A Deep Dive into Medusa Ransomware

Prepared by: Vlad Pasca, Senior Malware & Threat Analyst
# Table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of contents</td>
<td>1</td>
</tr>
<tr>
<td>Executive summary</td>
<td>2</td>
</tr>
<tr>
<td>Analysis and findings</td>
<td>2</td>
</tr>
<tr>
<td>Indicators of Compromise</td>
<td>18</td>
</tr>
<tr>
<td>Appendix</td>
<td>19</td>
</tr>
</tbody>
</table>
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](image-url)

When running with the “-v” parameter, the malware displays multiple strings in the command line window using WriteFile, as shown in Figure 2.
The malicious process creates multiple anonymous pipes via a function call to CreatePipe:

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

The malware creates a PowerShell process using the CreateProcessA API (0x08000000 = CREATE_NO_WINDOW):
The malicious process reads data from the pipe containing the above process output using `ReadFile`:

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

CryptStringToBinaryA is used to decode the RSA public key from Base64 (0x7 = `CRYPT_STRING_ANY`).
Medusa ransomware decodes a structure of the \texttt{RSA\_CSP\_PUBLICKEYBLOB} type by calling the 
\texttt{CryptDecodeObjectEx} function \( \text{0x10001} = \text{X509\_ASN\_ENCODING | PKCS\_7\_ASN\_ENCODING} \), 
\( \text{0x13} = \text{RSA\_CSP\_PUBLICKEYBLOB} \):

The process imports the RSA public key from a key BLOB using \texttt{BCryptImportKeyPair}:
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:
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](image13.png)

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

![Figure 14](image14.png)

![Figure 15](image15.png)
The process deletes the Volume Shadow Copies using the vssadmin command, as highlighted below:

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

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

It extracts the amount of space that is available on the disk volume using the GetDiskFreeSpaceExW routine, as shown below:
The sample spawns two processes in order to resize the maximum amount of storage space used for shadow copy storage:

The ransomware enumerates the files on the drives using the FindFirstFileExW and FindNextFileW APIs:
The following files and directories will be skipped from encryption:

```
.text:00278688 aSkipDir:                ; DATA XREF: sub_AB14A8+817f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aDesktopIni:            ; DATA XREF: sub_AB14A8+97f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aThumbsDb:              ; DATA XREF: sub_AB14A8+995f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aWindows:               ; DATA XREF: sub_AB27F0+294f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aWindowsOld:             ; DATA XREF: sub_AB27F0+458f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aPerflogs:              ; DATA XREF: sub_AB27F0+626f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aSocache:               ; DATA XREF: sub_AB27F0+7f6f0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aProgramFiles:           ; DATA XREF: sub_AB27F0+9eEf0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aProgramFilesX8:         ; DATA XREF: sub_AB27F0+88Df0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
.text:00278688 aProgramData:            ; DATA XREF: sub_AB27F0+80Df0
..rdata:00278688         .rdata:00278688   .data:00278688   .data:00278688
```

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

```
Figure 23
```

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

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

The malware creates a key object based on 32 bytes that were generated, which represent the AES256 key that is changing between iterations:
Firstly, the AES key is encrypted using the RSA public key via a call to BCryptEncrypt: 

![Image](image1.png)

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

![Image](image2.png)
The encrypted data is written back to the file by calling the WriteFile function:

The ransomware appends the “.MEDUSA” extension to all encrypted files (see 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).
Finally, if the "-d" parameter is not specified, the malware deletes itself:

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

The ransomware can execute a PowerShell script using the “-w” parameter, as highlighted in the figure below.
The last parameter, "-V," displays the Medusa ransomware version:

```
> Command Prompt
C:\Users\User\Desktop>malware.exe -V
--start--
Version:1.10
```

**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" "ArcSchedVsc" "Antivirus" "ARSM" "BackupExecAgentAccelerator"
"BackupExecAgentBrowser" "BackupExecDeviceMediaService" "BackupExecJobEngine"
"BackupExecManagementService" "BackupExecRPCService" "BackupExecVSSProvider" "bedbg" "DCAgent"
"EPSecurityService" "EPUtdateService" "EraseSvcs11710" "EsgShKernel" "FA_Scheduler" "IISAdmin" "I MAP4Svc"
"macmnsvc" "msavc" "MBAMService" "MBEndpontAgent" "McAfeeEngineService" "McAfeeFramework"
"McAfeeFramework" "McShield" "MCTaskManager" "mfmems" "mfvtap" "MMS" "mozybackup"
"MsDsServer" "MsDsServer100" "MsDsServer110" "MSExchangeES" "MSExchangeIG" "MSExchangeMGMT"
"MSExchangeMTA" "MSExchangeSA" "MSExchangeSRS" "M SOLAP$SQL_2008" "M SOLAP$SYSTEM_BGC" "M SOLAP$TPS"
"MSOLAP$TPSAMA" "MSSQL$BUKUPEXEC" "MSSQL$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$SBSMONITORING"
"MSSQLFDLauncherSHAREPOINT" "MSSQLFDLauncherSQL_2008" "MSSQLFDLauncherSYSTEM_BGC"
"MSSQLFDLauncherTPS" "MSSQLFDLauncherTPSAMA" "MSSQLSERVER" "MSSQLServerADHelper100"
"MSSQLServerOLAPService" "MySQL80" "MySQL57" "ntrtscn" "OracleClientCache80" "PDVFService" "POP3Svc"
"RESvc" "sacsvr" "SamSs" "SADMinService" "SAService" "SDRSVC" "-SepMasterService" "ShMonitor" "SmsInst"
"SmcService" "SMTSvc" "SNAC" "SntpService" "sophossps" "SQLAgent$BUKUPEXEC" "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$ECWDB2" "SQLWriter" "SstpSvc"
"svrcGenricHost" "swl_filter" "swl_service" "swl_update_64" "TmCCSF" "tmList" "TrueKey" "TrueKeyScheduler"
"TrueKeyServiceHelper" "UIODetect" "VeeamBackUpSvc" "VeeamBrokerSvc" "VeeamCatalogSvc" "VeeamCloudService"
"VeeamDeploymentService" "VeeamDeploySvc" "VeeamDeploymentManagerSvc" "VeeamMountSvc" "VeeamNFSSvc"
"VeeamRESTSvc" "VeeamTransportSvc" "W3Svc" "wbengine" "WR5VC" "MSSQL$VEEAMSQL2008R2"
"VeeamSQLServerIntegrationSvc" "swl_update" "SQLAgent$CDB"
"SQLAgent$CITRIX_METAFRAME" "SQL Backups" "MSSQL$PROD" "Zool 2 Service" "MSSQLServerADHelper"
"SQLAgent$PROD" "msfrels$PROD" "NetMsqActivator" "EhtftpSvc" "ekrn" "EHASHRVR" "MSSQL$SOPHOS"
"SQLAgent$SOPHOS" "AVP" "kinagent" "MSSQL$SQLEXPRESS" "SQLAgent$SQLEXPRESS" "wbengine" "kavfsslclp"
"KAFVSGT" "KAVFS" "mfefire"

List of processes

"zoolz.exe" "agntsvcs.exe" "dbheng50.exe" "dbnsmpe.exe" "encsvc.exe" "excel.exe" "firefoxconfig.exe" "infophath.exe"
"isqlplussvc.exe" "msaccess.exe" "msftesql.exe" "mspub.exe" "mydesktoopqos.exe" "mydesktopservice.exe" "mysql.exe"
"mysql-dl-nt.exe" "mysqld-opt.exe" "ocaoutoups.exe" "occom.r.exe" "ocssd.exe" "onnote.exe" "oracle.exe" "outlook.exe"
"powerpnt.exe" "sqbcorexservice.exe" "sqlagent.exe" "sqlbrowser.exe" "sqlservr.exe" "sqlwriter.exe" "steam.exe"
"syncm.exe" "tcpbconfexe.exe" "theba.exe" "theba56.exe" "thunderbird.exe" "visio.exe" "winword.exe" "wordpad.exe"
"xfssvccon.exe" "tmlisten.exe" "PccNTMon.exe" "CNTAoSMgr.exe" "Ntrtscn.exe" "mbamtray.exe"