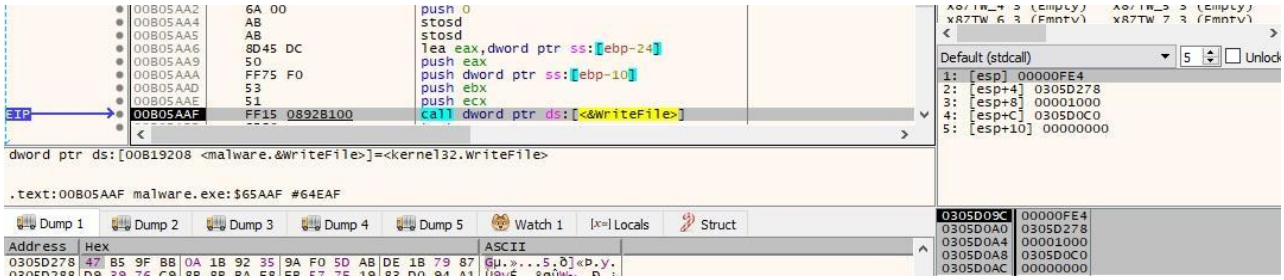


Figure 40

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

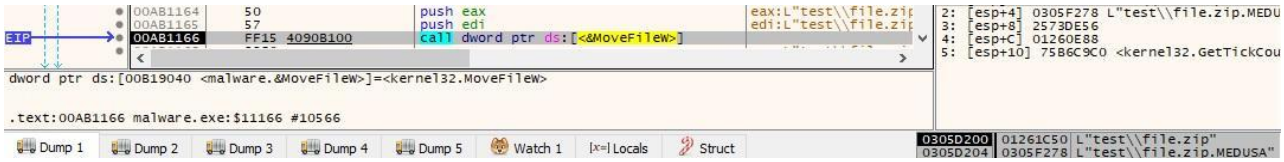


Figure 41

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

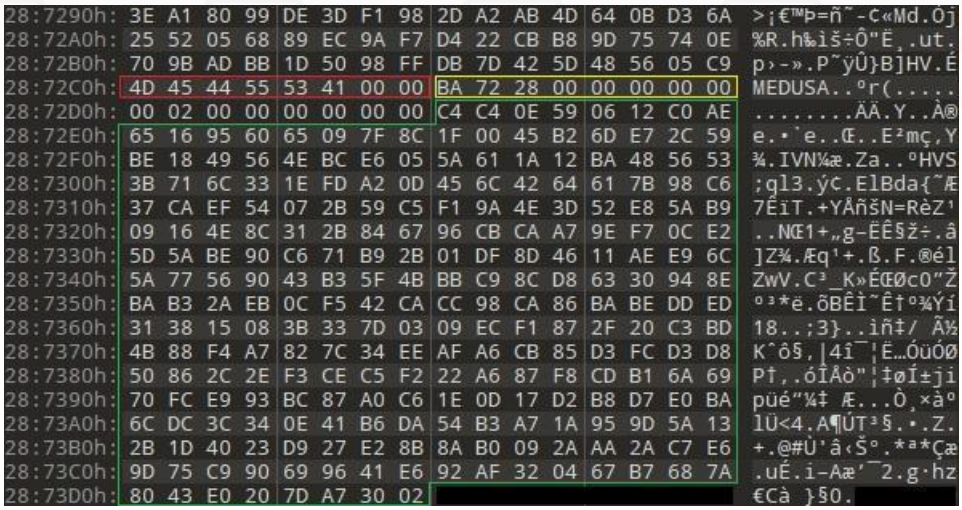


Figure 42

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

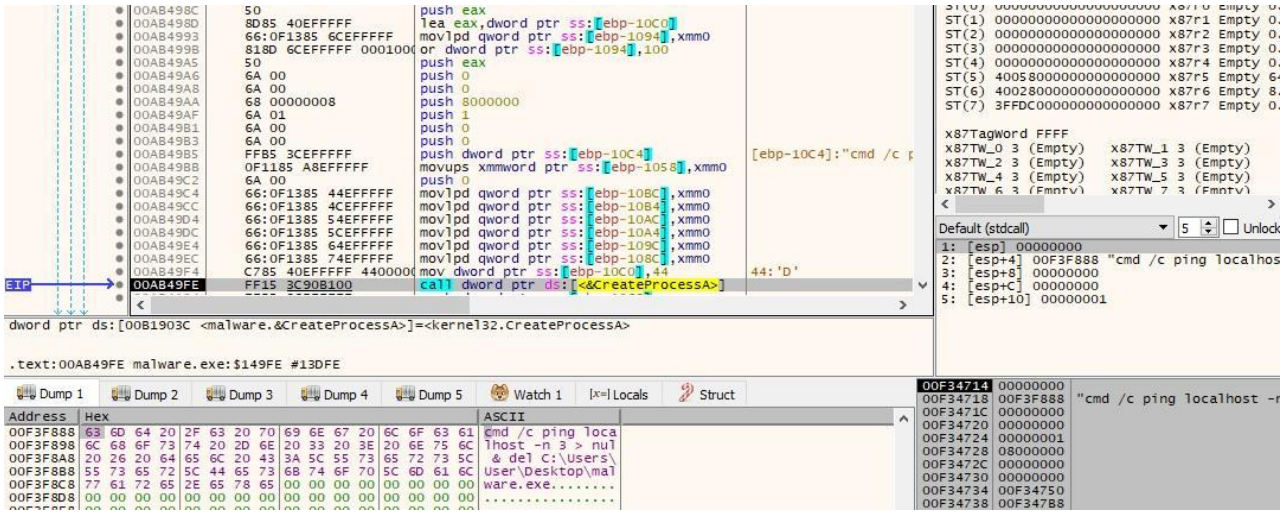


Figure 43

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

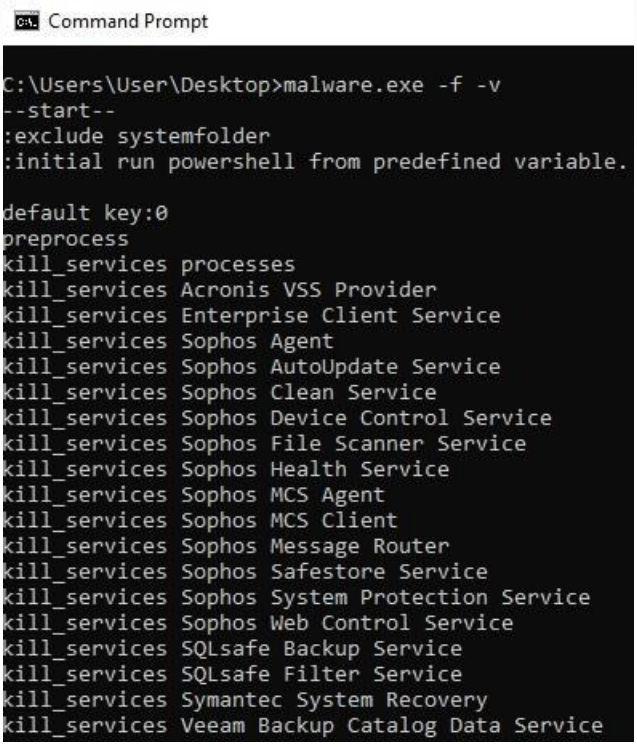


Figure 44

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:

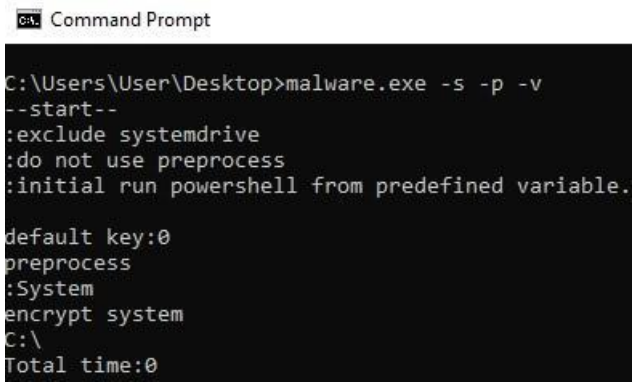


Figure 45

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

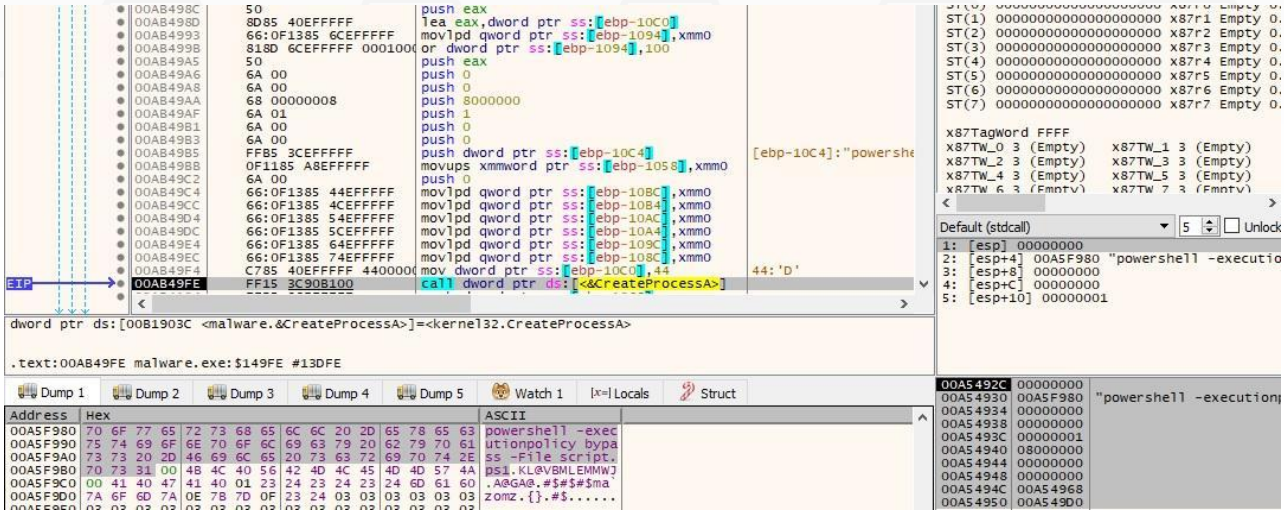


Figure 46

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

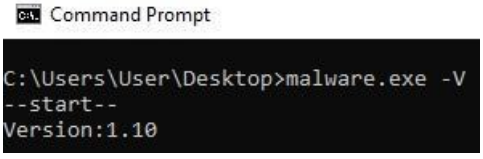


Figure 47

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" "EPUpdateService" "EraserSvc11710" "EsgShKernel" "FA_Scheduler" "IISAdmin" "IMAP4Svc" "macmnsvc" "masvc" "MBAMService" "MBEndpointAgent" "McAfeeEngineService" "McAfeeFramework" "McAfeeFrameworkMcAfeeFramework" "McShield" "McTaskManager" "mfemms" "mfevtp" "MMS" "mozyprobackup" "MsDtsServer" "MsDtsServer100" "MsDtsServer110" "MSExchangeES" "MSExchangeIS" "MSExchangeMGMT" "MSExchangeMTA" "MSExchangeSA" "MSExchangeSRS" "MSOLAP$SQL_2008" "MSOLAP$SYSTEM_BGC" "MSOLAP$TPS" "MSOLAP$TPSAMA" "MSSQL$BKUPEXEC" "MSSQL$ECWDB2" "MSSQL$PRACTICEMGT" "MSSQL$PRACTTICEBGC" "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" "sophossps" "SQLAgent$BKUPEXEC" "SQLAgent$ECWDB2" "SQLAgent$PRACTTICEBGC" "SQLAgent$PRACTTICEMGT" "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" "kavfsslp" "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" "xfssvccon.exe" "tmlisten.exe" "PccNTMon.exe" "CNTAoSMgr.exe" "Ntrtscan.exe" "mbamtray.exe"