
Malware Analysis Report

Sample Information

Sample Name: Ransomware.wannahusky.exe
MD5 Hash: 0287b38f8240a025b30c0a231ea403fc
SHA1 Hash: 691ac1b4b7b494f7b56eff0b48ba3e31a14e0d7d
Fuzzy Hash: 12288:dusXq0SdXCgSIAIZm/OSJsqrHrLOBHzI5WTVz:Qiq0S
dXCgSIAIZm/tJsqrLCBHzI5WTb
Imphash: a97ffe6ec502dacc4c154f9dc2b58725

Executive Summary

This Nim compiled binary performs a very small set of actions. The sample is ransomware, however, it only targets one file for encryption. The file `cosmo.jpeg` must be located on the Desktop for all actions to occur. If this file is present, the ransomware will execute properly. The Desktop wallpaper will be replaced with the ransom message, the file `cosmo.jpeg` will be encrypted, and the `tree` command will be executed. If the target file is not found, the only action taken is the `tree` command. No network activity was found.

Technical Analysis

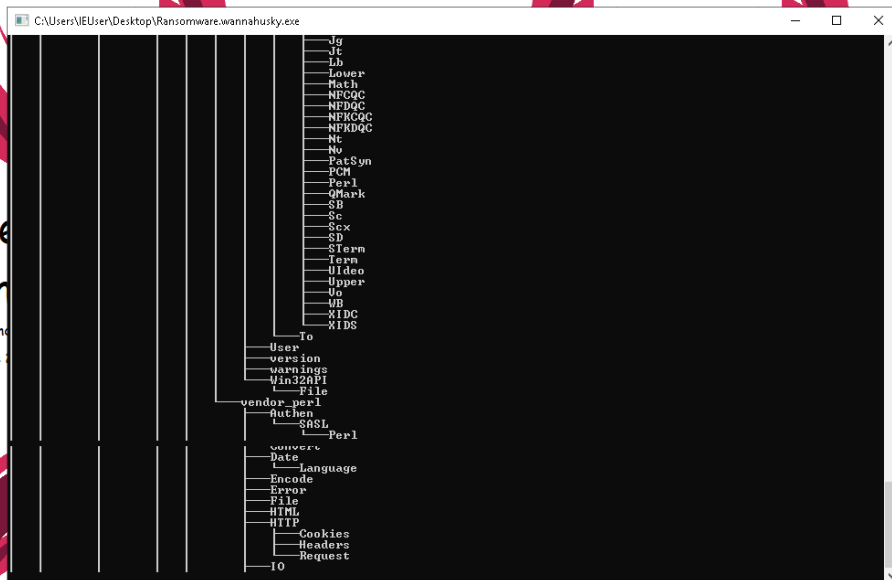
This sample appears to be compiled using Nim. This can be confirmed by looking at the symbols within the binary (Figure 1).



```
Symbols nim
_nimAddInt
_nimSubInt
_nimToCStringConv
_nimAddInt
_nimSubInt
_nimAddInt
_nimSubInt
_nimToCStringConv
_nimZeroMem
_nimGC_setStackBottom
@nimGCvisit@8
@nimIntToStr@4
@toNimStr@8
@cstringToNimStr@4
@nimRegisterThreadLocalMarker@4
@nimLoadLibrary@4
@nimLoadLibraryError@4
@nimGetProcAddr@8
@nimRegisterGlobalMarker@4
@nimInt64ToStr@8
@nimLeaveFinally@0
_nimAddInt
_nimAddInt
_nimSubInt
_nimAddInt.constprop.0
_nimAddInt
@nimcrypto_sysrandInit000@0
@nimcrypto_sysrandDatInit000@0
_nimAddInt
_nimSubInt
NimMain
@NimMainModule@0
_NimMainInner
```

Figure 1 - Symbol names indicating binary was compiled in Nim. The most telling is the presence of NimMain.

On first detonation, a command prompt window was seen running a command (Figure 2) and the Desktop wallpaper was changed to a ransom message (Figure 3). The ransom message appears to specifically mention a picture of cosmo being targeted for ransom. This will be confirmed through additional analysis later.



3 of 14



that picture of cosmo on your
desktop is now encrypted!

to save him, you must send 100 Huskycoin to <https://huskyhacks.dev>

hurry! you have 24 hours before we delete cosmo

Figure 3 – Ransom message set as the wallpaper to inform the user of their infection.

Utilizing procmon, a historical process tree is generated to understand the number of additional processes spawned by the sample (Figure 4).

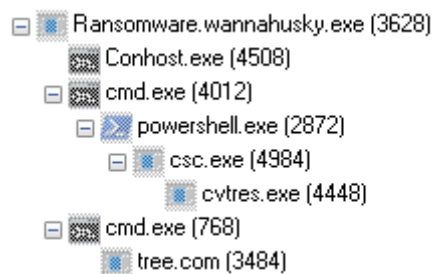


Figure 4 – Historical process tree generated from Procmon logs.

Through sandbox analysis, the basic steps of the ransomware sample were identified. Advanced analysis techniques were utilized to see the advanced details of the operations. First, the sample saves a PNG file to the Desktop called WANNAHUSKY.png (Figure 5). This PNG file is located within the sample's binary at offset 0x412100 (Figure 6). The sample will then write a Powershell script named ps1.ps1 to the Desktop as well (Figure 7). The Powershell script is located at offset 0x411ea0 within the sample's binary (Figure 8). The executable will then proceed to perform the encryption step. The only file it targets is "cosmo.jpeg" which was saved by the analyst to the Desktop before detonation (Figure 9).

```
7:52:0... Ransomware.w... 3628 CreateFile C:\Users\IEUser\Desktop\WANNAHUSKY.png
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\IEUser\Desktop\WANNAHUSKY.png
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\IEUser\Desktop\WANNAHUSKY.png
7:52:0... Ransomware.w... 3628 CloseFile C:\Users\IEUser\Desktop\WANNAHUSKY.png
```

Figure 5 - Sample saving WANNAHUSKY.png to the Desktop.

```
00412100 TM njEKfyRiYvomtvTKocFw0w_12:
00412100 dd 7e 00 00 dd 7e 00 40-89 50 4e 47 0d 0a 1a 0a-00 00 00 0d 49 48 44 52-00 00 02 aa 00 00 02 b9 .~...~.PNG.....IHDR.....
00412120 08 02 00 00 00 f8 a5 9b-08 00 00 00 01 73 52 47-42 00 ae ce 1c e9 00 00-00 04 67 41 4d 41 00 00 .....sRGB.....gAMA..
00412140 b1 8f 0b fc 61 05 00 00-00 09 70 48 59 73 00 00-0e c3 00 00 0e c3 01 c7-6f a8 64 00 00 7e 72 49 ....a.....pHYs.....o.d..~rI
00412160 44 41 54 78 5e ed bd dd-e8 66 c7 75 ef 99 bf 62-8c 84 84 44 3a 12 42 30-33 90 1b 43 4f ac e8 68 DATx^...f.u...b...D:.B03..CO..h
00412180 1a c7 f8 c8 28 8d 10 62-60 26 9a c1 d0 37 b1 6e-e6 ca 9e 5c 38 28 0a 62-14 e1 f4 20 81 15 c8 31 ....(.b"&...7.n....\8(.b... ..1
004121a0 34 32 13 05 04 7d 31 4a-73 50 74 2e 3a 83 71 90-90 e7 22 63 5b a4 07 21-47 a2 c1 42 88 c8 91 11 42...}1JsPt...q...c[...!G..B...
004121c0 02 43 a6 f6 de eb d9 4f-d5 77 d7 ae f7 aa fd f6-fd f0 85 ee df b3 ab 56-ad 5a 55 bb d6 f3 ec d7 .C.....0.w.....V,ZU....
004121e0 df fa 77 42 08 21 84 1c-0c a6 7f 42 08 21 e4 70-30 fd 13 42 08 21 87 83-e9 9f 10 42 08 39 1c 4c ..wB.!.....B.!..p0..B.!.....B.9..L
```

Figure 6 - PNG file header found within sample binary at offset 0x412100.

```
7:52:0... Ransomware.w... 3628 CreateFile C:\Users\IEUser\Desktop\ps1.ps1
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\IEUser\Desktop\ps1.ps1
7:52:0... Ransomware.w... 3628 CloseFile C:\Users\IEUser\Desktop\ps1.ps1
```

Figure 7 - Sample saving ps1.ps1 to the Desktop.

```
00411ea0 TM njFKfyRiYvomtyTKocFwDw 15:
00411ea0 30 02 00 00 30 02 00 40-24 63 6f 64 65 20 3d 20-40 27 0a 75 73 69 6e 67-20 53 79 73 74 65 6d 2e 0...0..@$code = @'.using System.
00411ec0 52 75 6e 74 69 6d 65 2e-49 6e 74 65 72 6f 70 53-65 72 76 69 63 65 73 3b-0a 0a 6e 61 6d 65 73 70 Runtime.InteropServices;..namesp
00411ee0 61 63 65 20 57 69 6e 33-32 7b 0a 20 20 20 0a-20 20 20 70 75 62 6c-69 63 20 63 6c 61 73 73 ace Win32{. public class
00411f00 20 57 61 6c 6c 70 61 70-65 72 7b 0a 0a 20 20 20 20-20 20 20 5b 44 6c 6c 49-6d 70 6f 72 74 28 22 75 Wallpaper{.. [DllImport("u
00411f20 73 65 72 33 32 2e 64 6c-6c 22 2c 20 43 68 61 72-53 65 74 3d 43 68 61 72-53 65 74 2a 41 75 74 6f ser32.dll", CharSet=CharSet.Auto
00411f40 29 5d 0a 20 20 20 20-20 73 74 61 74 69 63 20-20 65 78 74 65 72 6e 20-69 6e 74 20 53 79 73 74 )}. static extern int Syst
00411f60 65 6d 50 61 72 61 6d 65-74 65 72 73 49 6e 66 6f-20 28 69 6e 74 20 75 41-63 74 69 6f 6e 20 2c 20 emParametersInfo (int uAction ,
00411f80 69 6e 74 20 75 50 61 72-61 6d 20 2c 20 73 74 72-69 6e 67 20 6c 70 76 50-61 72 61 6d 20 2c 20 69 int uParam , string lpvParam , i
00411fa0 6e 74 20 66 75 57 69 6e-49 6e 69 29 20 3b 0a 0a-20 20 20 20 70 75-62 6c 69 63 20 73 74 61 nt fuWinIni) ;.. public sta
00411fc0 74 69 63 20 76 6f 69 64-20 53 65 74 57 61 6c 6c-70 61 70 65 72 28 73 74-72 69 6e 67 20 74 68 65 tic void SetWallpaper(string the
00411fe0 50 61 74 68 29 7b 0a 20-20 20 20 20 20 20 20-20 53 79 73 74 65 6d 50 61-72 61 6d 65 74 65 72 73 Path){. SystemParameters
00412000 49 6e 66 6f 28 32 30 2c-30 2c 74 68 65 50 61 74-68 2c 33 29 3b 0a 20 20-20 20 20 7d 0a 20 20 Info(20,0,thePath,3);. }.
00412020 20 20 7d 0a 7d 0a 27 40-0a 61 64 64 2d 74 79 70-65 20 24 63 6f 64 65 0a-0a 24 63 75 72 72 44 69 }.}.@.add-type $code..$currDi
00412040 72 20 3d 20 47 65 74 2d-4c 6f 63 61 74 69 6f 6e-0a 24 77 61 6c 6c 70 61-70 65 72 20 3d 20 22 2e r = Get-Location.$wallpaper = ".
00412060 5c 57 41 4e 4e 41 48 55-53 4b 59 2e 50 4e 47 22-0a 24 66 75 6c 6c 70 61-74 68 20 3d 20 4a 6f 69 \WANNAHUSKY.PNG". $fullpath = Joi
00412080 6e 2d 50 61 74 68 20 2d-70 61 74 68 20 24 63 75-72 72 44 69 72 2d 43-68 69 6c 64 50 61 74 68 n-Path -path $currDir -ChildPath
004120a0 20 24 77 61 6c 6c 70 61-70 65 72 0a 0a 5b 57 69-6e 33 32 2e 57 61 6c 6c-70 61 70 65 72 5d 3a 3a $wallpaper..[Win32.Wallpaper]::
004120c0 53 65 74 57 61 6c 6c 70-61 70 65 72 28 24 66 75-6c 6c 70 61 74 68 29 0a-00 00 00 00 SetWallpaper($fullpath).....
```

Figure 8 - Powershell script stored within sample binary at offset 0x411ea0.

```
7:52:0... Ransomware.w... 3628 CreateFile C:\Users\VEUser\Desktop\cosmo.jpeg SUCCESS Desired Access: Generic Reac
7:52:0... Ransomware.w... 3628 QueryStandard... C:\Users\VEUser\Desktop\cosmo.jpeg SUCCESS AllocationSize: 1,757,184, End
7:52:0... Ransomware.w... 3628 ReadFile C:\Users\VEUser\Desktop\cosmo.jpeg SUCCESS Offset: 0, Length: 1,753,088, F
7:52:0... Ransomware.w... 3628 ReadFile C:\Users\VEUser\Desktop\cosmo.jpeg SUCCESS Offset: 1,753,088, Length: 1,5;
7:52:0... Ransomware.w... 3628 ReadFile C:\Users\VEUser\Desktop\cosmo.jpeg SUCCESS Offset: 1,754,626, Length: 4,0;
7:52:0... Ransomware.w... 3628 CloseFile C:\Users\VEUser\Desktop\cosmo.jpeg SUCCESS
7:52:0... Ransomware.w... 3628 CreateFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Desired Access: Generic Write
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 0, Length: 4,096, Priorit
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 4,096, Length: 4,096, F
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 8,192, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 12,288, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 16,384, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 20,480, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 24,576, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 28,672, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 32,768, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 36,864, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 40,960, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 45,056, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 49,152, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 53,248, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 57,344, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 61,440, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 65,536, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 69,632, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 73,728, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 77,824, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 81,920, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 86,016, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 90,112, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 94,208, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 98,304, Length: 4,096
7:52:0... Ransomware.w... 3628 WriteFile C:\Users\VEUser\Desktop\cosmo.WANNAHUSKY SUCCESS Offset: 102,400, Length: 4,096
```

Figure 9 - Sample reading cosmo.jpeg test file and writing the encrypted bytes to cosmo.WANNAHUSKY on the Desktop.

It can be confirmed that cosmo.jpeg is the only file targeted by this binary by performing static code analysis. Within the strings, the only filename mentioned is cosmo.jpeg and the encrypted version name of cosmo.WANNAHUSKY (Figure 10). The cross references of these strings show the use of them within a function where the encryption function is called. When looking at the function, the variable containing these strings are appended to a variable containing the path of the user's home directory (Figure 11). The appended string is then utilized for read/write operations using CreateFileA, ReadFile, and WriteFile API calls in which the memory buffer of ReadFile will be passed into the encryption function.

```
0041a0a0 1f 00 00 00 1f 00 00 40-44 65 73 6b 74 6f 70 5c-74 61 72 67 65 74 5c 63-6f 73 6d 6f 2e 57 41 4e .....@Desktop\target\cosmo.WAN
0041a0c0 4e 41 48 55 53 4b 59 00 NAHUSKY.
0041a0c8 _TM_njFKfyRiYvmtvTKocFwDw_5:
0041a0c8 12 00 00 00 12 00 00 40-44 65 73 6b 74 6f 70 5c-63 6f 73 6d 6f 2e 6a 70 .....@Desktop\cosmo.jp
0041a0e0 65 67 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 eg.....
```

Figure 10 - Strings in sample specifically targeting cosmo.jpeg.

```
uint32_t ebx_1 = eax_1
int32_t eax_2 = homeDirectoryPath
int32_t ecx = 0x12
if (eax_2 != 0)
    ecx = *eax_2 + 0x12
void* eax_3
eax_3, ecx = @rawNewString@4(ecx)
_appendString(ecx, homeDirectoryPath)
_appendString.part.0(eax_3, &cosmo.jpeg)
int32_t eax_5 = homeDirectoryPath
int32_t ecx_2 = 0x1f
if (eax_5 != 0)
    ecx_2 = *eax_5 + 0x1f
void* eax_6
eax_6, ecx_2 = @rawNewString@4(ecx_2)
_appendString(ecx_2, homeDirectoryPath)
_appendString.part.0(eax_6, &cosmo.WANNAHUSKY)
int32_t* eax_7 = @readFile__4PGnM9bWmsH0Nu7dnr3XzgA@4(eax_3)
```

Figure 11 - Decompiled function where the sample is appending the user's HOME directory with the /Desktop/cosmo.jpeg string to specifically read the file in preparation for encryption.

An additional confirmation of the cosmo.jpeg file being the only target is that if this file is not present, the sample outputs in error in a console window (Figure 12). The only step that is then taken by the sample is the tree command. No other actions are taken by the sample and the user's cosmo picture remains safe.

```
Administrator: C:\Users\IEUser\Desktop\Ransomware.wannahusky.exe
cannot open: C:\Users\IEUser\Desktop\cosmo.jpeg
```

Figure 12 - Sample will output an error into console if no cosmo.jpeg exists on the Desktop.

There is also the issue of the encryption process itself and the way it encrypts files. The bytes of cosmo.jpeg is stored in a memory buffer using the ReadFile API call. This buffer is then put through the encryption process. However, the original cosmo.jpeg file is deleted using DeleteFileW instead of being

overwritten (Figure 13). The encrypted bytes are stored in a new file called cosmo.WANNAHUSKY. This allows the original file to be recovered from the MFT via traditional data recovery techniques.

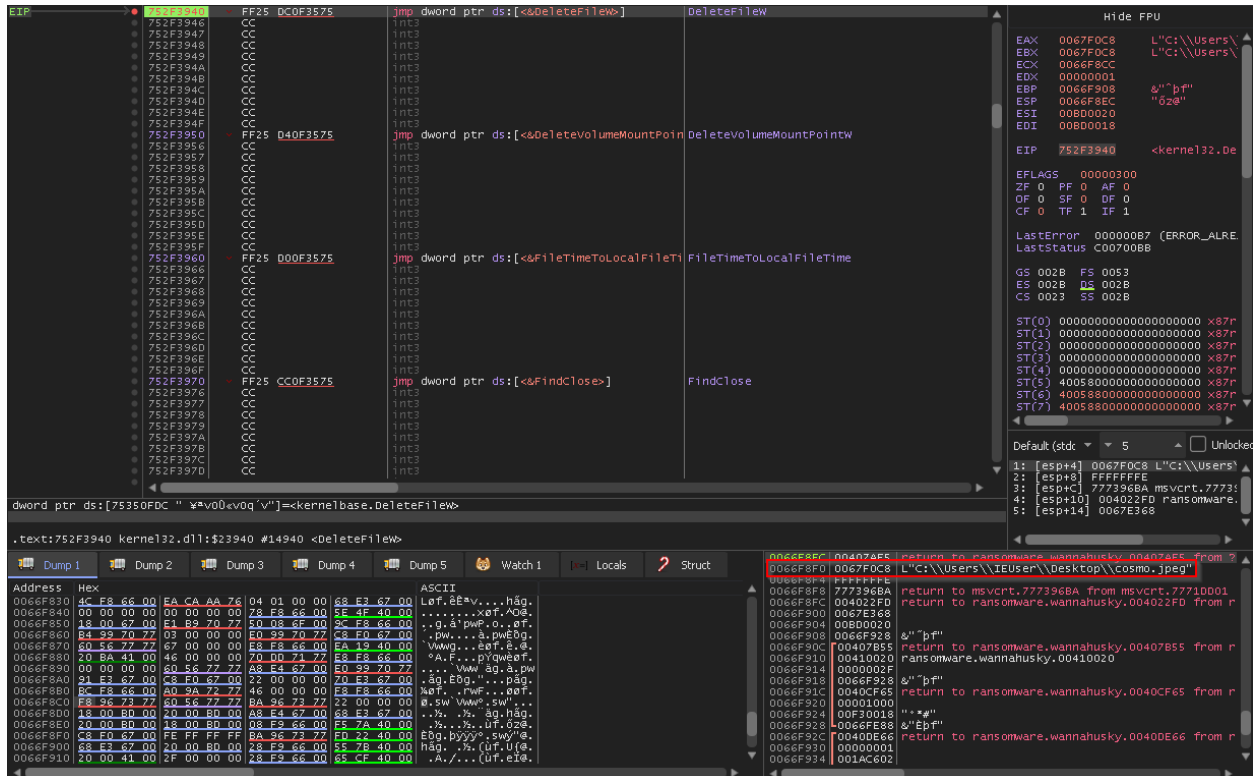


Figure 13 - The original cosmo.jpeg being deleted via DeleteFileW instead of being overwritten.

After the encryption routine is finished, the sample spawns a new cmd.exe process which executes the dropped Powershell script (Figure 14). After execution, the script is deleted from disk using DeleteFileW (Figures 15 and 16).

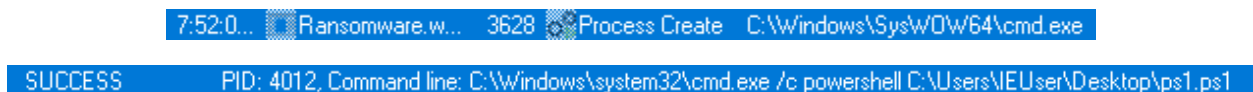


Figure 14 - Sample spawning a new cmd.exe process that calls powershell in order to execute the dropped ps1.ps1 script.

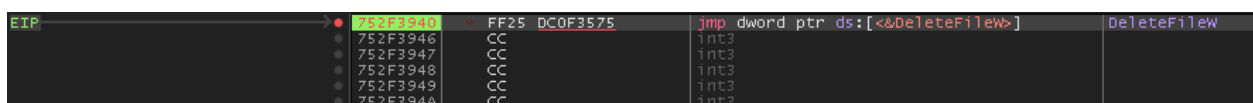


Figure 15 - DeleteFileW API call used to remove Powershell script from disk.

0066FE4C	00407AF5	return to ransomware.wannahusky.00407AF5 from ???
0066FE50	0067F2F8	L"C:\Users\IEUser\Desktop\ps1.ps1"
0066FE54	00000000	
0066FE58	0066FE78	&"bf"
0066FE5C	00404F88	return to ransomware.wannahusky.00404F88 from ransomware.wannahusky.00404F3A
0066FE60	0067B4A8	
0066FE64	00000032	
0066FE68	0066FE88	&"ëbf"

Figure 16 - Argument on the stack of the file to be deleted. Shows the ps1.ps1 file being targeted for deletion.

By setting a breakpoint on DeleteFilew, the script was able to be viewed by the analyst before being removed by the sample (Figure 17). This script is responsible for setting the ransom message PNG file as the Desktop wallpaper.

```

1 $code = @'
2 using System.Runtime.InteropServices;
3
4 namespace Win32{
5
6     public class Wallpaper{
7
8         [DllImport("user32.dll", CharSet=CharSet.Auto)]
9         static extern int SystemParametersInfo (int uaction , int uParam , string lpvParam , int fuWinIni) ;
10
11         public static void SetWallpaper(string thePath){
12             SystemParametersInfo(20,0,thePath,3);
13         }
14     }
15 }
16 ' @
17 add-type $code
18
19 $currDir = Get-Location
20 $wallpaper = ".\WANNAHUSKY.PNG"
21 $fullpath = Join-Path -path $currDir -ChildPath $wallpaper
22
23 [Win32.Wallpaper]::SetWallpaper($fullpath)
24

```

Figure 17 - The contents of the Powershell script written to the Desktop.

The final step of the sample is the spawning of a new cmd.exe process using CreateProcessA which executes the tree command on disk C:\ (Figures 18, 19, and 20).

```

7:52:1... Ransomware.w... 3628 Process Create C:\Windows\System32\cmd.exe
SUCCESS PID: 768, Command line: C:\Windows\system32\cmd.exe /c tree C:\

```

Figure 18 - Sample spawning a new cmd.exe process that calls the tree command with the C:\ drive as the argument.

EIP	75303050	8BFF	mov edi,edi	CreateProcessA
	75303052	55	push ebp	
	75303053	8BEC	mov ebp,esp	
	75303055	5D	pop ebp	
	75303056	FF25 88143575	jmp dword ptr ds:[<&CreateProcessA>]	JMP.&CreateProcessA
	7530305C	CC	int3	
	7530305D	CC	int3	
	7530305E	CC	int3	

Figure 19 - CreateProcessA call used to spawn a new cmd.exe process.

— — —

```
0066FD3C 77705250 return to msvcrt.77705250 from ???
0066FD40 007A16E0 "C:\\windows\\system32\\cmd.exe"
0066FD44 007A3500 "C:\\windows\\system32\\cmd.exe /c tree C:\\\"
0066FD48 00000000
0066FD4C 00000000
0066FD50 00000001
0066FD54 00000000
0066FD58 00000000
0066FD5C 00000000
```

Figure 20 - Stack arguments containing the command argument to be included when spawning the new cmd.exe process.

— — —

Finally, once the tree command is finished, the sample calls `ExitProcess` to terminate.

EIP	→	752F4B7E	CC	int3	
		752F4B7F	CC	int3	
		752F4B80	55	push ebp	
		752F4B81	8BEC	mov ebp,esp	ExitProcess
		752F4B83	6A FF	push FFFFFFFF	
		752F4B85	68 80F3E877	push 77E8F3B0	
		752F4B8A	FF75 08	push dword ptr ss:[ebp+8]	
		752F4B8D	FF15 381B3576	call dword ptr ds:[<Error! Exit user process>]	
		752F4B89	CC	int3	
		752F4B94	CC	int3	
		752F4B95	CC	int3	

Figure 21 - ExitProcess called after all operations are finished.

— — —

MITRE ATT&CK Techniques

Here a list of all MITRE ATT&CK classifications of the malware sample will be included.

ATT&CK ID	Tactic Name	Description of use
T1059.001	Command and Scripting Interpreter: Powershell	Sample saves and executes a Powershell script.
T1059.003	Command and Scripting Interpreter: windows Command Shell	Sample executes commands via cmd.exe.
T1486	Data Encrypted for Impact	Sample encrypts data on target systems.

YARA Signature

```
rule WANNAHUSKY
{
    meta:
        author = "KrknSec"
        info = "WANNAHUSKY Ransomware"

    strings:
        $s1 = "_NimMain." ascii
        $s2 = "@mwannahusky.nim.c_asgnRef0" ascii
        $s3 = "@Desktop\\ps1.ps1" ascii
        $s4 = "@Desktop\\WANNAHUSKY.png" ascii
        $s5 = "@Desktop\\target\\cosmo.WANNAHUSKY" ascii
        $s6 = "@Desktop\\cosmo.jpeg" ascii

    condition:
        all of them
}
```

Extractor Script

```
import pefile
import binascii

def getRdataSection(file):
    pe_file = pefile.PE(file)
    for section in pe_file.sections:
        if b".rdata" in section.Name:
            return section.get_data()

def main():
    # Get path of executable
    sample = input("Enter path of executable: ")

    # Find the .rdata section
    rdata_section = getRdataSection(sample)

    # Extract the embedded PNG file
    pngFile = rdata_section[4360:]
    fullData = pngFile[0:36839]
    f = open("ransomNote.png", "wb")
    f.write(pngFile)
    f.close()

    # Extract the Ransom target
    target = rdata_section[37071:]
    targetData = target[0:49]
    print("\n[+] Targeted file is: \n" + str(targetData) + "\n")

    # Extract the Powershell Script
    pshell = rdata_section[3753:]
    pshellData = pshell[0:559]
    f = open("extractedPowershell.txt", "wb")
    f.write(pshellData)
    f.close()

main()
```

Indicators of Compromise

- WANNAHUSKY.png
 - MD5: 28b6be9ee7d9fc481b51f077a077191e
 - SHA1: 2897ee9611876a8be3111ca9738e2c0942c5b71e
 - Location: %USERPROFILE%\Desktop\
- Ps1.ps1
 - MD5: 7c1bbff5820495dd9f7a294777a49d33
 - SHA1: 23436dcbe4a79b5a6a4930909f0b6a7bfff2434b1
 - Location: %USERPROFILE%\Desktop\
 - ****Deleted after execution****
- .WANNAHUSKY file extension