Inside Maranhão Stealer: Node.js-Powered InfoStealer Using Reflective DLL Injection

⋮ 9/15/2025

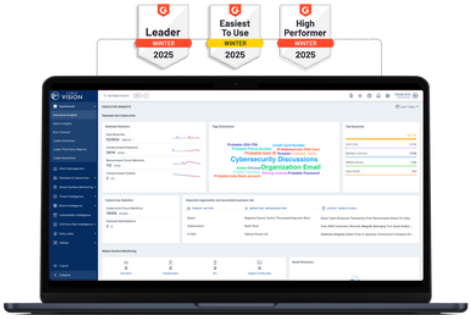
Executive Summary

CRIL identified an active Maranhão Stealer campaign that is being distributed through social engineering websites hosted on cloud platforms. Current intelligence indicates that the malware has been active since May 2025 and is actively being developed.

The threat actors primarily target gaming users by distributing gaming-related links, cheats, and pirated software downloads. (e.g., *hxxps://derelictsgame.in/DerelictSetup.zip*). The ZIP archives include an Inno Setup installer, which launches a Node.js-compiled binary responsible for exfiltrating credentials.

Key takeaways

- Maranhão Stealer is actively spreading through social engineering websites that distribute pirated software, cracked game launchers, and cheats, leveraging cloud-hosted platforms for delivery.
- The malware is written in Node.js and packaged as an Inno Setup installer, reflecting a trend in modern stealer campaigns.
- It establishes persistence through Run registry keys and scheduled tasks, hides its payloads as system and hidden attributes, and performs detailed host reconnaissance, including hardware, network, and geolocation profiling.
- Sensitive information such as credentials, cookies, browsing history, and wallet data is harvested through reflective DLL injection into browsers, bypassing protections like AppBound encryption.
- Exfiltration is carried out to attacker-controlled infrastructure, including multiple maranhaogang[.]fun API endpoints, enabling infection tracking, victim monitoring, and stolen data uploads.



Overview

CRIL identified an active Maranhão Stealer campaign that is being distributed through social engineering websites hosted on cloud platforms. Based on the intel gathered so far, we believe the malware has been active since May 2025 and is still in active stages of development.

The threat actors lure victims by creating gaming-related links, cheats, and pirated software downloads (e.g., *hxxps://derelictsgame.in/DerelictSetup.zip*). The stealer malware is delivered as an Inno Setup installer, which, upon execution, drops a Node.js-compiled binary package.

The depiction of the kill chain is shown below. (see Figure 1)

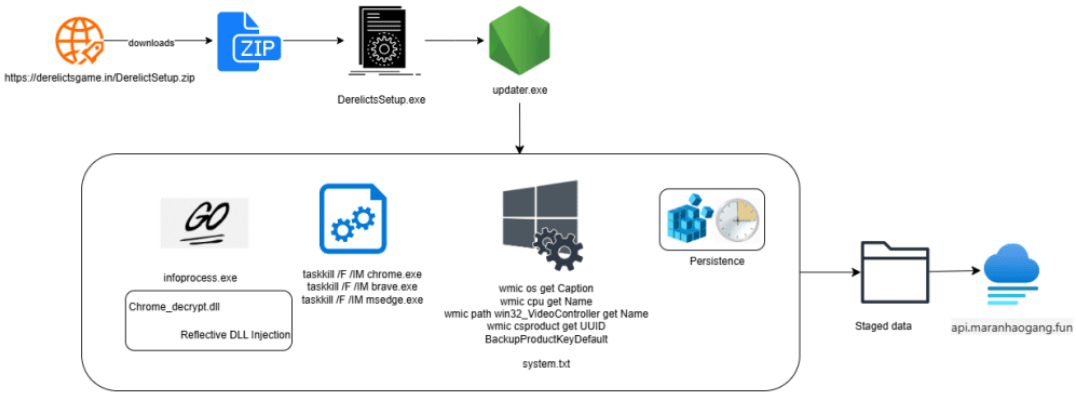


Figure 1 – Infection chain

Once executed, the malware disguises itself in a directory named “Microsoft Updater” located under %localappdata%\Programs. It achieves persistence by creating Run registry keys and a scheduled task before launching its main component, updater.exe. From this point, the malware conducts extensive system reconnaissance, screen capturing, and credential theft, with a particular focus on web browsers and cryptocurrency wallets. To evade security controls such as Chrome’s AppBound encryption, it employs reflective DLL injection into browser processes, enabling reliable access to sensitive data, including cookies, stored credentials, and session tokens.

Stolen artifacts—including credentials, cookies, browser history, and system details—are staged locally before being exfiltrated to attacker-controlled infrastructure, including the domain maranhaogang[.]fun.

The initial variant of Maranhão Stealer, dated back to May 2025, was a simpler build that relied on PsExec to spawn child processes such as taskkill and a Go-based utility named decryptor.exe, which was dropped directly into the C:\Windows directory for plaintext password recovery. Artifacts associated with the group were present in the file details as well. (See Figure 2)

Property	Value
Comments	This installation was built with Inno Setup.
CompanyName	Maranhão Stealer, a Maranhão Gang company
FileDescription	Maranhão Stealer
FileVersion	1.0.0.0
LegalCopyright	© 2025 Maranhão Stealer, a Maranhão Gang company. All right...
OriginalFileName	Maranhão Stealer.exe
ProductName	HauntedSetup.exe

Property	Value
Comments	This installation was built with Inno Setup.
CompanyName	unity company
FileDescription	Unity games
FileVersion	1.0.0.0
LegalCopyright	© 2025 unity.
OriginalFileName	Fnafdoomlauncher.exe
ProductName	Fnaf doom

Figure 2 – Initial variant (left), New variant (right)

In contrast, the newer versions removed traces of these clear artefacts and have shifted to dropping their components under *"C:\Users\MalWorkstation\AppData\Local\Programs\Microsoft Updater"*. The password-decrypting functionality is now embedded in infoprocess.exe, written in Go but obfuscated for stealth. Instead of using PsExec, the malware now creates child processes directly through Win32 API calls, reflecting a clear evolution toward stealthier and more sophisticated execution techniques.

While minor variations have appeared across different Maranhão Stealer samples, the core functionality and operational objectives remain consistent. The campaign demonstrates how threat actors blend social engineering, commodity tools, and modern development stacks to distribute sophisticated information-stealing malware at scale.

Technical Analysis

Infection vector:

The infection vector relies on social engineering through pirated software and gaming-related content. Threat actors distribute trojanized installers, cracked launchers, and cheats, luring users into execution under the guise of popular or modified games. Some examples are listed below:

- Fnafdoomlauncher.exe
- essentiallauncher.exe
- Silent Client.exe
- Fnaf.exe
- clonets.exe
- VersionX64_Setup.exe
- slinky.zip
- Install ROOTED.exe
- RootedTheGameSetup.zip
- Slinkyhook.exe

We performed a technical analysis of a recently identified binary

SHA-1: 97813e1c66dc8922b8242d24a7a56409b57ce19c61042ffda93031c43a358b9b

Filename: Fnafdoomlauncher.exe

Installer: Inno Setup Module (v6.4.3)User Execution

The installer, packaged with Inno Setup, runs in "/VERYSILENT" mode to suppress installation dialogs and reduce user awareness. Once complete, it drops multiple components—updater.exe, crypto.key, and unins000—into the directory *C:\Users\<username>\AppData\Local\Programs\Microsoft Updater*. (See Figure 3)

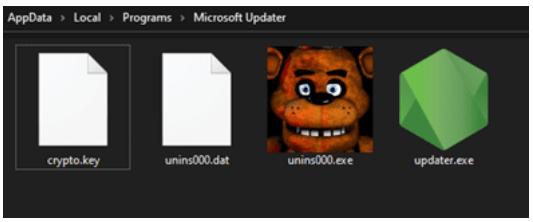


Figure 3 – Install files

The main binary (updater.exe) is then launched with the command-line argument e90de8b2-eb79-4614-94f8-308f0f81573b. This unique identifier, also stored in crypto.key, is used both for victim identification and within the malware's network communications.

Persistence

Upon execution, **updater.exe** establishes persistence by creating a Run registry key via reg.exe, adding an entry that ensures the binary located in the *Microsoft Updater* directory is executed automatically at every user logon. (See Figure 4)

Command: reg.exe ADD HKCU\Software\Microsoft\Windows\CurrentVersion\Run

/v updater /t REG_SZ /d "\"C:\Users\MalWorkstation\AppData\Local\Programs\Microsoft Updater\Updater.exe\"" /f

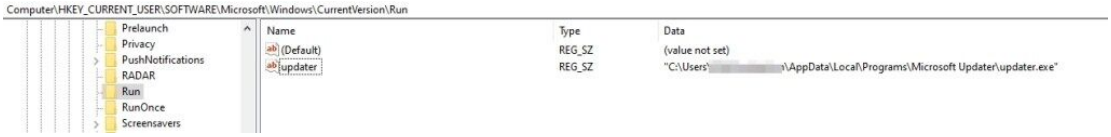


Figure 4 – Persistence through registry

Masquerading, Hidden Files/Directories

Following the persistence setup, the malware attempts to evade detection by disguising its components. Files within the *Microsoft Updater* directory are marked with both the *System* and *Hidden* attributes using attrib.exe, as shown below. (See Figure 5)

- attrib +h +s infoprocess.exe
- attrib +h +s crypto.key
- attrib +h +s "C:\Users\MalWorkstation\AppData\Local\Programs\Microsoft Updater"
- attrib +h +s "C:\Users\MalWorkstation\AppData\Local\Programs\Microsoft Updater\updater.exe"

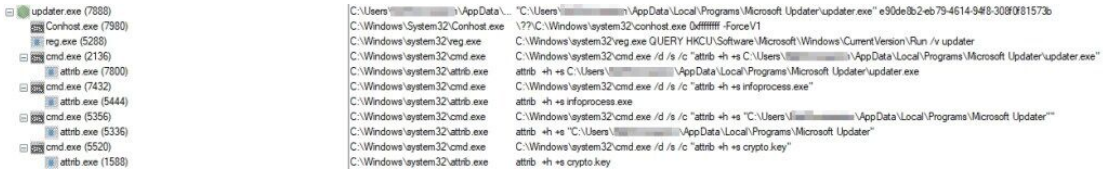


Figure 5 – attrib.exe

System Information Discovery, System Location Discovery

The updater.exe executes a series of WMI queries to enumerate host details, including the operating system version, processor model, graphics controller, hardware UUID, and logical disk information such as size and available free space. This reconnaissance allows the malware to fingerprint the environment, identify potential virtualization or sandboxing, and assess the host's suitability for further exploitation.

- wmic os get Caption
- wmic cpu get Name
- wmic path win32_VideoController get Name
- wmic csproduct get UUID
- wmic logicaldisk get Caption,FreeSpace,Size,Description /format:list

In addition to hardware and system profiling, the malware collects network and geolocation details by sending a request to ip-api.com/json. The response provides information such as the country and country code, region and city, ZIP code, latitude and longitude, time zone, ISP, organization, and ASN. (See Figure 6)

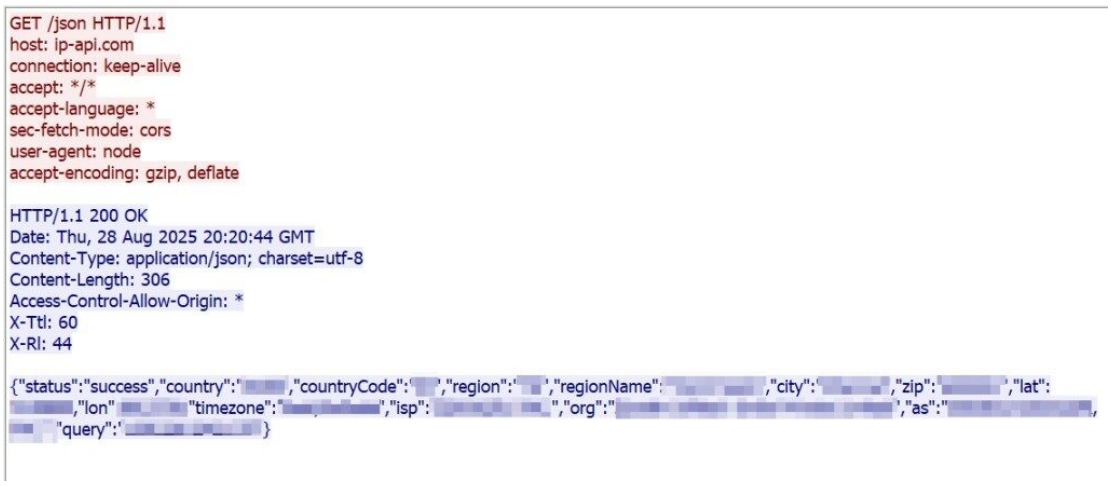


Figure 6 – ip-api.com to collect the victim details

Screen Capture

Continuing its reconnaissance activities, the stealer (updater.exe) also implements screen capture functionality to collect visual information from the victim's environment. It uses inline C# code within PowerShell to enumerate all connected displays (Screen.AllScreens) and capture the contents of each screen (See Figure 7).

```
public class Screenshot
{
    public static List<Bitmap> CaptureScreens()
    {
        var results = new List<Bitmap>();
        var allScreens = Screen.AllScreens;

        foreach (Screen screen in allScreens)
        {
            try
            {
                Rectangle bounds = screen.Bounds;
                using (Bitmap bitmap = new Bitmap(bounds.Width, bounds.Height))
                {
                    using (Graphics graphics = Graphics.FromImage(bitmap))
                    {
                        graphics.CopyFromScreen(new Point(bounds.Left, bounds.Top), Point.Empty, bounds.Size);
                    }

                    results.Add((Bitmap)bitmap.Clone());
                }
            }
            catch (Exception)
            {
                // Handle any exceptions here
            }
        }

        return results;
    }
}
```

Figure 7 – Screen capture

For every detected monitor, the script:

- Determines screen boundaries and resolution.
- Creates a bitmap image of the display.
- Copies pixel data using Graphics.CopyFromScreen.
- Saves the images as sequential PNG files (Display (1).png, Display (2).png, etc.).

This capability allows the threat actor to exfiltrate sensitive information, monitor user activity, and validate the compromise, complementing the system information previously collected.

Credentials from Web Browsers

After completing initial system reconnaissance, the stealer payload (updater.exe) shifts its focus to data theft from web browsers. In our analysis environment, the malware was observed actively collecting data from Google Chrome, Microsoft Edge, Brave, and Opera. For these browsers, it systematically enumerates user profiles and extracts artifacts such as browsing history, cookies, download records, and saved login credentials. (See Figure 8)

```
MOV RAX,QWORD PTR DS:[<&JMP.&CreateFileW>]
XOR R9D,R9D
MOV RCX,QWORD PTR SS:[RSP+50]          [rsp+50]:L"C:\\Users\\[redacted]\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Network\\Cookies"
MOV R8D,R14D
MOV QWORD PTR SS:[RSP+30],R13
MOV EDX,R12D
MOV DWORD PTR SS:[RSP+28],R15D
MOV DWORD PTR SS:[RSP+20],EBP
CALL RAX                               CreateFileW
```

```
48:8B05 19B71900  MOV RAX,QWORD PTR DS:[<&JMP.&CreateFileW>]
45:33C9           XOR R9D,R9D
48:8B4C24 50      MOV RCX,QWORD PTR SS:[RSP+50]          [rsp+50]:L"C:\\Users\\[redacted]\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Web Data"
45:8BC6           MOV R8D,R14D
4C:896C24 30      MOV QWORD PTR SS:[RSP+30],R13
41:8BDA           MOV EDX,R12D
44:897C24 28      MOV DWORD PTR SS:[RSP+28],R15D
896C24 20         MOV DWORD PTR SS:[RSP+20],EBP
FFD0              CALL RAX
48:8BF8           MOV RDI,RAX
48:83F8 FF        CMP RAX,FFFFFFFFFFFFFFFF
```

```
MOV RAX,QWORD PTR DS:[<&JMP.&CreateFileW>]
XOR R9D,R9D
MOV RCX,QWORD PTR SS:[RSP+50]          [rsp+50]:L"C:\\Users\\[redacted]\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
MOV R8D,R14D
MOV QWORD PTR SS:[RSP+30],R13
MOV EDX,R12D
MOV DWORD PTR SS:[RSP+28],R15D
MOV DWORD PTR SS:[RSP+20],EBP
CALL RAX                               CreateFileW
```

```
MOV RAX,QWORD PTR DS:[<&JMP.&CreateFileW>]
XOR R9D,R9D
MOV RCX,QWORD PTR SS:[RSP+50]          [rsp+50]:L"C:\\Users\\[redacted]\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\History"
MOV R8D,R14D
MOV QWORD PTR SS:[RSP+30],R13
MOV EDX,R12D
MOV DWORD PTR SS:[RSP+28],R15D
MOV DWORD PTR SS:[RSP+20],EBP
CALL RAX                               CreateFileW
```

Figure 8 – Stealing browser data

Interestingly, additional targets — including other browsers and cryptocurrency wallets — were identified in memory dump analysis, although they were not directly accessed during execution in our setup.

Category	Applications Targeted
Web Browsers	Google Chrome, Chromium, Mozilla Firefox, Microsoft Edge, Opera, Waterfox, Brave, Pale Moon, Comodo IceDragon, Lunar Client, K-Meleon
Cryptocurrency Wallets	Electrum, Atomic Wallet, Exodus, Coinomi, Guarda, Mercury Wallet, Feather Wallet

This suggests that the malware has broader capabilities and can adapt its behaviour depending on the victim's environment.

Reflective DLL Injection:

The injection chain begins with updater.exe, which spawns a secondary process named infoprocess.exe and passes the targeted browser's name as a parameter (e.g., Chrome, Edge, Brave). The helper process then launches the specified browser in headless mode, allowing the malware to interact with it without displaying a visible browser window. (See Figure 9)

```
  sub_1400012A0(
    v39,
    L"https://google.com --headless --disable-gpu --disable-software-rasterizer --disable-dev-shm-usage --disable-accele"
     "rated-2d-canvas --no-sandbox --disable-setuid-sandbox --disable-extensions --disable-component-extensions-with-ba"
     "ckground-pages --disable-default-apps --mute-audio --no-zygote --disable-backgrounding-occluded-windows --memory-"
     "pressure-off --force-low-power-gpu --disable-logging --log-level=3 --v=0",
    412i64);
  v12 = (WCHAR *)v39;
  if ( *((_QWORD *)&v40 + 1) > 7ui64 )
    v12 = (WCHAR *)v39[0];
  v13 = (const WCHAR *)lpApplicationName;
  if ( lpApplicationName[3] > (LPCWSTR)7 )
    v13 = lpApplicationName[0];
  if ( !CreateProcessW(v13, v12, 0i64, 0i64, 0, 0x8000000u, 0i64, 0i64, &StartupInfo, &ProcessInformation) )
  {
    if ( *((_QWORD *)&v40 + 1) <= 7ui64 )
    {
LABEL_84:
```

Figure 9 – Starts the browser in headless mode

Once the browser is running, infoprocess.exe extracts a malicious module (PAYLOAD_DLL) from its resources and injects it into the browser's memory space (e.g., chrome.exe). This injection is carried out using low-level Windows APIs such as NtAllocateVirtualMemory and NtWriteProcessMemory, which map the DLL into the target process. (Figure 10)

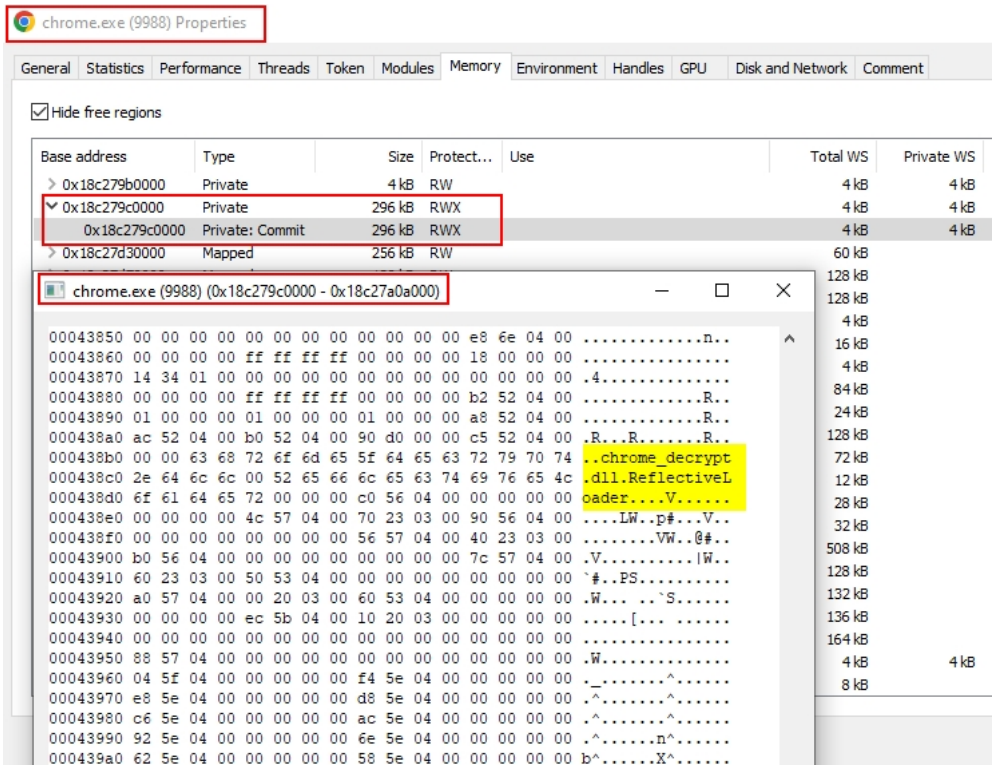


Figure 10 – Reflective Loader in chrome.exe

The injected code is then executed via CreateThreadEx, giving the malware the ability to run inside the browser context. From there, it attempts to retrieve encrypted sensitive information, such as stored credentials and cookies. The stolen data is transmitted back to the calling process over a dedicated named pipe (\\.\pipe\ChromeDecryptIPC_). (See Figure 11)

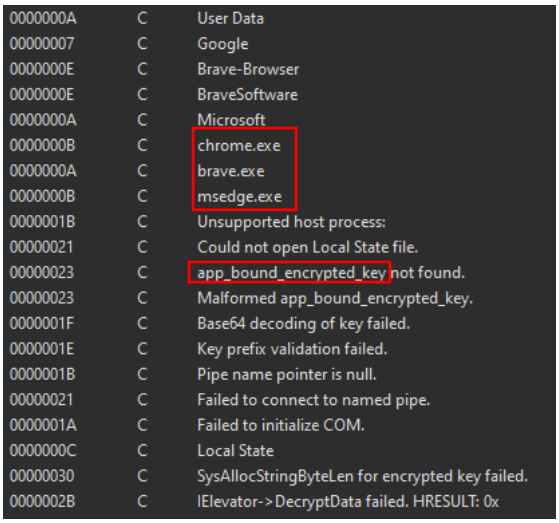


Figure 11 – Contents of reflective loader

As this process completes, the stealer consolidates the harvested browser data and stores it in the %temp% directory, staging it for later exfiltration to the attacker's infrastructure. (See Figure 12)

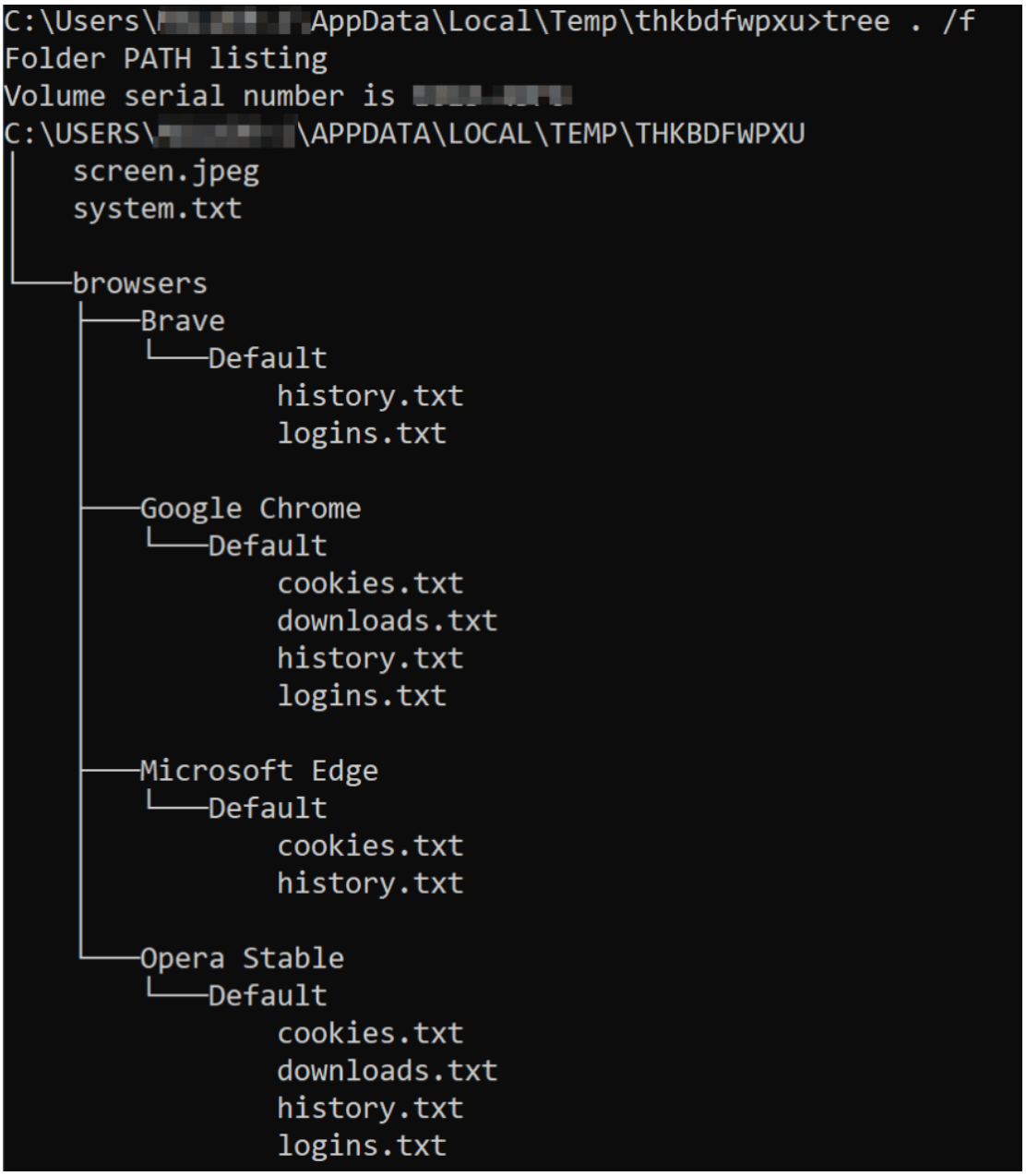


Figure 12 – Stolen data

Command and Control

After gathering system information, screenshots, and sensitive browser data, updater.exe establishes a connection to the attacker-controlled endpoint at 104.234.65.186.

This communication serves as a notification of successful infection. During this phase, the malware transmits key details about the compromised host, including a unique user identifier (derived from crypto.key), the victim's IP address, geographic location (country), and operating system information. (See Figure 13)

Figure 13 – C&C communication

The malware was also observed reaching out to several API's hosted under the domain **maranhaogang[.]fun**, which serves as the attacker panel. (see Figure 14)

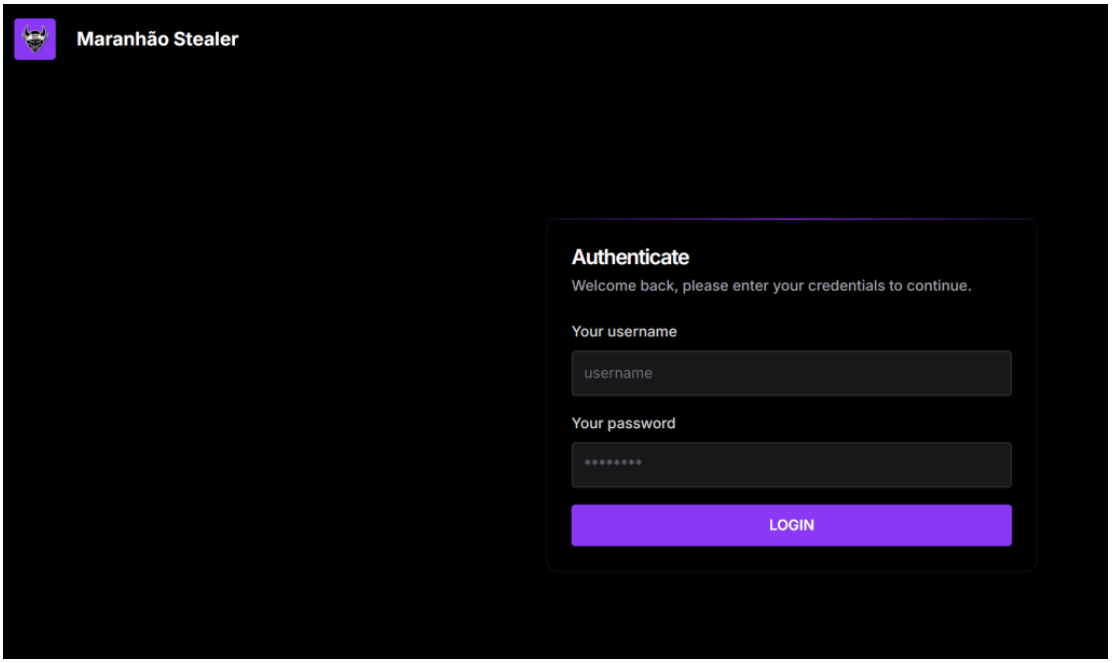


Figure 14 – Attacker panel

The list of URLs identified during our analysis is:

- hxxps://api.maranhaogang.fun/infect
- hxxps://api.maranhaogang.fun/victim
- hxxps://api.maranhaogang.fun/upload

These endpoints appear to serve distinct roles within the attacker's command-and-control (C2) infrastructure, likely handling initial infection reporting, victim tracking, and the exfiltration of stolen data.

Conclusion

The Maranhão Stealer campaign demonstrates threat actors' continued reliance on social engineering via pirated gaming software as an effective infection vector.

Its design clearly emphasizes credential harvesting and cryptocurrency theft, coupled with obfuscation and persistence techniques to evade casual detection. The inclusion of reflective DLL injection and AppBound-aware data collection further underlines its sophistication.

If successful, infections could lead to widespread credential compromise, account hijacking, theft of digital assets, and further malware deployment within victim environments.

Recommendations

- Deploy advanced endpoint detection and response (EDR) solutions to monitor suspicious behaviours such as process injection, registry modifications, and unusual API calls.
- Implement network monitoring to identify unauthorized exfiltration attempts and suspicious outbound traffic to attacker-controlled infrastructure.
- Integrating threat intelligence feeds into firewalls and proxies to strengthen defenses against newly emerging campaigns.
- Restrict the execution of unauthorized binaries using application allowlisting or application control policies.
- Develop and maintain clear incident response (IR) playbooks for malware infections, ensuring security teams can act quickly when a compromise is detected.

MITRE Tactic and Techniques

Tactic	Technique	Procedure Observed
Initial Access (TA0001)	User Execution (T1204.002)	Delivered via trojanized game launchers and pirated software installers.
Persistence (TA0003)	Registry Run Keys/Startup Folder (T1547.001)	Creates Run key via reg.exe to execute Updater.exe at logon.
Privilege Escalation (TA0004)	Process Injection: Dynamic-link Library Injection (T1055.001)	Injects a Reflective DLL payload into the browser's memory
Defense Evasion (TA0005)	Masquerading (T1036)	Places components in the "Microsoft Updater" directory to appear legitimate.
	Hide Artifacts (T1564.001)	Uses attrib +h +s to mark files as hidden/system.
Discovery (TA0007)	System Information Discovery (T1082)	Executes WMI queries (wmic os, wmic cpu, etc.) to profile the host.
	System Location Discovery (T1614.001)	Collects geolocation/network data via ip-api.com.
Collection (TA0009)	Screen Capture (T1113)	Uses inline PowerShell C# to capture screenshots of all connected displays.
Credential Access (TA0006)	Credential from Web Browsers (T1555.003)	Extracts history, cookies, logins, and wallet data from Chrome, Edge, Brave, etc.
Credential Access (TA0006)	Reflective Code Injection (T1620 / T1055.012)	Injects PAYLOAD_DLL into the browser process using NtAllocateVirtualMemory, NtWriteProcessMemory, and CreateThreadEx.
Exfiltration (TA0010)	Exfiltration Over C2 Channel (T1041)	Sends collected data to attacker endpoints (104.234.65.186, maranhaogang[.]fun).
Command and Control (TA0011)	Application Layer Protocol (T1071.001)	Uses HTTP(S) endpoints for infection reporting, victim tracking, and data upload.

Indicators of Compromise (IOCs)

Indicator	Indicator Type	Description
97813e1c66dc8922b8242d24a7a56409b57ce19c61042ffda93031c43a358b9b	SHA-256	Inno Setup file
439eb3631638c61842a20e47e1a31d3c1e917f37688bc3ccdac67dae030117a6	SHA-256	Stealer component
55fc5069e54a35f693bde04f82503752c6dafa5f36c5c35ffbb8ee7c0bd745c6	SHA-256	Passwords decrytor
1c0fb1550b2ac6173c4861fd2a0dd84d0ddcefeb8aeb33b6ba4dc25d9fefaeb6	SHA-256	Fnaf Doom.zip
30dce6d07ea67d4e9dfe848a9245051b26dd3f8c84b9b09a490668d2d01ed715	SHA-256	clonets.zip
5c29934925df4dad85f5930c61b32b738fb1cfc9befd60966208ccb73dbd8db0	SHA-256	Starbirds.zip
b50924f958bb6b49ede6497401dcadc328e3538adf5dca6d66362bcd321a3d00	SHA-256	slinky.zip
d312535b87913542d3f3d0814bb792773c3a2ed561cca43e03892642bf59027a	SHA-256	clonets.zip
ec335c3d2048bb62418526d4d34b386fcad10b8f8805f07d460962ecbd48ab41	SHA-256	RootedTheGameSetup.zip
0080f5a06a9f64019a7d5c7bec4fa390a781be762c2581939bb52135afddb940	SHA-256	Similar Maranhão Stealer file
15fafd21e86ed8a066543d13957e8de14ac68de58d65ec7e8a3b7600c20b9e8e	SHA-256	Similar Maranhão Stealer file
16837d2715bc4afb190c08013ba185b4e62dc65fcbd5320f2dfe6f6be2ca9c27	SHA-256	Similar Maranhão Stealer file

1c0fb1550b2ac6173c4861fd2a0dd84d0ddcefeb8aeb33b6ba4dc25d9fefaeb6	SHA-256	Similar Maranhão Stealer file
299ebbec35850a7a3aaedb743186580fcd4329e2a4cd606560227f817f99557e	SHA-256	Similar Maranhão Stealer file
30dce6d07ea67d4e9dfe848a9245051b26dd3f8c84b9b09a490668d2d01ed715	SHA-256	Similar Maranhão Stealer file
30f4b6d879b7a0a5a817bbfc9bdbcc5171f2000b76c5a90e29a3158cbbe197af	SHA-256	Similar Maranhão Stealer file
393b50b37922fb6dbf183d9b403110f5c4dee18ae5cddd68ca99a38bf84e049f	SHA-256	Similar Maranhão Stealer file
3a71b8f0e4881d8d6888abd7830b4aeede20c7db9687307ae0faa25d53e6002c	SHA-256	Similar Maranhão Stealer file
3ed719b54995c349e6e898064521321961679702407533db8e5552ab97ee46a6	SHA-256	Similar Maranhão Stealer file
4b13407aaf3a4bb239387de96840db6f246f651a010298212b1020c927fa8f96	SHA-256	Similar Maranhão Stealer file
4fdada503206c41d77a5949aee1404c40830d76c4a14c59abea6c235e7a2b9d5	SHA-256	Similar Maranhão Stealer file
5c29934925df4dad85f5930c61b32b738fb1cfc9befd60966208ccb73dbd8db0	SHA-256	Similar Maranhão Stealer file
61c01c3bd2ed568eea8cf9f51de4cabeebecb7db437a46b424ffff6e1d0ca3a4	SHA-256	Similar Maranhão Stealer file
7782f373c32dd2c2017a1cf44b070944fb24add03cc95c6106c2ef4ef01bbc27	SHA-256	Similar Maranhão Stealer file
7eb7103109977c1af4076be0f234160ce356150173b0e536aa97598d4583ef9b	SHA-256	Similar Maranhão Stealer file
863b34c260b9b393f466f99b9199d28a588a2bf4daf83174664fff0b7073093b	SHA-256	Similar Maranhão Stealer file
97813e1c66dc8922b8242d24a7a56409b57ce19c61042ffda93031c43a358b9b	SHA-256	Similar Maranhão Stealer file
97eda27517bb85a0385c4ad6c090a84be38e97998248f4dacfc379b2958209c0	SHA-256	Similar Maranhão Stealer file
9da9d5717b7ee173854a0a4646964415e80b9ec2fa2a0cbe932c0054d5b71362	SHA-256	Similar Maranhão Stealer file
9e6d264b3ab48faf8c89a6e3afb7fe05039bdd82f1fc4af7d3298f9d4337578e	SHA-256	Similar Maranhão Stealer file
a6b68fbdb15945a83bfc84c47f9ee584126f085efac95a89785302134b0a11c0	SHA-256	Similar Maranhão Stealer file
b0973b4a9b8f713a0760e65f717b6fb7b392c2e8e14e07dddfefecb915cca6b2	SHA-256	Similar Maranhão Stealer file
b0a3311f94eb2e87c560b2cde9029a8a5293883777a28fddbf4e4d0672d985f0	SHA-256	Similar Maranhão Stealer file
b50924f958bb6b49ede6497401dcadc328e3538adf5dca6d66362bcd321a3d00	SHA-256	Similar Maranhão Stealer file
c20e72a39a2e4b808bc86dd2a7c88a54c58accbdbe96e405b769f9096b9c97af	SHA-256	Similar Maranhão Stealer file
c8a0cd84d6c8a4d5f7a893744538cbc8b08417468b9c5bd5032b7cdf6d060b34	SHA-256	Similar Maranhão Stealer file
d312535b87913542d3f3d0814bb792773c3a2ed561cca43e03892642bf59027a	SHA-256	Similar Maranhão Stealer file
d45faeb90d706476c2ad52c183c4ca2e2d72fe2bf840d0f38b83193997a2cdde	SHA-256	Similar Maranhão Stealer file
0737f726e751d757e253b0c7aefd697552b075aff9dd661e354c1e87bc132c9a	SHA-256	Similar Maranhão Stealer file
4b13407aaf3a4bb239387de96840db6f246f651a010298212b1020c927fa8f96	SHA-256	Similar Maranhão Stealer file
hxxps://api[.]maranhaogang.fun/infect	URL	Notifies TA about the infection in the victim
hxxps://api[.]maranhaogang.fun:443/socket.io/?id=undefined&EIO=4&transport=	URL	URL found in memory
hxxps://api[.]maranhaogang.fun/victim	URL	URL found in memory
hxxps://api[.]maranhaogang.fun/upload	URL	Uploads the exfiltrated data to TA
api[.]maranhaogang.fun	Domain	Used for api based communication