

A Deep Dive Into Black Basta Ransomware

Prepared by: Vlad Pasca, Senior Malware &
Threat Analyst



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

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

Table of contents

Executive summary	2
Analysis and findings	2
Thread activity – sub_F33DA0 function	12
Case 1 – File size < 704 bytes	17
Case 2 – File size < 4KB	18
Case 3 – File size > 4KB	19
Indicators of Compromise	21

Executive summary

Black Basta ransomware is a recent threat that compiled its first malware samples in February 2022. The ransomware deletes all Volume Shadow Copies, creates a new JPG image set as the Desktop Wallpaper and an ICO file representing the encrypted files. Unlike other ransomware families, the malware doesn't skip files based on their extensions. However, it doesn't encrypt critical folders that would make the system inoperable.

The files are encrypted using the ChaCha20 algorithm, with the key and nonce being encrypted using the RSA public key that is hard-coded in the sample. The malware can fully or partially encrypt a file depending on its size. The extension of the encrypted files is changed to .basta by the ransomware.

Analysis and findings

SHA256: ae7c868713e1d02b4db60128c651eb1e3f6a33c02544cc4cb57c3aa6c6581b6e

The process displays "ENCRYPTION" in the program window using WriteFile:

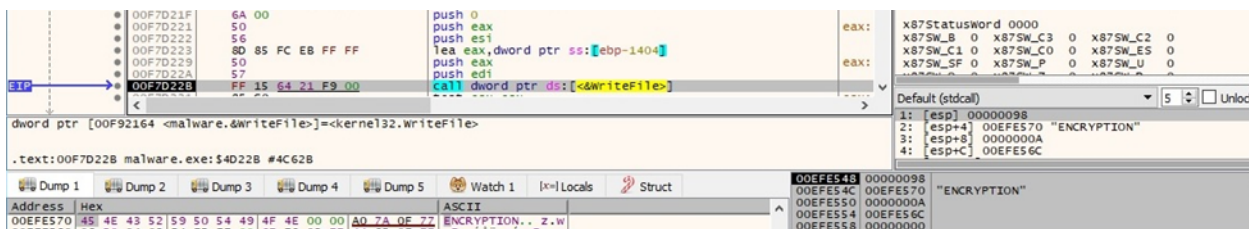


Figure 1



Figure 2

The binary retrieves the process ID via a function call to GetCurrentProcessId:

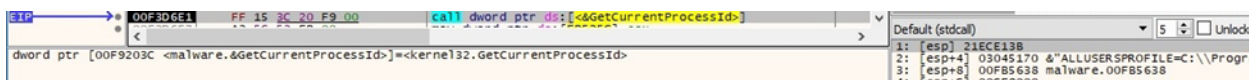


Figure 3

The malicious process detaches itself from its console by calling the FreeConsole API:

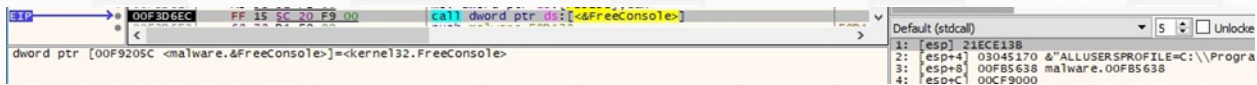


Figure 4

The executable obtains the "COMSPEC" environment variable value, which points to the command line:

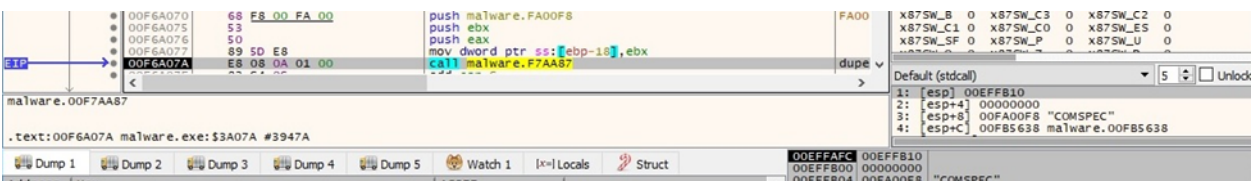


Figure 5

The ransomware deletes all Volume Shadow Copies by running the "C:\Windows\SysNative\vssadmin.exe delete shadows /all /quiet" command, as highlighted below:

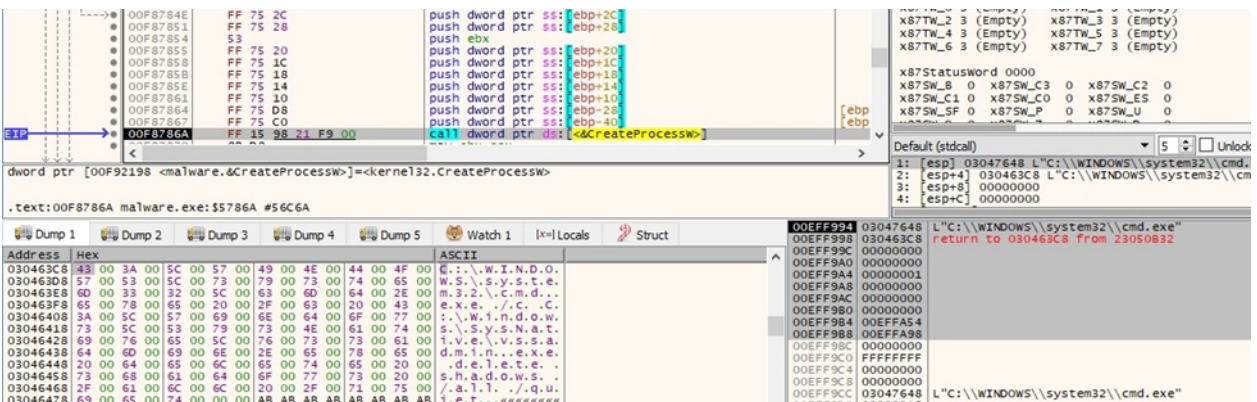


Figure 6

The sample waits until the spawned process finishes using the WaitForSingleObject routine:

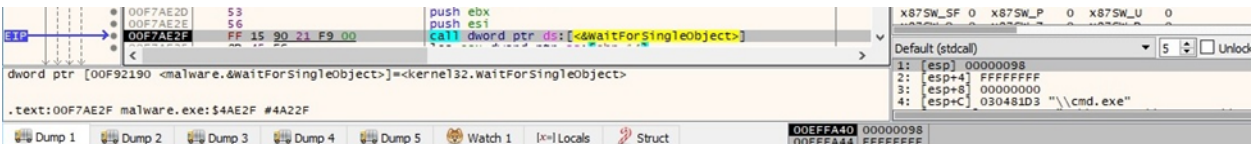


Figure 7

A similar process as above that deletes the Volume Shadow Copies is spawned:

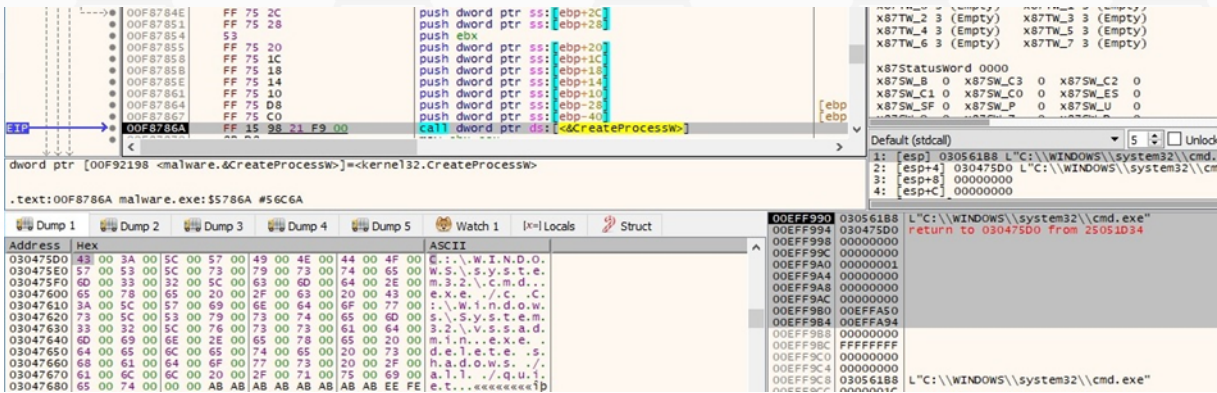


Figure 8

The binary extracts the path of the executable of the current process via a call to GetModuleFileNameW:

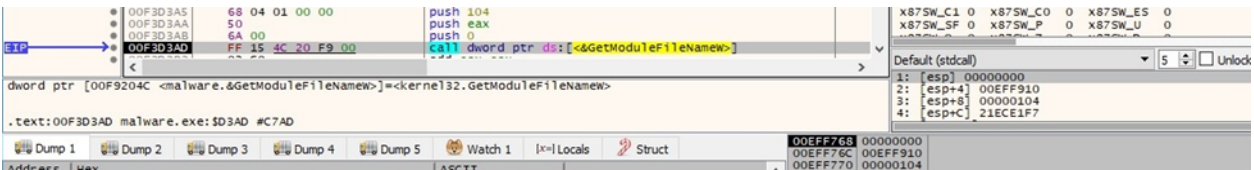


Figure 9

The GetTempPathW API is utilized to retrieve the path of the Temp directory:



Figure 10

A file called "dlaksjdoiwq.jpg" is created in the Temp directory (0x40 = **_SH_DENYNO**):

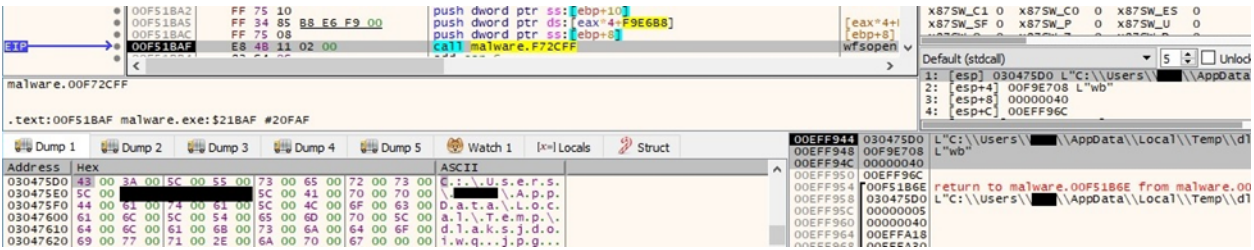


Figure 11

The process moves the file position indicator to the beginning of the file using the fsetpos function:

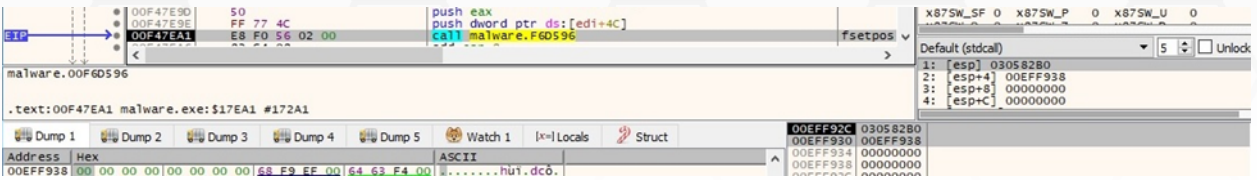


Figure 12

The WriteFile routine is used to populate the JPG file, which contains instructions from the threat actor:

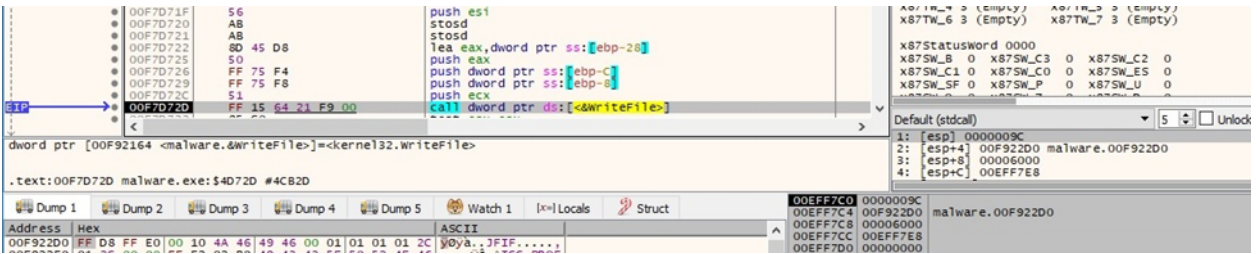


Figure 13



Figure 14

The newly created image is set as the Desktop Wallpaper using SystemParametersInfoW (0x14 = SPI_SETDESKWALLPAPER, 0x1 = SPIF_UPDATEINIFILE):

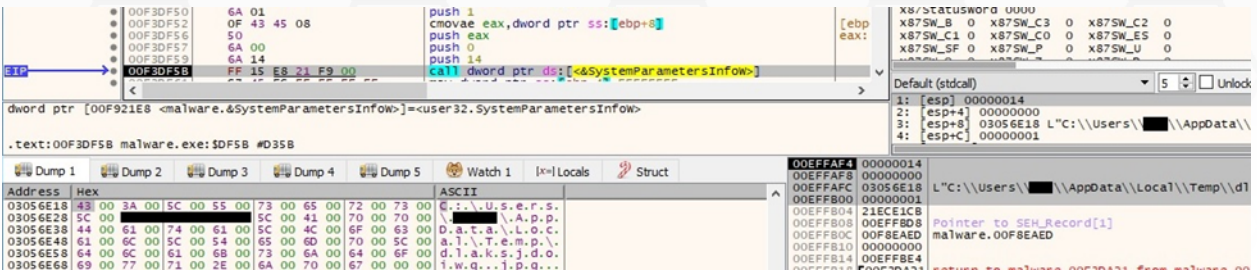


Figure 15

The executable creates an ICO file called "fkdsadasd.ico" in the Temp directory:

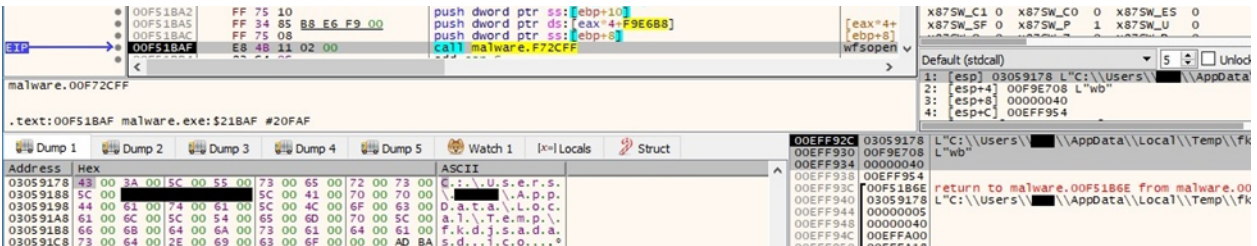


Figure 16

The ransomware writes content to the ICO file, which will represent the icon of the encrypted files:



Figure 17

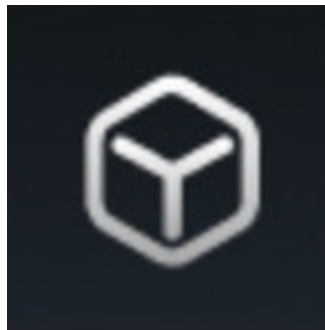


Figure 18

Black Basta ransomware creates the ".basta\DefaultIcon" registry key using RegCreateKeyExW

(0x80000000 = HKEY_CLASSES_ROOT, 0x103 = KEY_WOW64_64KEY | KEY_SET_VALUE | KEY_QUERY_VALUE):

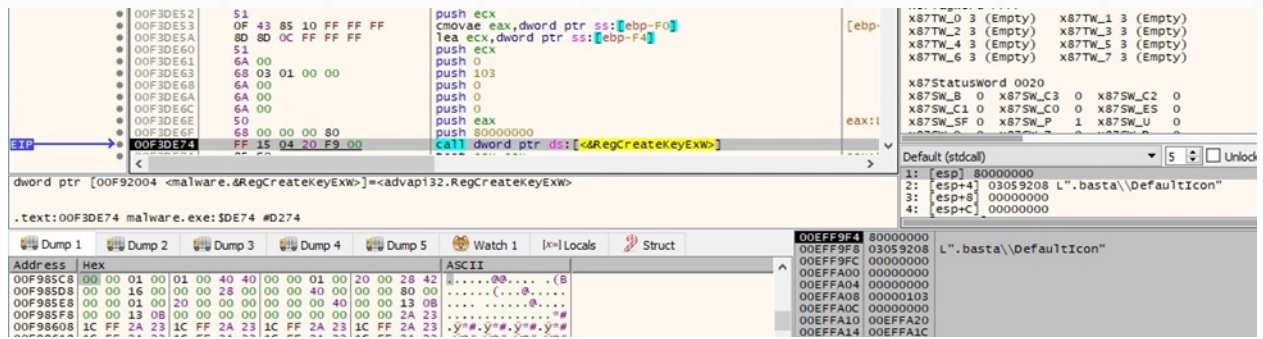


Figure 19

The “(Default)” value of the above key is set to the path of the ICO file:

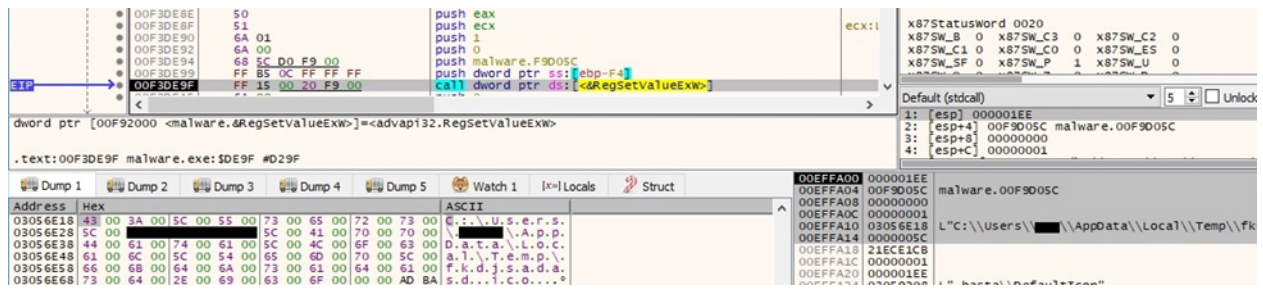


Figure 20



Figure 21

The malicious binary notifies the system that the icon has been changed by calling the SHChangeNotify function (0x08000000 = SHCNE_ASSOCCHANGED, 0x3000 = SHCNF_FLUSHNOWAIT):

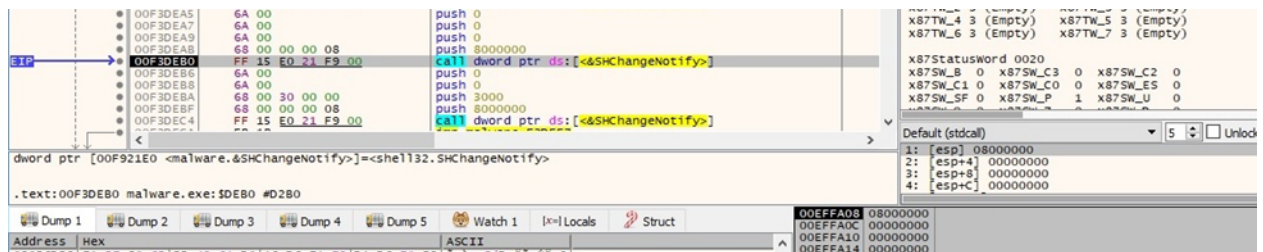


Figure 22

The malware starts scanning for volumes on the system using FindFirstVolumeW:

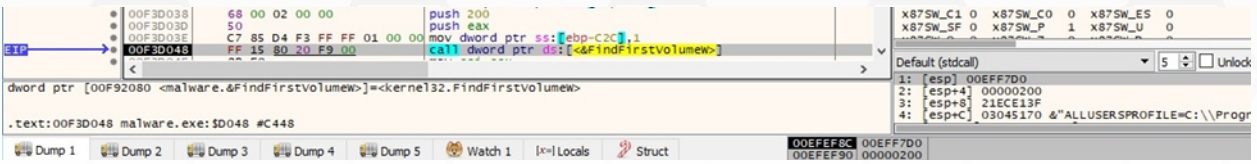


Figure 23

GetVolumePathNamesForVolumeNameW is utilized to obtain the list of drive letters and mounted folder paths for the volume:

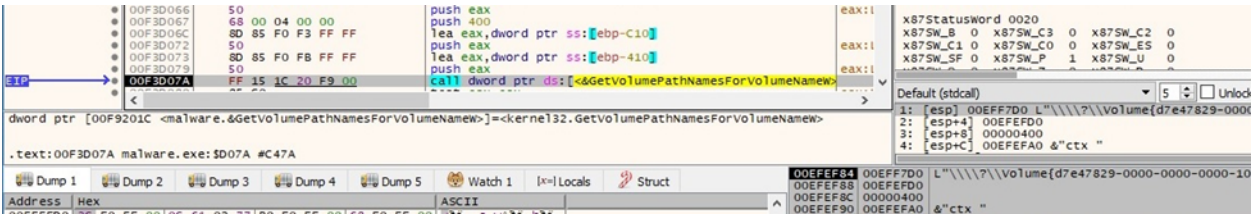


Figure 24

For each drive found, the process performs a call to the GetVolumeInformationW API (see figure 25). As opposed to other ransomware families, Black Basta only targets the mounted volumes and doesn't mount the hidden volumes.

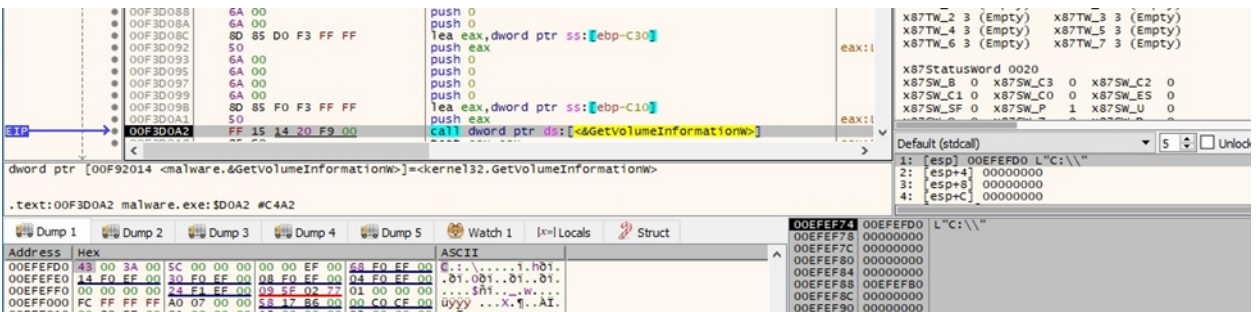


Figure 25

The volume's enumeration continues by calling the FindNextVolumeW routine:

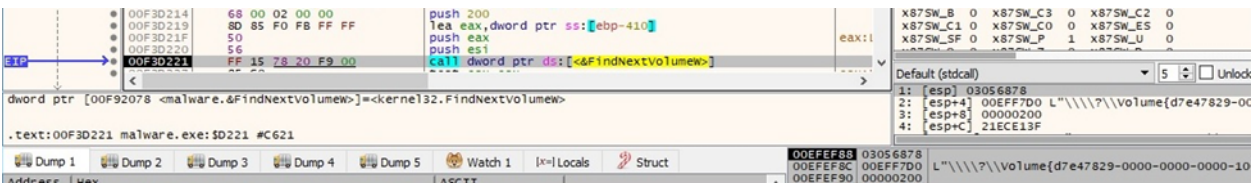


Figure 26

The ransomware extracts a standard set of attribute information from the drives found via a function call to GetFileAttributesExW (0x0 = **GetFileExInfoStandard**):

The malware creates multiple threads that will handle the file encryption. The function responsible for encryption is sub_F33DA0 and not the starting address of the thread:

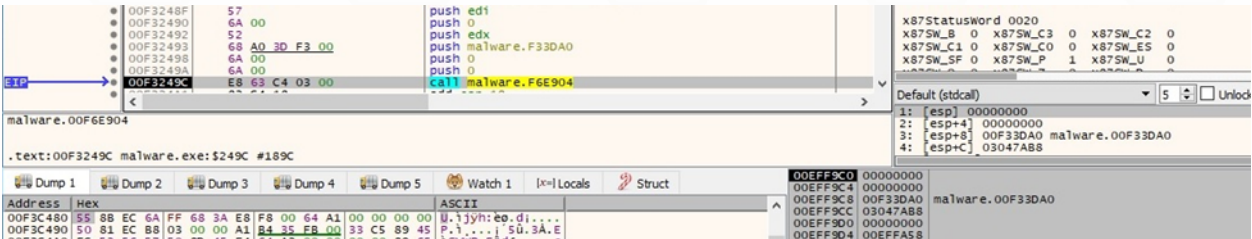


Figure 32

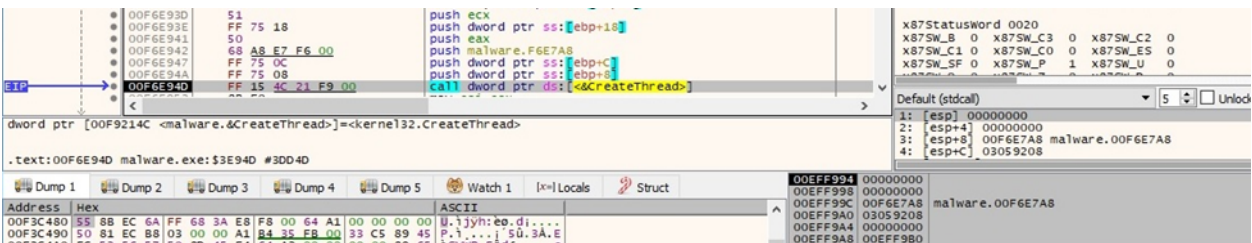


Figure 33

The malicious process starts enumerating the files on the drive using FindFirstFileW:

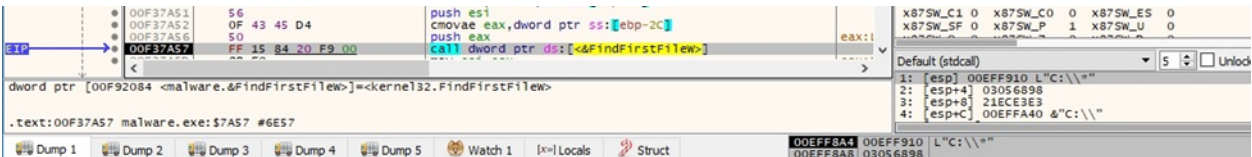


Figure 34

As shown in figure 35, the following files/directories will be skipped:

- \$Recycle.Bin
- Windows
- boot
- readme.txt
- dlaksjdoiwq.jpg
- NTUSER.DAT
- fkdjsadasd.ico

```

.rdata:00F9D0A8          text "UTF-16LE", '$Recycle.Bin',0
.rdata:00F9D0C2          align 4
.rdata:00F9D0C4          aWindows:                ; DATA XREF: sub_F38BE0:loc_F3BF44f0
.rdata:00F9D0C4          text "UTF-16LE", 'Windows',0
.rdata:00F9D0D4          aBoot:                    ; DATA XREF: sub_F38BE0:loc_F3BF8Bf0
.rdata:00F9D0D4          text "UTF-16LE", 'boot',0
.rdata:00F9D0DE          align 10h
.rdata:00F9D0E0          aReadmeTxt:              ; DATA XREF: sub_F38B00+3Cf0
.rdata:00F9D0E0          ; sub_F38BE0:loc_F38FD5f0
.rdata:00F9D0E0          text "UTF-16LE", 'readme.txt',0
.rdata:00F9D0F6          align 4
.rdata:00F9D0F8          aDlaksjdoiwqjpg:        ; DATA XREF: sub_F38BE0:loc_F3C01Ff0
.rdata:00F9D0F8          ; sub_F3DCA0+5Cf0
.rdata:00F9D0F8          text "UTF-16LE", 'dlaksjdoiwq.jpg',0
.rdata:00F9D118          aNtuserDat:              ; DATA XREF: sub_F38BE0:loc_F3C069f0
.rdata:00F9D118          text "UTF-16LE", 'NTUSER.DAT',0
.rdata:00F9D12E          align 10h
.rdata:00F9D130          aError755                db 'Error 755: ',0
.rdata:00F9D130          ; DATA XREF: sub_F38BE0:loc_F3C29Df0
.rdata:00F9D13C          aFkdjsadasdIco:         ; DATA XREF: sub_F3DB50+5Cf0
.rdata:00F9D13C          text "UTF-16LE", 'fkdjsadasd.ico',0
.rdata:00F9D15A          align 4

```

Figure 35

The FindNextFileW routine is utilized to continue the files enumeration:

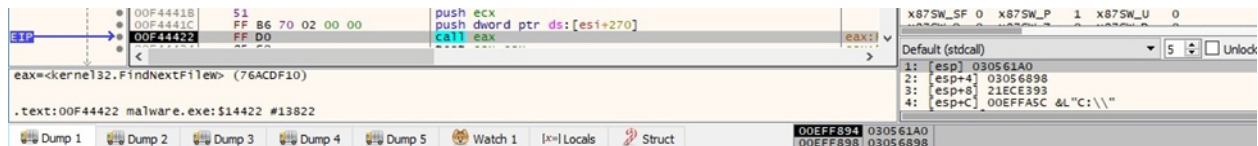


Figure 36

Black Basta ransomware calls the GetFullPathNameW API with a targeted file as a parameter:

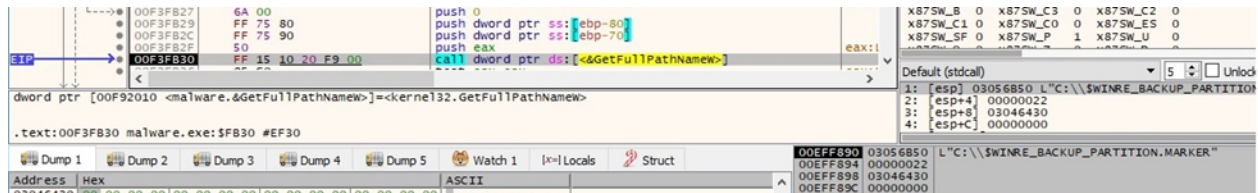


Figure 37

The process obtains a standard set of attribute information for the file via a call to GetFileAttributesExW:

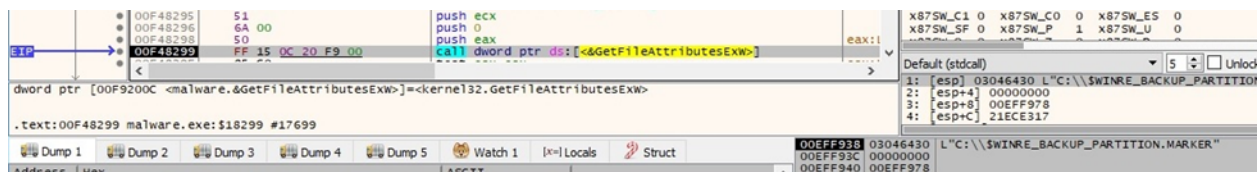


Figure 38

The ransomware has embedded a list of extensions (.exe, .cmd, .bat, and .com) in a section; however, it still encrypts these file extensions.

The executable retrieves the thread identifier of the calling thread using GetCurrentThreadId:



Figure 39

The malicious process blocks the main thread until all encryption threads finish execution (see figure 40).

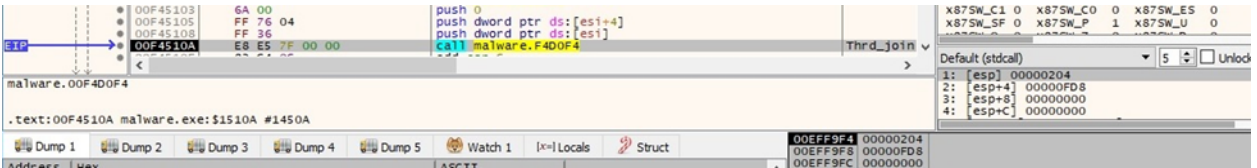


Figure 40

Thread activity – sub_F33DA0 function

The GetFileAttributesW API is utilized to retrieve file system attributes for a targeted file:

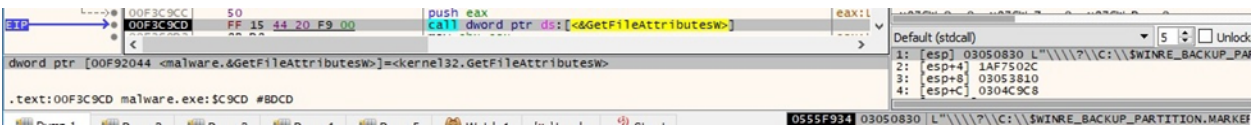


Figure 41

The malicious process opens a file for reading using wfsopen:

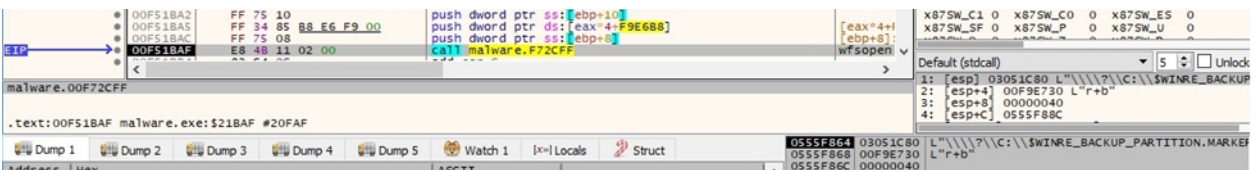


Figure 42

The ransomware moves the file pointer to the position of the last 4 bytes. Whether the file would be encrypted, these would represent the length of the encrypted ChaCha20 key and nonce, as we'll see later on:

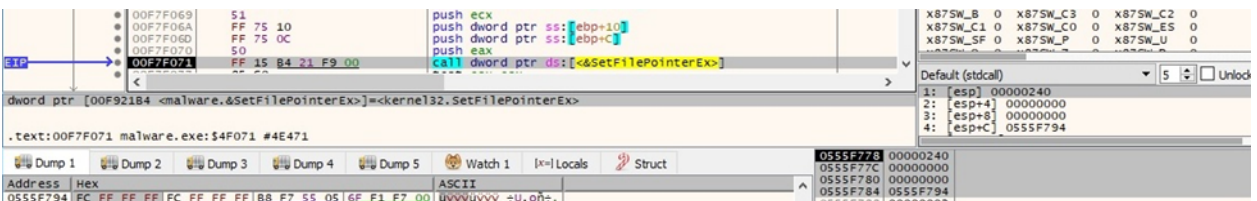


Figure 43

Black Basta ransomware generates 32 random bytes representing the ChaCha20 key and then

8 bytes representing the nonce using rand_s:

```
.text:00F3D690  
.text:00F3D690 loc_F3D690:  
.text:00F3D690 lea    eax, [ebp+arg_0]  
.text:00F3D693 push  eax  
.text:00F3D694 call   _rand_s  
.text:00F3D699 mov    al, byte ptr [ebp+arg_0]  
.text:00F3D69C add    esp, 4  
.text:00F3D69F mov    [esi+edi], al  
.text:00F3D6A2 inc    esi  
.text:00F3D6A3 cmp    esi, 28h ; '('  
.text:00F3D6A6 jnb   short loc_F3D690
```

Figure 44

Address	Hex	ASCII
03051748	69 0D BC E1 9E 49 F7 5D D2 E9 DF 20 69 DC FB AC	i.%ä.I+]öéß iÜü-
03051758	C8 04 34 F2 54 81 E3 C0 A7 AE E9 13 59 BD 6B E3	È.4òT.ää§°é.Y%kã
03051768	15 3A AE 4B 1B 1B 7A CE AB AB AB AB AB AB AB AB	.:°K..zI««««««««

Figure 45

The binary implements the RSA algorithm using the Mini-GMP library, which is fully available on [Github](#):

```
.text:00F48AB0  
.text:00F48AB0 ; Attributes: bp-based frame  
.text:00F48AB0 sub_F48AB0 proc near  
.text:00F48AB0  
.text:00F48AB0 var_C= dword ptr -0Ch  
.text:00F48AB0 var_8= dword ptr -8  
.text:00F48AB0 var_4= dword ptr -4  
.text:00F48AB0 arg_0= dword ptr 8  
.text:00F48AB0 arg_4= dword ptr 0Ch  
.text:00F48AB0 arg_8= dword ptr 10h  
.text:00F48AB0 arg_C= dword ptr 14h  
.text:00F48AB0 arg_10= dword ptr 18h  
.text:00F48AB0 arg_14= dword ptr 1Ch  
.text:00F48AB0 arg_18= dword ptr 20h  
.text:00F48AB0  
.text:00F48AB0 push  ebp  
.text:00F48AB1 mov    ebp, esp  
.text:00F48AB3 mov    eax, [ebp+arg_10]  
.text:00F48AB6 sub    esp, 0Ch  
.text:00F48AB9 cmp    [ebp+arg_14], 0  
.text:00F48ABD jnz   loc_F48BBA  
  
:FFFFFFFh  
ix  
:bp+arg_18]  
ix  
.text:00F48BBA loc_F48BBA:  
.text:00F48BBA push  offset aMpImportNails ; "mpz_import: Nails not supported."  
.text:00F48BBF call   sub_F49670  
.text:00F48BBF sub_F48AB0 endp  
.text:00F48BBF
```

Figure 46

```

.text:00F4BDA0
.text:00F4BDA0 push    ebp
.text:00F4BDA1 mov     ebp, esp
.text:00F4BDA3 sub     esp, 60h
.text:00F4BDA6 xor     eax, eax
.text:00F4BDA8 mov     [ebp+var_1C], eax
.text:00F4BDA8 mov     eax, [ebp+arg_8]
.text:00F4BDAE push   esi
.text:00F4BDAF mov     esi, [ebp+arg_C]
.text:00F4BD82 push   edi
.text:00F4BD83 mov     eax, [eax+4]
.text:00F4BD86 cdq
.text:00F4BD87 mov     ecx, eax
.text:00F4BD89 mov     eax, [esi+4]
.text:00F4BD8C xor     ecx, edx
.text:00F4BD8E sub     ecx, edx
.text:00F4BD8C cdq
.text:00F4BDC1 mov     edi, eax
.text:00F4BDC3 mov     [ebp+var_24], ecx
.text:00F4BDC6 xor     edi, edx
.text:00F4BDC8 sub     edi, edx
.text:00F4BDCA mov     [ebp+var_18], edi
.text:00F4BDCD jz      loc_F4C2B0

;st  ecx, ecx
;z   short loc_F4BE32

.text:00F4C2B0
.text:00F4C2B0 loc_F4C2B0:
.text:00F4C2B0 push   offset aMpzPowmZeroMod ; "mpz_powm: Zero modulo."
.text:00F4C2B5 call  sub_F49670

```

Figure 47

The RSA public key used to encrypt the randomly generated ChaCha20 key and the nonce is presented in the figure below:

```

.rdata:00F9D1F8 aZz11ttcaoj0zrc db 'zz11tCaoj0Zrc3xITyJf3g80U808kMvQR3vA/EVuvXFMg+jdmyjEhLhEqLATJKqg'
.rdata:00F9D1F8 ; DATA XREF: sub_F3D6B0+188To
.rdata:00F9D1F8 db '/8nWlQ2T6dPvX6ycqNxo6FYbjbmS2nmsznrRNN6e04vyXIo7c2gBwh0rSS1qSIVPs'
.rdata:00F9D1F8 db '0r2kF0mj0ES6ukt9/7gXUB7qAFQp2eY2iiraxqpI4YUM5A2EK+AYNbXymv2qqABYbB'
.rdata:00F9D1F8 db 'QuhX0yHu6z24cC4GrNRVktL0wk1FeY6JFSzG70zcfH2Jxo23oArVb/c0ZGzYHncrN'
.rdata:00F9D1F8 db 'X17bGLTPiu9ZGz+TV1jo76cvi/DF5qPfh7jq5VBzHNNXEYfdedXle9rate17Y2MAI'
.rdata:00F9D1F8 db 'EUhJPeb9oGAeN9n8jF6HTASx48+bu6Vn+EF64a1JHoEj/KGY0mw6FwN8Wex50v1I'
.rdata:00F9D1F8 db 'a/U5qQtOURCJ0o3EaOHuPG02eVUVMiE056iabcbIVR3PordNEi58t6bTgZs01SiiQ'
.rdata:00F9D1F8 db 'XRu4eT3jCTSj1FvU8VrruAinyOOLi6vI2pUgCR3axFRwldzyUuX5kd67/03yXHTuw'
.rdata:00F9D1F8 db 'VkiMTuFtpCnyj+7bMfg46LXIXC4PtzQvgxDewRppyzmT55dx28TXOwTwaGvMiZhl1'
.rdata:00F9D1F8 db 'y+66Nu+wmpHZ7u/WTDJg9H8V/AZbVDataZ1vFy2t2ws3IFVNE6dqI6lvIzgpCa4o'
.rdata:00F9D1F8 db 'xy18+53IhoZdSRpW05+A8vk0AkMWPNGU=',0

```

Figure 48

The process constructs the initial state of ChaCha20 using the key, the nonce, and some constant values:

```

.text:00F368D0
.text:00F368D0
.text:00F368D0 ; Attributes: bp-based frame
.text:00F368D0
.text:00F368D0 sub_F368D0 proc near
.text:00F368D0
.text:00F368D0 arg_0= dword ptr 8
.text:00F368D0 arg_4= dword ptr 0Ch
.text:00F368D0
.text:00F368D0 push    ebp
.text:00F368D1 mov     ebp, esp
.text:00F368D3 push   esi
.text:00F368D4 mov     esi, [ebp+arg_0]
.text:00F368D7 push   edi
.text:00F368D8 mov     edi, ecx
.text:00F368DA mov     dword ptr [edi], 'apxe'
.text:00F368E0 mov     dword ptr [edi+4], '3 dn'
.text:00F368E7 mov     dword ptr [edi+8], 'yb-2'
.text:00F368EE mov     dword ptr [edi+0Ch], 'k et'
.text:00F368F5 movzx  edx, byte ptr [esi+3]
.text:00F368F9 movzx  eax, byte ptr [esi+2]
.text:00F368FD shl     edx, 8
.text:00F36900 or      edx, eax
.text:00F36902 movzx  eax, byte ptr [esi+1]

```

Figure 49

Address	Hex	ASCII
0555F870	65 78 70 61 6E 64 20 33 32 2D 62 79 74 65 20 68	expand 32-byte k
0555F880	69 0D BC E1 9E 49 F7 5D D2 E9 DF 20 69 DC FB AC	i.%ä.I+}0éß iÜÜ-
0555F890	C8 04 34 F2 54 81 E3 C0 A7 AE E9 13 59 BD 6B E3	È.40T.äÅ@é.Ykã
0555F8A0	00 00 00 00 00 00 00 00 15 3A AE 4B 1B 1B 7A CE:®K..zİ

Figure 50

The sample obtains the current position in the targeted file by calling the fgetpos function:



Figure 51

The file content is read by the process via a call to the _read function:

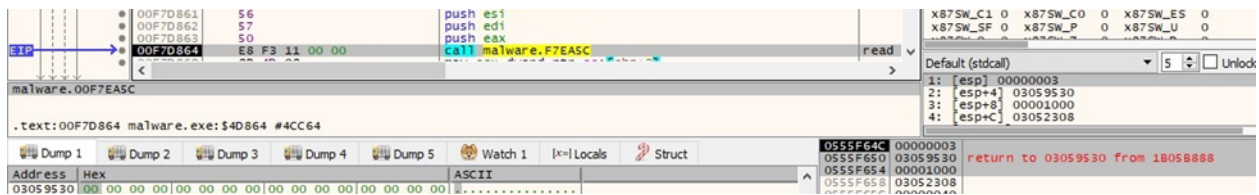


Figure 52

The content is encrypted by the ChaCha20 algorithm 64 bytes at a time:

```

.text:00F454C7
.text:00F454C7 loc_F454C7:
.text:00F454C7 add     ecx, [ebp+var_18]
.text:00F454CA xor     ebx, ecx
.text:00F454CC mov     edx, [ebp+var_1C]
.text:00F454CF rol     ebx, 10h
.text:00F454D2 add     edx, ebx
.text:00F454D4 mov     esi, edx
.text:00F454D6 xor     esi, [ebp+var_14]
.text:00F454D9 rol     esi, 0Ch
.text:00F454DC add     ecx, esi
.text:00F454DE mov     [ebp+var_40], ecx
.text:00F454E1 xor     ecx, ebx
.text:00F454E3 mov     ebx, [ebp+var_34]
.text:00F454E6 rol     ecx, 8
.text:00F454E9 mov     [ebp+var_1C], ecx
.text:00F454EC add     ecx, edx
.text:00F454EE mov     edx, [ebp+var_4]
.text:00F454F1 mov     [ebp+var_48], ecx
.text:00F454F4 xor     ecx, esi
.text:00F454F6 rol     ecx, 7
.text:00F454F9 mov     [ebp+var_4C], ecx
.text:00F454FC mov     ecx, [ebp+var_10]
.text:00F454FF add     ecx, [ebp+var_20]
.text:00F45502 xor     ebx, ecx
.text:00F45504 rol     ebx, 10h
.text:00F45507 add     edx, ebx
.text:00F45509 mov     esi, edx
.text:00F4550B xor     esi, [ebp+var_10]
.text:00F4550E rol     esi, 0Ch
.text:00F45511 add     ecx, esi
.text:00F45513 mov     [ebp+var_34], ecx
.text:00F45516 xor     ecx, ebx
.text:00F45518 mov     ebx, [ebp+var_38]
.text:00F4551B rol     ecx, 8
.text:00F4551E mov     [ebp+var_4], ecx
.text:00F45521 add     ecx, edx
.text:00F45523 mov     edx, [ebp+var_8]
.text:00F45526 add     edx, edi
.text:00F45528 mov     [ebp+var_54], ecx
.text:00F4552B xor     ebx, edx
.text:00F4552D xor     ecx, esi
.text:00F4552F mov     esi, [ebp+var_24]

```

Figure 53

Address	Hex	ASCII
03059530	5B C1 DC 78 0F DD 4B 0D 56 AC 5C 2B 73 FC 89 2A	[AU{.Yk.V-\}+sü.*
03059540	DE FD 1F 9F 4A 2A 68 AB 85 FB 7C 8D B6 87 C8 E3	by..J"heµü .ŋ.Ëä
03059550	6C 74 77 2F F6 95 2D 8D 65 4C 99 79 A4 CA F8 74	ltw/o.-.el.y=Ëot
03059560	0D D1 8C 35 A7 A2 1C E3 DF 21 F9 C6 B3 97 D3 BF	.N.5ÿc.ãß!üæ*.0¿
03059570	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
03059580	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
03059590	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
030595A0	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
030595B0	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
030595C0	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
030595D0	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
030595E0	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
030595F0	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA
03059600	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	AAAAAAAAAAAAAAAA

Figure 54

The encrypted data is written back to the file using the WriteFile API:

Debugger screenshot showing a WriteFile API call. The instruction is `jmp dword ptr ds:[<writeFile>]`. The dump window shows the hex data from Figure 54. The register window shows return addresses.

Figure 55

The buffer containing the RSA encrypted ChaCha20 key and nonce is appended to the encrypted file. The length of the encrypted information (0x200 = 512) is added as well:

Debugger screenshot showing a call to WriteFile. The instruction is `call dword ptr ds:[<writeFile>]`. The dump window shows the hex data for the key and nonce. The register window shows return addresses.

Figure 56

The encrypted file extension is changed to ".basta" using MoveFileW:

Debugger screenshot showing a MoveFileW API call. The instruction is `call dword ptr ds:[<MoveFileW>]`. The dump window shows the hex data for the file path. The register window shows return addresses.

Figure 57

Case 1 – File size < 704 bytes

In this case, the entire file content is encrypted by the ransomware:

```

SWINRE_BACKUP_PARTITION.MARKER.basta
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 5B C1 DC 7B 0F DD 4B 0D 56 AC 5C 2B 73 FC 89 2A
00000010 DE FD 1F 9F 4A 2A 68 AB B5 FB 7C 8D B6 87 C8 E3
00000020 6C 74 77 2F F6 95 2D 8D 65 4C 99 79 A4 CA F8 74
00000030 0D D1 8C 35 A7 A2 1C E3 DF 21 F9 C6 B3 97 D3 BF
00000040 34 89 77 B5 EA C9 AD 3C A9 FA AB 80 A7 CE 7A E7
00000050 7E 4F 67 DE 41 36 51 BB DF 43 99 69 7F 15 55 08
00000060 4C D9 8B 2B 77 F8 D9 59 7A A9 70 38 A5 1B 4B 79
00000070 FD 5D FE 71 56 F3 AF 26 0A 29 50 16 54 52 09 66
00000080 1C FE B9 93 BE F5 4B F2 86 92 A0 73 09 D3 49 E1
00000090 7C 26 BF 7D E6 BA E4 81 10 2D C3 B5 F5 E2 85 2B
000000A0 74 36 41 5E 7C EB AF 8A FE FB 80 F1 E8 57 8A 96
000000B0 12 15 3F 28 33 60 9F 9D 0E 61 DC 18 7D 77 B4 88
000000C0 78 45 8F E6 4B C6 EB 25 8A 74 C2 FE 49 D8 32 A6
000000D0 46 43 E2 F6 A2 E9 D7 F9 F1 51 D2 6E BE 6B 53
000000E0 36 B4 11 C7 6A DC 79 BD 9B C4 95 07 64 E9 B1 04
000000F0 3E 62 E9 A9 B1 0C A6 BC EC 3B 63 6D 0F BA D2 79
00000100 95 91 98 C9 A5 02 63 BC 93 8A DF DA 8B 79 A8 05
00000110 A5 BB 5F 94 FC 74 5A B3 59 9E A5 D1 32 83 D7 5E
00000120 CB A0 8C 3A DE E5 84 6B B9 AB F7 8E ED 65 98 1B
00000130 B0 2A 75 21 C7 AD DA FC 3C 86 11 05 77 06 DE B2
00000140 31 BA B1 AF 01 BA 50 48 77 6A A0 A9 E2 93 83 AD
00000150 53 C3 99 1E EE 11 E0 C8 0C 15 24 5E 90 73 22 B7
00000160 C8 6D 65 09 0A 10 29 E4 E3 E6 8B 6D 85 A0 96 5E
00000170 FE D0 35 19 0D 07 D6 06 C1 A2 7D BA 16 2C B5 0D
00000180 3C 16 EF 4B F7 E0 02 FF 53 E8 A5 A9 9A 7B 9A 0F
00000190 49 9D C2 3E 3A 30 5A 7A 72 A1 9C E7 28 69 A5 B0
000001A0 8A ED 7A EE BC 07 7D AD D2 35 6D DD FC AE C6 8B
000001B0 D5 C8 92 76 3B 74 89 33 0C 03 D4 56 CD CE F8 D9
000001C0 4B 60 30 B2 A9 49 A6 A5 A3 C0 A3 CC EB 39 1A CE
000001D0 EC 7F 52 75 9E 1C FC 55 48 95 78 9B B3 5C 7C B6
000001E0 CE AB 17 06 E6 A7 17 C0 30 D2 55 FD B1 6A A8 4D
000001F0 1A D2 67 4C 94 57 CE CC 8A FA 53 D6 E1 09 94 AC
00000200 F8 B1 20 70 E5 23 79 BA 9B 9A F3 6C DE A6 13 89
00000210 63 03 FA 71 C9 BA DD E7 15 88 7E 9D 4B 70 84 CB
00000220 A2 59 DE 56 62 4E D0 1E DA DA 7A 4D AA 17 F1 F4
00000230 28 36 B1 E1 48 E8 5D 3E 34 93 BB 8E 87 B3 80 16
00000240 06 5B 66 8B 95 8F 71 47 9C 16 54 92 FA 78 15 11
00000250 09 10 9B C8 53 57 BB 34 D5 64 92 94 1D 92 96 35
00000260 CD 8D CB 6A 75 AB DC AE AE BB 3F 6D CA 03 E1 67
00000270 FA 2F 8D 9D A8 A0 CF EF 32 17 20 A8 93 45 F7 B1
00000280 43 AF CF C1 15 2C D6 A9 31 5C 6C 5F FF 81 30 52
00000290 FF C6 96 09 D7 1D 41 D7 B8 96 11 BB 74 64 C9 7D
000002A0 04 3F 72 F6 3E E1 A3 D5 D7 A8 10 DA 38 0F D1 84
000002B0 4A 56 1F E2 AC 50 A6 B3 E1 AF 4B FD 67 3F F9 13
000002C0 89 57 B2 FC 1D B4 DC B1 A2 A6 DF 45 78 31 2E 16
000002D0 BD 07 4E FA A5 79 7E 2F 32 12 80 17 FA B6 8C 85
000002E0 00 02 00 00

```

```

[[[AÛ{.ÝK.V-\+sùk*
Fý.ÝJ*h«µù|.†±Èã
ltw/ö*- .eL™yKÈøt
.ÑE5Sc.ãã!ùE³-Ó¿
4twµèÉ.<@ú«@SÍzç
~OgBA6Q»8C™i..U.
LÛ«+wøÛYz@p8¥.Ky
ý]pqVó⁻&.)P.TR.f
.p³™%òKòt' s.ÓIá
|&¿)æ°ä..-Äµðã...+
t6A^|è⁻ŠpúèñèWŠ-
...?(3`ÿ..aÛ.)w`^
xE.æKEè%ŠtÄpI02|
FCãöcé«*ùñQ0n%kS
6`.çjÛy«>Ä°.dé±.
>bé@±.¡;i;cm.°òy
.«`É¥.c«"Š8Ú<y".
¥»_ "ütZ³Yz¥Ñ2f«^
È Æ:Pã„k²«-Žie".
°*u!Ç.Úú<+..w.P²
1°±⁻.°PHwJ @ã"f.
SÄ™.i.àÈ..$^s"-
Ème...)ããæm... -^
pD5...Ö.Äc)°..µ.
<.iK-à.ySè¥@š(š.
I.Â>:0Zzr;æç(i¥°
Šizi¼.).05mÛú@È<
ÖÈ'v;t#3..ÖVíføÛ
K'0°@I;¥£ÄèIè9.Í
i.Ruž.úUH*x>⁻\|¶
Í«..æS.À00Uý±j"™
.ÖgL"WíiŠúSÓá."⁻
ø± pã#y°>šó1B!..%
c.úqÉ°Ýç. ^~.Kp„È
cYFVbNÐ.ÚÚzM².ñó
(6±áHè]>4"»Ž+³€..
.[Í<°.qGø.T'úx..
..>ÈSW»4Öd'".'-5
Í.Èju«Ü00»?mÈ.ág
ú/.." Íi2. "E±±
C⁻IÁ.,Ö01\1_y.0R
ýE-.x.Ax,-.»tdÉ}
.¿r0>á£Öx".Úš.Ñ„
JV.ã-P;³á⁻Kýg?ù.
%W²ú. 'Ú±c|8Ex1..
%.Nú¥y~/2.È.úqE...
....

```

Figure 58

Case 2 – File size < 4KB

In this case, the file is partially encrypted. The ransomware encrypts 64 bytes, skips 192 bytes,

encrypts 64 bytes again, and so on.

```
1KB.exe.basta
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 6C FE 7A E5 94 39 63 A1 A6 19 45 42 65 B3 1F 83 1pzâ"9c;|.EBe³.f
00000010 F4 A2 25 4E 22 2F 62 E9 6A E2 19 5E 97 AF 91 40 ôc%N"/béjâ.^-'\@
00000020 5C 45 A0 96 2B 36 C6 69 1A 74 D3 EE 1A 73 71 F2 \E -+6Ei.tÓi.sqò
00000030 97 5C 44 08 E2 CF 2E 98 99 DC BF 57 DC E1 58 65 -\D.âï.~"Ü¿WÜáXe
00000040 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000050 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000060 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000070 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000080 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000090 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000000A0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000000B0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000000C0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000000D0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000000E0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000000F0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000100 35 08 A4 ED AB D6 81 C7 3B 8A 04 BA 65 2C 25 13 5.¼i«Ö.Ç;Š.°e,%.
00000110 94 7B E4 A1 CE AD 9B D0 1F 6C 9F DA 66 7F 66 D0 "{ä;î. »D.1YÚf.fÐ
00000120 3D 1A 54 D4 4F 95 A4 31 D6 FC FA 9F B3 AB F3 03 =.TÔO•¼1Öúúÿ'«ó.
00000130 15 1D B8 62 3F 9D 1B F0 DD 29 16 13 76 5E 19 FE ..,b?..δÝ)..v^p
00000140 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000150 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000160 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000170 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000180 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000190 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000001A0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000001B0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000001C0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000001D0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000001E0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
000001F0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000200 60 4D 0B 33 40 DF D4 98 1D 10 E1 C5 D3 0F AC 60 `M.3@BÔ~..áÁÓ.-`
00000210 7A 99 98 45 35 75 AD 4E 24 37 49 5C 49 5E FE 45 z"~E5u.N$7I\I^pE
00000220 E1 89 BE E4 08 EF FC 22 10 DA 62 F1 B5 A0 80 E7 á%ä.iü".Úbñµ €ç
00000230 AE BF 91 27 84 39 D5 BA 65 26 85 6A 52 FE C0 1B ©¿ '' „9Ö°e&...jRpÀ.
```

Figure 59

Case 3 – File size > 4KB

In this case, the file is partially encrypted. The ransomware encrypts 64 bytes, skips 128 bytes, encrypts 64 bytes again, and so on.

```

4KB.exe.basta
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000F30 1F 1F 75 45 88 D1 FE 36 E4 C1 18 EF 37 0A 58 FC ..uE^Np6aA.i7.Xu
00000F40 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000F50 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000F60 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000F70 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000F80 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000F90 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000FA0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000FB0 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAA
00000FC0 70 98 6C 3E D2 19 3C A8 51 C8 01 CB F0 F9 76 0F p~1>Ò.<"QÈ.Èsùv.
00000FD0 E1 63 4A 31 A0 1E D3 AF 99 F6 98 B3 92 89 9F DA ácj1 .Ó™ø~'':YÚ
00000FE0 12 E0 F2 9E 50 23 C0 27 94 7A D4 55 46 32 3E 09 .àòZP#À'"zÔUF2>.>
00000FF0 C9 FA DB 8C 12 55 9E 5C 89 6E 69 DF 11 E6 5A 26 ÉúŪE.UZ\kniš.æZ&
00001000 5C 35 42 5A CC 47 79 1A CA FF F9 A3 A2 C5 28 59 \5BZiGy.Êyù&cA(Y
00001010 F7 57 0D 06 D3 F7 81 24 65 7F 1A 2C 7F 82 AE C9 +W..Ó÷.Œe...,,@É
00001020 4D 12 18 EE EE 92 FF 57 B3 2E DE 8D D8 B8 E2 27 M..ií'yW'.P.Ø.â'
00001030 3E 85 07 BA 8E 86 D6 F2 F3 CE 97 26 F1 52 10 54 >...°Ž+ÒòóÍ-ššR.T
00001040 D6 D7 77 DA B0 08 33 66 CF 3B 57 1F B1 77 56 AF Ö×wÚ°.3fİ;W.±wV
00001050 41 86 EA A2 DB B0 4F BB 0D 3C 0A 48 2B 3A 5C DB A+êcŪ°O>.<.H+:\Ū
00001060 2B 48 48 13 FD 9E 1C C8 38 C5 14 A7 76 F1 DB 28 +HH.yž.ÈšÅ.ŠvñŪ(
00001070 9A 50 75 6C AD FC 02 61 1E 6C EE E7 14 04 C7 45 šPul.ü.a.liç..ÇE
00001080 19 51 53 58 1D 4F C7 C1 5B 0D 79 A0 9C 4A 17 C5 .QSX.OÇÁ[.y œJ.Å
00001090 9D 55 AC 83 95 A7 F3 31 23 95 68 7C 4E 40 86 C1 .U-f*šól#*n|Nê+Å
000010A0 6D 5A 1D 2B E4 07 A2 BD 88 58 40 DC 70 EE 76 B0 mZ.+ä.cš~X@Ūpiv°
000010B0 A6 9A 18 B4 2A 79 0D 71 64 4B 25 91 F6 18 BD B0 !š.'*y.qdK%`ò.š°
000010C0 8B 1A 2A FD 80 52 04 5C 20 2D 19 CF E4 09 C1 90 <.*yêR.\-.iä.Å.
000010D0 8F 63 3F 88 D0 7A 15 B1 B8 1F 8F 5A 5D 73 93 9F .c?^Dz.±,..Z]s~Y
000010E0 44 1D 24 B5 CF FA FD 5F 42 71 0A B1 BD 8F BE CC D.šuiúy Bq.±š.šİ
000010F0 BF E9 F8 7E 9D FD 74 9F 9F 54 AE 09 EA E4 15 37 çéø~.ýtYŪT@.èä.7
00001100 49 F0 AE 02 02 68 53 C7 1F 04 64 AF 8D CF 75 78 Ið@..hSÇ..d~.İux
00001110 EA FC 03 CE 75 BB 1B 35 B5 6E 9D 6F 2A 93 84 1F èü.İu».5un.o*"„.
00001120 D9 BC E6 A2 43 B7 2A 9E 6B 42 30 ED 8E EF 27 7C Ū~æcC.*žkB0iž'i'|
00001130 AE 49 B3 64 CB 26 C5 94 84 A5 A8 8D 10 F9 27 6D @I'dÈ&Å"„Ÿ"'.ù'm
00001140 01 34 F2 C7 B5 2F 4C DD 51 61 F9 61 97 19 5A 86 .4òçµ/LYQaua-.Z+
00001150 D3 D8 7C 5A A7 74 A7 8D CF C9 0D BD 82 C0 EA 4C ÓØ|ZStS.İÉ.š.ÀèL
00001160 7F 98 FB 43 BD 19 F5 2A C3 9E E2 EF 05 80 2C C4 .~úCš.š*Åžâi.ε.Å
00001170 43 63 3F 5B 72 5F CA 71 F3 61 E3 76 50 52 BE FE Cc?[r ÊqóaaŷvPRšp
00001180 C2 5F FD 14 A7 B4 37 A7 56 F5 F6 68 44 5D 9A 68 Å_y.S^7šVðòhD]šh
00001190 99 08 EE C4 F3 64 F7 AC D2 2D D6 09 78 3C D0 4C ™.iÄód÷-ò-ò.x<DL
000011A0 C4 3B 0A 9E 36 17 7F 2F F8 95 A2 89 80 7B B6 13 Ä;.ž6../ø*cšE(Ū.
000011B0 27 43 95 5E 77 27 0E 71 D4 81 4D D2 56 2D 04 4C 'C^w'.qð.MÖV-.L
000011C0 26 73 08 81 F4 2A F0 D8 D3 36 3F 90 01 01 60 A4 &s..š*ð006?...`š
000011D0 FD 3D 99 FB 55 FA D9 3B 56 85 60 3A 19 2D CF D2 ý=™ŪUúŪ;V...`.-İò
000011E0 E7 7D CF 1E B4 B2 63 8A 36 E1 D9 14 83 DC 3D 99 ç}İ.'`cšŠáŪ.fŪ=™
000011F0 C0 62 10 A5 6E 33 75 14 04 B5 C4 0D 11 99 39 26 Åb.ŷn3u..pÄ..™9&
00001200 00 02 00 00

```

Figure 60

Finally, the ransomware tries to write the time spent during the execution and the total size of

encrypted files to the console; however, it raises an error because the process was detached from its console:

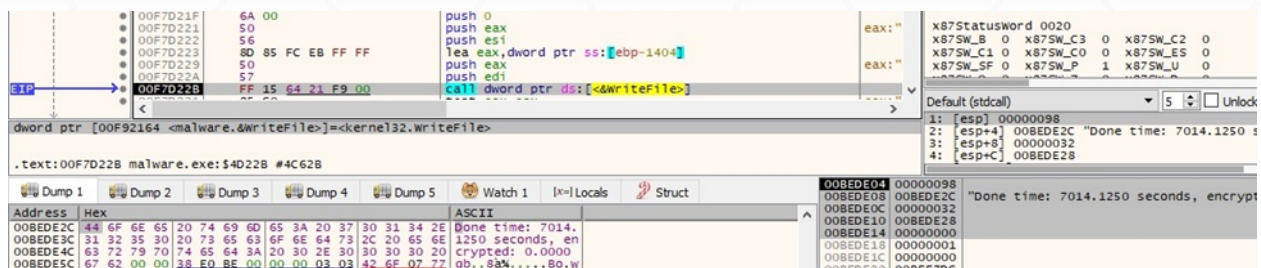


Figure 61

Indicators of Compromise

Black Basta Ransom Note

readme.txt

Files created

%Temp%\fkdjsadasd.ico

%Temp%\dlaksjdoiwq.jpg

Processes spawned

cmd.exe /c "C:\Windows\SysNative\vssadmin.exe delete shadows /all /quiet"

cmd.exe /c "C:\Windows\System32\vssadmin.exe delete shadows /all /quiet"

Registry key created

HKEY_CLASSES_ROOT\.basta