New Stealit Campaign Abuses Node.js Single Executable Application

Eduardo Altares Eduardo Altares ⋮ ⋮ 10/10/2025

By Eduardo Altares and Joie Salvio | October 10, 2025

Affected Platforms: Microsoft Windows
Impacted Users: Any organization
Impact: Compromised machines are under the control of the threat actor and stolen information can be used for future attacks
Severity Level: Medium

FortiGuard Labs has encountered a new and active Stealit malware campaign that leverages Node.js’ Single Executable Application (SEA) feature to distribute its payloads. This campaign was uncovered following a spike in detections of a particular Visual Basic script, which was later determined to be a component for persistence.

Earlier Stealit campaigns were built using Electron, an open-source framework that packages Node.js scripts as NSIS installers for distribution. This new campaign has adopted Node.js' native Single Executable Application, which similarly bundles scripts and their assets into standalone binaries. Both approaches are effective for distributing Node.js-based malware, as they allow execution without requiring a pre-installed Node.js runtime or additional dependencies.

2025 Global Threat Landscape Report

Use this report to understand the latest attacker tactics, assess your exposure, and prioritize action before the next exploit hits your environment.

Based on the observed filenames, this malware is still being distributed as disguised installers for games and VPN applications, as was the case in previous campaigns. Recent samples we observed are bundled in PyInstaller and common compressed archives and uploaded to file-sharing sites such as Mediafire and Discord.

This blog provides a detailed technical analysis of this new Stealit campaign.

The Stealit Website

Along with the updated malware binary, Stealit has relocated its panel website to new domains. When we first observed this campaign, the panel—also functioning as the Command-and-Control (C2) server—was hosted at stealituptaded[.]lol. However, that domain quickly became inaccessible as the C2 server was moved to iloveanimals[.]shop.

Accessing the panel leads to a commercial website for Stealit, which promotes itself as offering "professional data extraction solutions" through various subscription plans. A dedicated features page outlines its capabilities, highlighting typical remote access trojan (RAT) functionalities such as file extraction, webcam control, live screen monitoring, and ransomware deployment targeting both Android and Microsoft Windows systems. The site also features instructional videos that demonstrate how the service operates on each platform.

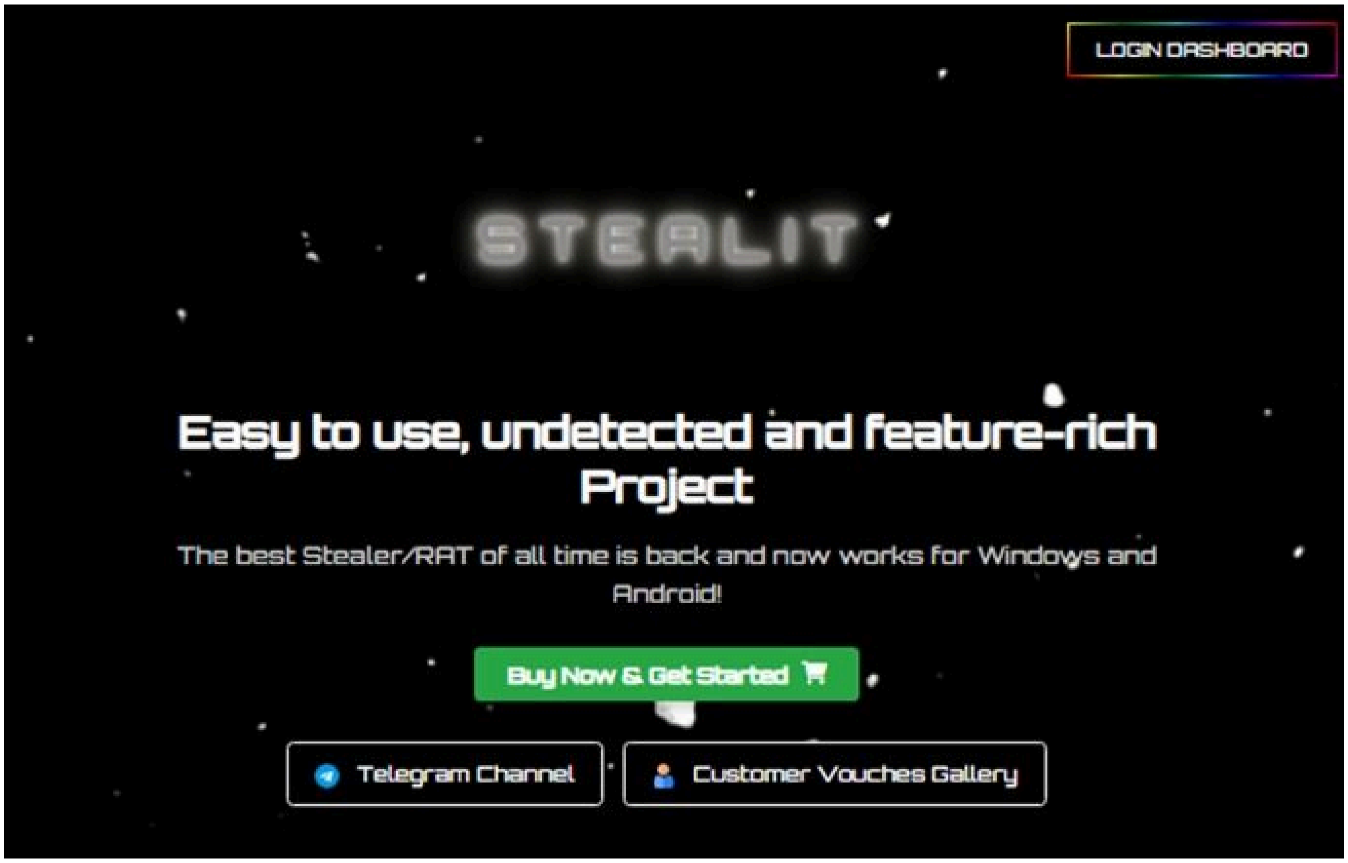
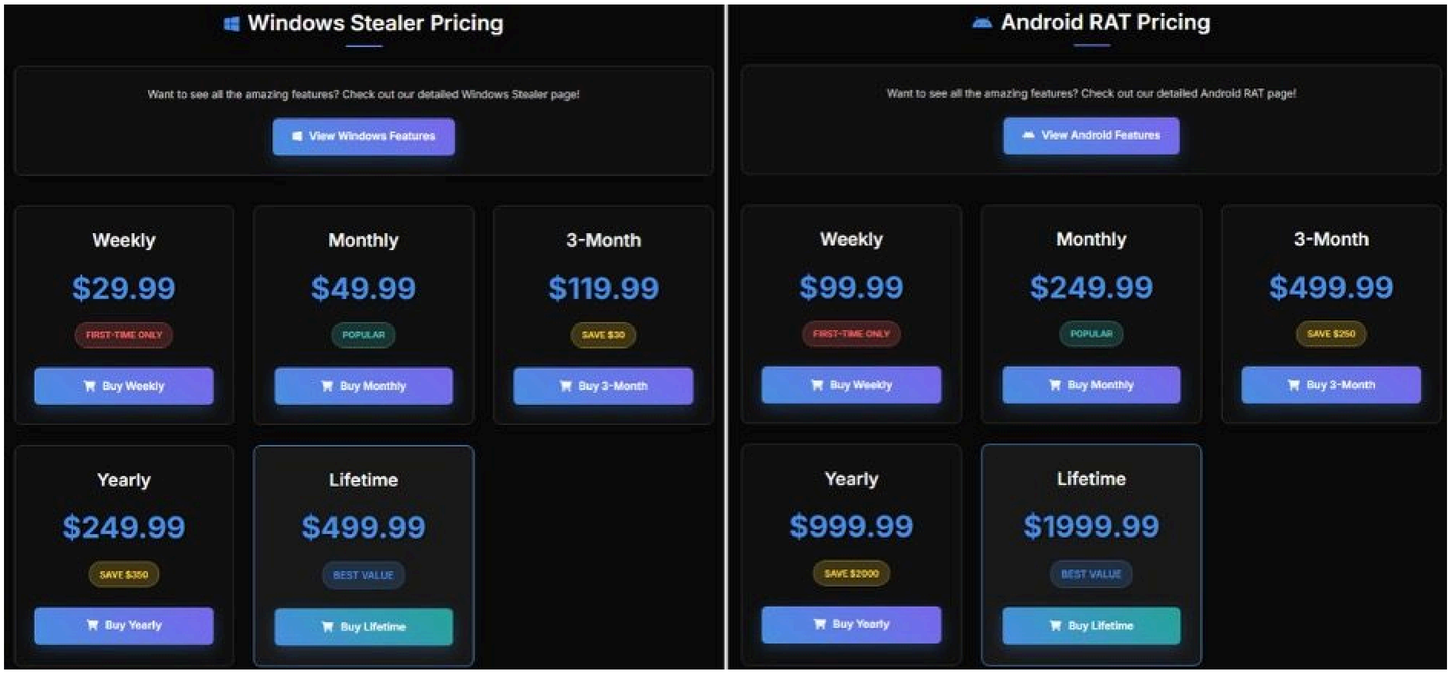

LOGIN DASHBOARD
STEALIT
Easy to use, undetected and feature-rich Project
The best Stealer/RAT of all time is back and now works for Windows and Android!
Buy Now & Get Started
Telegram Channel
Customer Vouches Gallery

Figure 1: Stealit homepage

The website offers payment plans for the Windows and Android versions of the stealer, with lifetime subscriptions available for approximately $ 500 and $ 2,000, respectively.



Windows Stealer Pricing

Want to see all the amazing features? Check out our detailed Windows Stealer page!

Plan	Price	Badge
Weekly	$29.99	FIRST-TIME ONLY
Monthly	$49.99	POPULAR
3-Month	$119.99	SAVE $30
Yearly	$249.99	SAVE $350
Lifetime	$499.99	BEST VALUE

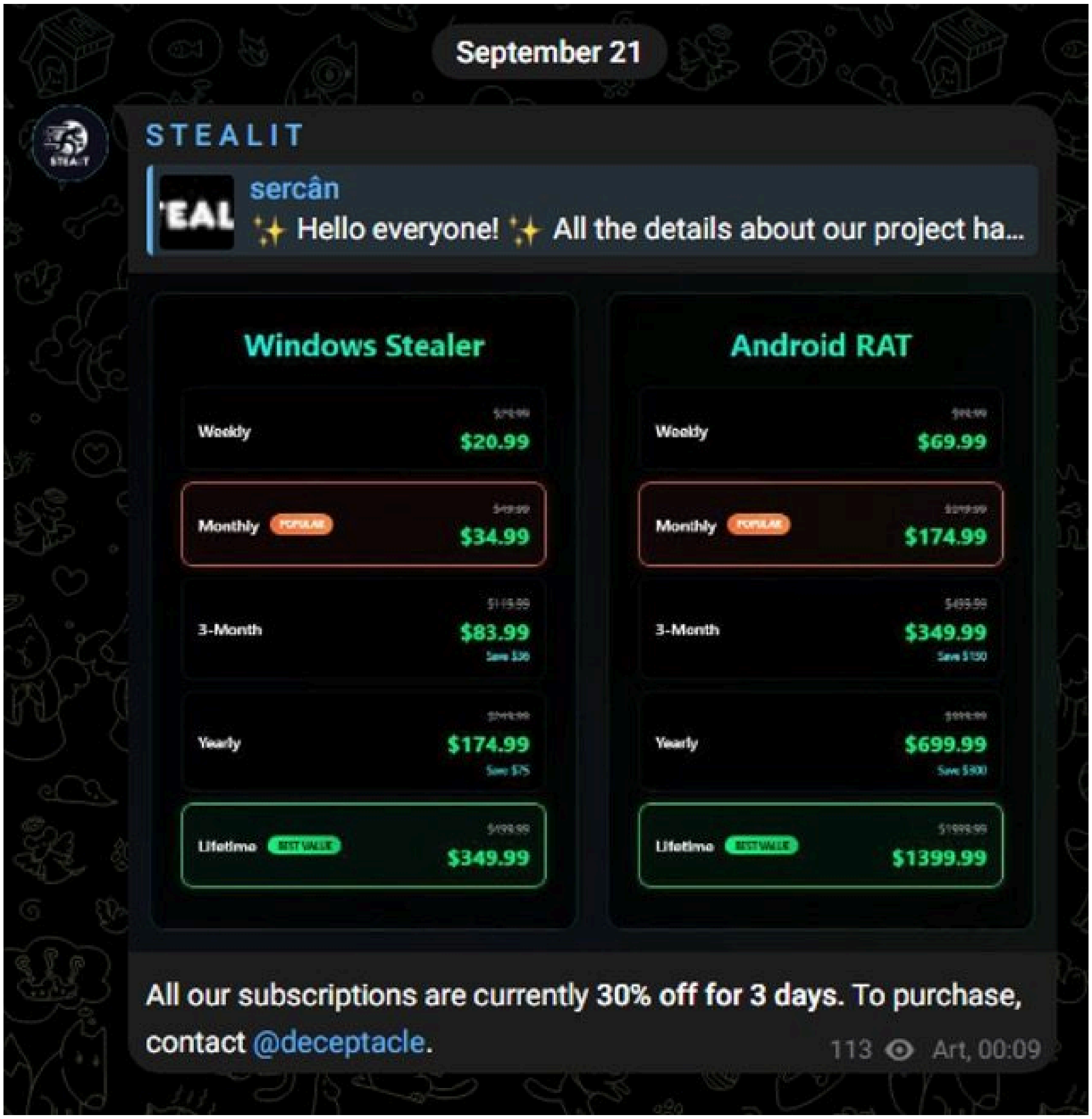
Android RAT Pricing

Want to see all the amazing features? Check out our detailed Android RAT page!

Plan	Price	Badge
Weekly	$99.99	FIRST-TIME ONLY
Monthly	$249.99	POPULAR
3-Month	$499.99	SAVE $250
Yearly	$999.99	SAVE $2000
Lifetime	$1999.99	BEST VALUE

Figure 2: Stealit subscription pricing

The service also has a Telegram channel named *StealitPublic,* where they post updates and promotions to possible clients. The main contact person is a Telegram user with the handle @deceptacle.


September 21

STEALIT

sercân
✨ Hello everyone! ✨ All the details about our project ha...

Windows Stealer		Android RAT	
Weekly	~~$29.99~~ $20.99	Weekly	~~$99.99~~ $69.99
Monthly POPULAR	~~$49.99~~ $34.99	Monthly POPULAR	~~$249.99~~ $174.99
3-Month	~~$119.99~~ $83.99 Save $36	3-Month	~~$499.99~~ $349.99 Save $150
Yearly	~~$249.99~~ $174.99 Save $75	Yearly	~~$999.99~~ $699.99 Save $300
Lifetime BEST VALUE	~~$499.99~~ $349.99	Lifetime BEST VALUE	~~$1999.99~~ $1399.99

All our subscriptions are currently 30% off for 3 days. To purchase, contact @deceptacle.

113 Art, 00:09

Figure 3: A promotional post on StealIt's Telegram channel

Technical Analysis

This Stealit's campaign begins with the installer component that downloads additional components from its C2 server. All of Node.js's scripts bundled in the executables are heavily obfuscated to complicate analysis.

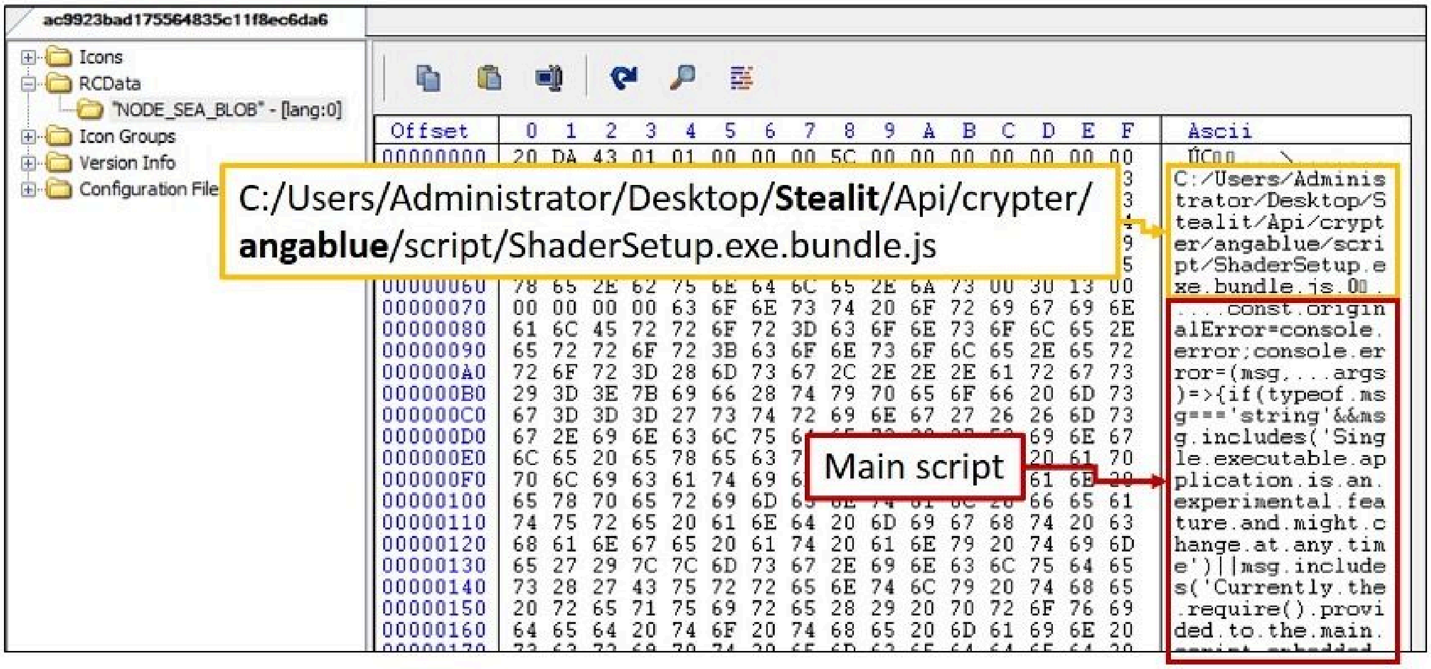
The installer component involves several layers before the actual main installer script is executed.

Installer - First layer

As mentioned earlier, this malware campaign distributes the malicious Node.js scripts using the SEA feature.

Node.js SEA is currently an experimental feature that is being actively developed, with the primary purpose of distributing and running Node.js applications as a single executable binary on systems that do not have Node.js installed. As a tradeoff, since all the required application code, dependencies, and assets need to be built into a single executable file, a simple hello world Node.js SEA for Windows could be 85MB in size.

The malicious installer script to be executed by Node.js is stored as a raw data resource (RCDATA) named NODE_SEA_BLOB. In addition to the main script, that resource data also contains the original path of the script before packaging. In the samples we encountered, this path includes a directory named StealIt, strongly suggesting that it belongs to the stealer malware service of the same name. Moreover, the path also includes angablue, which indicates that it was built using AngaBlue — an open-source project that aims to automate the building of Node.js SEA executables.

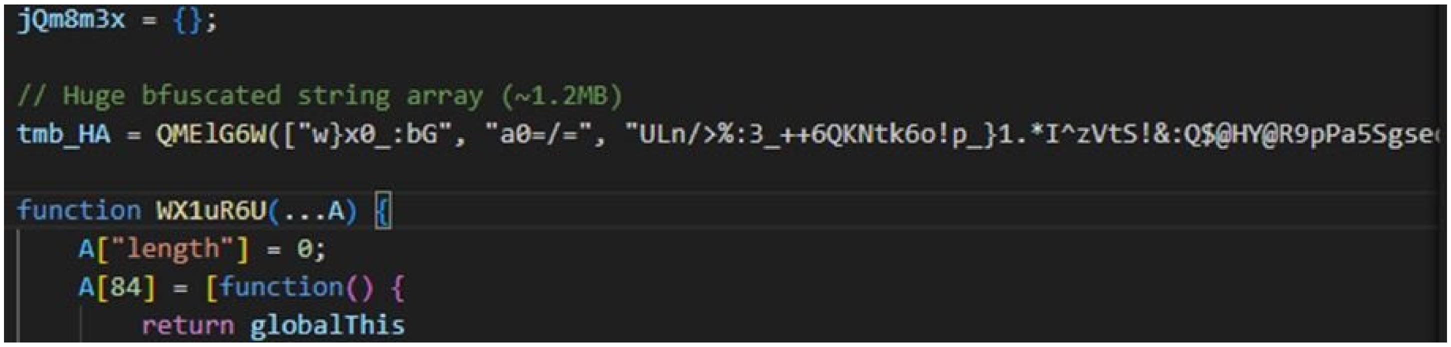


ac9923bad175564835c11f8ec6da6
Icons
RCData
"NODE_SEA_BLOB" - [lang:0]
Icon Groups
Version Info
Configuration File
C:/Users/Administrator/Desktop/**Stealit**/Api/crypter/**angablue**/script/ShaderSetup.exe.bundle.js
Main script

```
Offset     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    Ascii
00000000  20 DA 43 01 01 00 00 00 5C 00 00 00 00 00 00 00    ÚC
                                                              C:/Users/Adminis
                                                              trator/Desktop/S
                                                              tealit/Api/crypt
                                                              er/angablue/scri
                                                              pt/ShaderSetup.e
00000060  78 65 2E 62 75 6E 64 6C 65 2E 6A 73 00 30 13 00    xe.bundle.js.0.
00000070  00 00 00 00 63 6F 6E 73 74 20 6F 72 69 67 69 6E    ....const.origin
00000080  61 6C 45 72 72 6F 72 3D 63 6F 6E 73 6F 6C 65 2E    alError=console.
00000090  65 72 72 6F 72 3B 63 6F 6E 73 6F 6C 65 2E 65 72    error;console.er
000000A0  72 6F 72 3D 28 6D 73 67 2C 2E 2E 2E 61 72 67 73    ror=(msg,...args
000000B0  29 3D 3E 7B 69 66 28 74 79 70 65 6F 66 20 6D 73    )=>{if(typeof.ms
000000C0  67 3D 3D 3D 27 73 74 72 69 6E 67 27 26 26 6D 73    g==='string'&&ms
000000D0  67 2E 69 6E 63 6C 75                      69 6E 67    g.includes('Sing
000000E0  6C 65 20 65 78 65 63                      20 61 70    le.executable.ap
000000F0  70 6C 69 63 61 74 69                      61 6E 20    plication.is.an.
00000100  65 78 70 65 72 69 6D                      66 65 61    experimental.fea
00000110  74 75 72 65 20 61 6E 64 20 6D 69 67 68 74 20 63    ture.and.might.c
00000120  68 61 6E 67 65 20 61 74 20 61 6E 79 20 74 69 6D    hange.at.any.tim
00000130  65 27 29 7C 7C 6D 73 67 2E 69 6E 63 6C 75 64 65    e')||msg.include
00000140  73 28 27 43 75 72 72 65 6E 74 6C 79 20 74 68 65    s('Currently.the
00000150  20 72 65 71 75 69 72 65 28 29 20 70 72 6F 76 69    .require().provi
00000160  64 65 64 20 74 6F 20 74 68 65 20 6D 61 69 6E 20    ded.to.the.main.
```

Figure 4: NODE_SEA_BLOB resource data

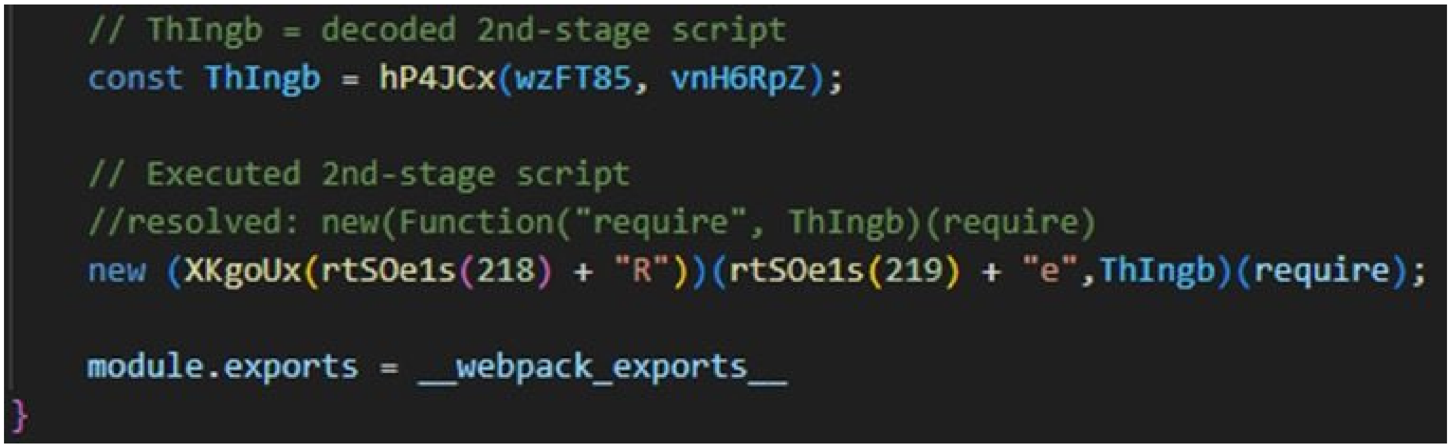
The extracted 1.3MB installer script is heavily obfuscated to complicate analysis. In brief, it contains a large blob (~1.2MB) that is later decoded and executed for the second layer.



```
jQm8m3x = {};

// Huge bfuscated string array (~1.2MB)
tmb_HA = QMElG6W(["w}x0_:bG", "a0=/=", "ULn/>%:3_++6QKNtk6o!p_}1.*I^zVtS!&:Q$@HY@R9pPa5Sgse

function WX1uR6U(...A) {
    A["length"] = 0;
    A[84] = [function() {
        return globalThis
```

Figure 5: Second-layer script as an array of encoded strings

At the end of the script, this blob is decoded and executed directly in memory using Node.js’ require function, which is commonly used to import modules.

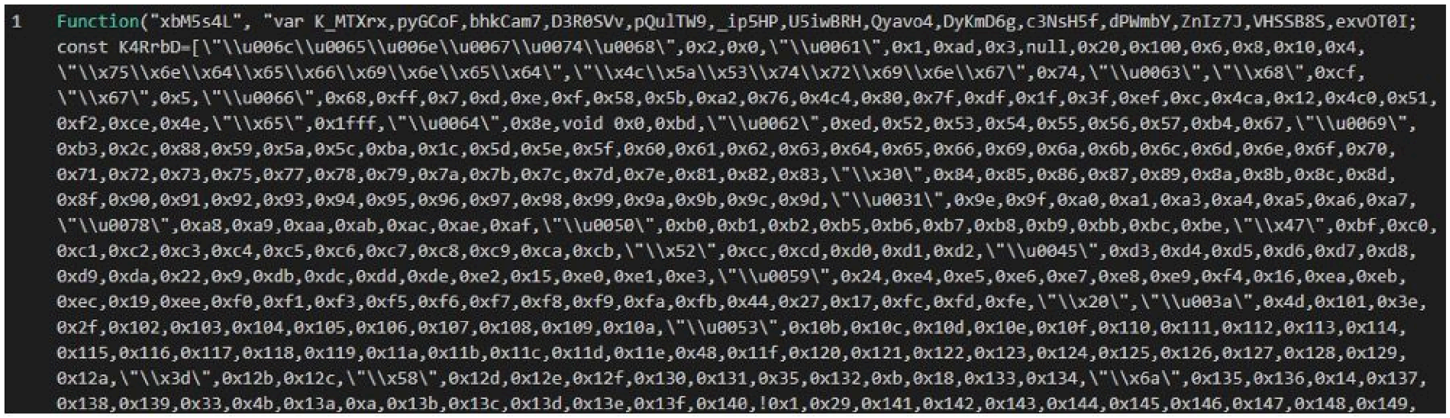


```
    // ThIngb = decoded 2nd-stage script
    const ThIngb = hP4JCx(wzFT85, vnH6RpZ);

    // Executed 2nd-stage script
    //resolved: new(Function("require", ThIngb)(require)
    new (XKgoUx(rtSOe1s(218) + "R"))(rtSOe1s(219) + "e",ThIngb)(require);

    module.exports = __webpack_exports__
}
```

Figure 6: Code snippet for the decoding and execution of the second layer

Installer - Second Layer

Essentially, the second layer is a function object with another large script taken as the second argument. This time, the script is not encoded but is still heavily obfuscated.



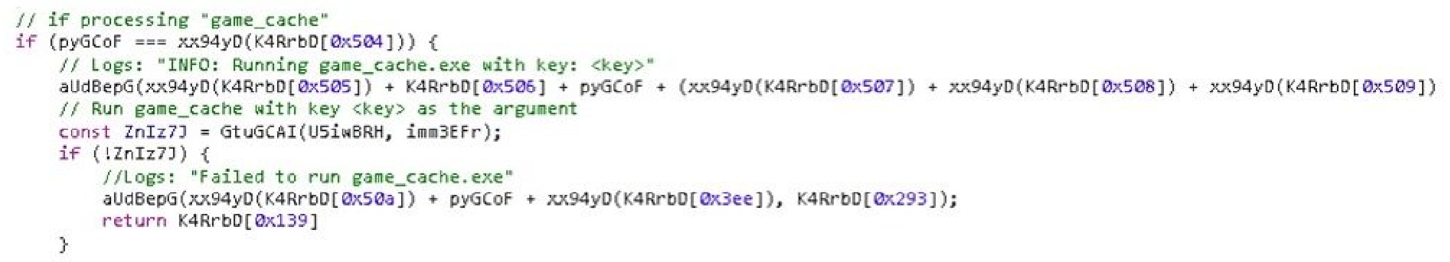
```
Function("xbM5s4L", "var K_MTXrx,pyGCoF,bhkCam7,D3R0SVv,pQulTW9,_ip5HP,U5iwBRH,Qyavo4,DyKmD6g,c3NsH5f,dPWmbY,ZnIz7J,VHSSB8S,exvOT0I;
const K4RrbD=[\"\\u006c\\u0065\\u006e\\u0067\\u0074\\u0068\",0x2,0x0,\"\\u0061\",0x1,0xad,0x3,null,0x20,0x100,0x6,0x8,0x10,0x4,
\"\\x75\\x6e\\x64\\x65\\x66\\x69\\x6e\\x65\\x64\",\"\\x4c\\x5a\\x53\\x74\\x72\\x69\\x6e\\x67\",0x74,\"\\u0063\",\"\\x68\",0xcf,
\"\\x67\",0x5,\"\\u0066\",0x68,0xff,0x7,0xd,0xe,0xf,0x58,0x5b,0xa2,0x76,0x4c4,0x80,0x7f,0xdf,0x1f,0x3f,0xef,0xc,0x4ca,0x12,0x4c0,0x51,
0xf2,0xce,0x4e,\"\\x65\",0x1fff,\"\\u0064\",0x8e,void 0x0,0xbd,\"\\u0062\",0xed,0x52,0x53,0x54,0x55,0x56,0x57,0xb4,0x67,\"\\u0069\",
0xb3,0x2c,0x88,0x59,0x5a,0x5c,0xba,0x1c,0x5d,0x5e,0x5f,0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x69,0x6a,0x6b,0x6c,0x6d,0x6e,0x6f,0x70,
0x71,0x72,0x73,0x75,0x77,0x78,0x79,0x7a,0x7b,0x7c,0x7d,0x7e,0x81,0x82,0x83,\"\\x30\",0x84,0x85,0x86,0x87,0x89,0x8a,0x8b,0x8c,0x8d,
0x8f,0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99,0x9a,0x9b,0x9c,0x9d,\"\\u0031\",0x9e,0x9f,0xa0,0xa1,0xa3,0xa4,0xa5,0xa6,0xa7,
\"\\u0078\",0xa8,0xa9,0xaa,0xab,0xac,0xae,0xaf,\"\\u0050\",0xb0,0xb1,0xb2,0xb5,0xb6,0xb7,0xb8,0xb9,0xbb,0xbc,0xbe,\"\\x47\",0xbf,0xc0,
0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,0xc9,0xca,0xcb,\"\\x52\",0xcc,0xcd,0xd0,0xd1,0xd2,\"\\u0045\",0xd3,0xd4,0xd5,0xd6,0xd7,0xd8,
0xd9,0xda,0x22,0x9,0xdb,0xdc,0xdd,0xde,0xe2,0x15,0xe0,0xe1,0xe3,\"\\u0059\",0x24,0xe4,0xe5,0xe6,0xe7,0xe8,0xe9,0xf4,0x16,0xea,0xeb,
0xec,0x19,0xee,0xf0,0xf1,0xf3,0xf5,0xf6,0xf7,0xf8,0xf9,0xfa,0xfb,0x44,0x27,0x17,0xfc,0xfd,0xfe,\"\\x20\",\"\\u003a\",0x4d,0x101,0x3e,
0x2f,0x102,0x103,0x104,0x105,0x106,0x107,0x108,0x109,0x10a,\"\\u0053\",0x10b,0x10c,0x10d,0x10e,0x10f,0x110,0x111,0x112,0x113,0x114,
0x115,0x116,0x117,0x118,0x119,0x11a,0x11b,0x11c,0x11d,0x11e,0x48,0x11f,0x120,0x121,0x122,0x123,0x124,0x125,0x126,0x127,0x128,0x129,
0x12a,\"\\x3d\",0x12b,0x12c,\"\\x58\",0x12d,0x12e,0x12f,0x130,0x131,0x35,0x132,0xb,0x18,0x133,0x134,\"\\x6a\",0x135,0x136,0x14,0x137,
0x138,0x139,0x33,0x4b,0x13a,0xa,0x13b,0x13c,0x13d,0x13e,0x13f,0x140,!0x1,0x29,0x141,0x142,0x143,0x144,0x145,0x146,0x147,0x148,0x149,
```

The script in the argument, which serves as the third layer, is also executed in memory, following the same method as the second layer.

Installer - Third Layer

This stage performs all the functions to install the main components of the malware in the system.

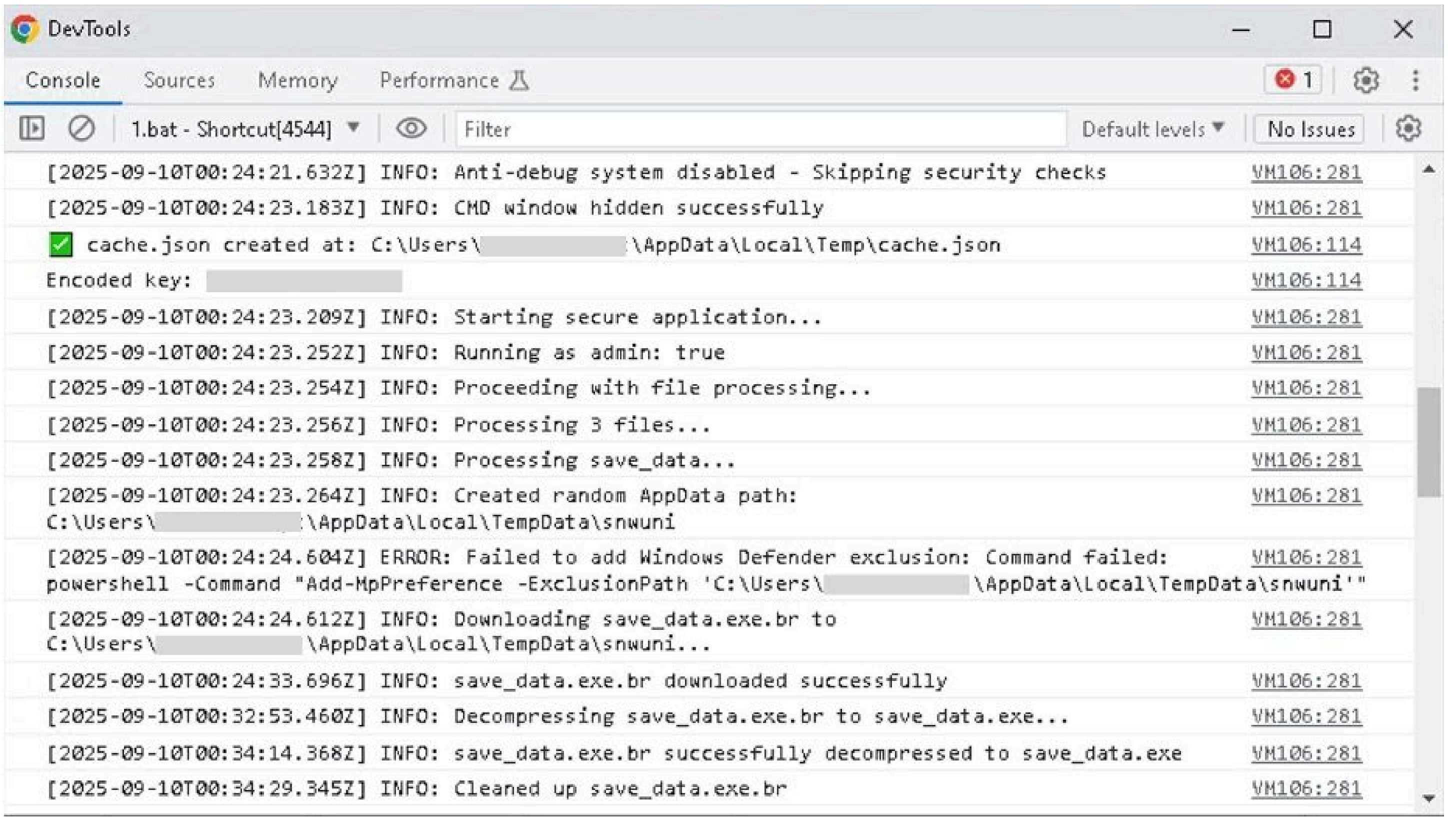
Here is a manually commented code snippet of the third stage, which executes a previously downloaded and decoded component.



```
// if processing "game_cache"
if (pyGCoF === xx94yD(K4RrbD[0x504])) {
    // Logs: "INFO: Running game_cache.exe with key: <key>"
    aUdBepG(xx94yD(K4RrbD[0x505]) + K4RrbD[0x506] + pyGCoF + (xx94yD(K4RrbD[0x507]) + xx94yD(K4RrbD[0x508]) + xx94yD(K4RrbD[0x509])
    // Run game_cache with key <key> as the argument
    const ZnIz7J = GtuGCAI(U5iwBRH, imm3EFr);
    if (!ZnIz7J) {
        //Logs: "Failed to run game_cache.exe"
        aUdBepG(xx94yD(K4RrbD[0x50a]) + pyGCoF + xx94yD(K4RrbD[0x3ee]), K4RrbD[0x293]);
        return K4RrbD[0x139]
    }
```

Figure 7: Commented code snippet of component execution

The variants we encountered log a detailed execution progress that is visible in the console log when they are run under a debugger.



```
[2025-09-10T00:24:21.632Z] INFO: Anti-debug system disabled - Skipping security checks      VM106:281
[2025-09-10T00:24:23.183Z] INFO: CMD window hidden successfully                             VM106:281
✅ cache.json created at: C:\Users\[redacted]\AppData\Local\Temp\cache.json                 VM106:114
Encoded key: [redacted]                                                                      VM106:114
[2025-09-10T00:24:23.209Z] INFO: Starting secure application...                              VM106:281
[2025-09-10T00:24:23.252Z] INFO: Running as admin: true                                      VM106:281
[2025-09-10T00:24:23.254Z] INFO: Proceeding with file processing...                          VM106:281
[2025-09-10T00:24:23.256Z] INFO: Processing 3 files...                                       VM106:281
[2025-09-10T00:24:23.258Z] INFO: Processing save_data...                                     VM106:281
[2025-09-10T00:24:23.264Z] INFO: Created random AppData path:                                VM106:281
C:\Users\[redacted]\AppData\Local\TempData\snwuni
[2025-09-10T00:24:24.604Z] ERROR: Failed to add Windows Defender exclusion: Command failed:  VM106:281
powershell -Command "Add-MpPreference -ExclusionPath 'C:\Users\[redacted]\AppData\Local\TempData\snwuni'"
[2025-09-10T00:24:24.612Z] INFO: Downloading save_data.exe.br to                             VM106:281
C:\Users\[redacted]\AppData\Local\TempData\snwuni...
[2025-09-10T00:24:33.696Z] INFO: save_data.exe.br downloaded successfully                    VM106:281
[2025-09-10T00:32:53.460Z] INFO: Decompressing save_data.exe.br to save_data.exe...          VM106:281
[2025-09-10T00:34:14.368Z] INFO: save_data.exe.br successfully decompressed to save_data.exe VM106:281
[2025-09-10T00:34:29.345Z] INFO: Cleaned up save_data.exe.br                                 VM106:281
```

Figure 8: Malware execution logs under a debugger

In addition to console logging, if this malware is executed with high privileges, it writes the logs to a specific, hardcoded filepath: C:\Users\PC\AppData\Local\errorlx\error.log, creating all the necessary subdirectories beforehand.

If enabled, it performs various checks to detect if it’s running in an analysis environment, categorized based on the type of specific artifacts it’s looking for in the system. These categories are internally referred to as the following:

- **Virtual Environment** – checks if it’s running under a virtual machine by doing a series of tests, including the following:
  - Total system memory must be at least 2GB
  - The number of CPU cores must be at least 2
  - Hostname must not contain keywords from the hardcoded list (See Appendix: Installer - Virtual Environment Hostname Blacklist)
  - Username must not contain keywords from the hardcoded list (See Appendix: Installer - Virtual Environment Username Blacklist).

  If two or more of these checks are not satisfied, it determines that it’s inside a virtual setup.

- **Suspicious Files** – Checks for the existence of directories and files related to VMware and VirtualBox (See Appendix: Installer – Virtual Environment Path Blacklist)

- **Timing Analysis** – Checks execution duration after iterations of some mathematical operations. It determines that it is under analysis if the duration is longer than 500 milliseconds

Advanced Process Detection – This check contains logic bugs that result in inaccurate outcomes. However, the intention is to run the wmic process command with the get Name, ProcessId, ExecutablePath, CommandLine /format:csv option to list all running processes and their execution information in the system. It determines if any of the process information contains the following:

- Contains filenames or paths related to common analysis applications (See Appendix: Installer – Analysis Application Path Blacklist)
- “node.exe” and “debug” as well as other keywords such as –debug-port, --inspect, --debug-brk. These connote a Node.js process running in debug mode.
- Any process with the strings visual studio and code.exe.
- Keywords for arguments related to debugging and/or monitoring (See Appendix: Installer - Process Argument Blacklist).

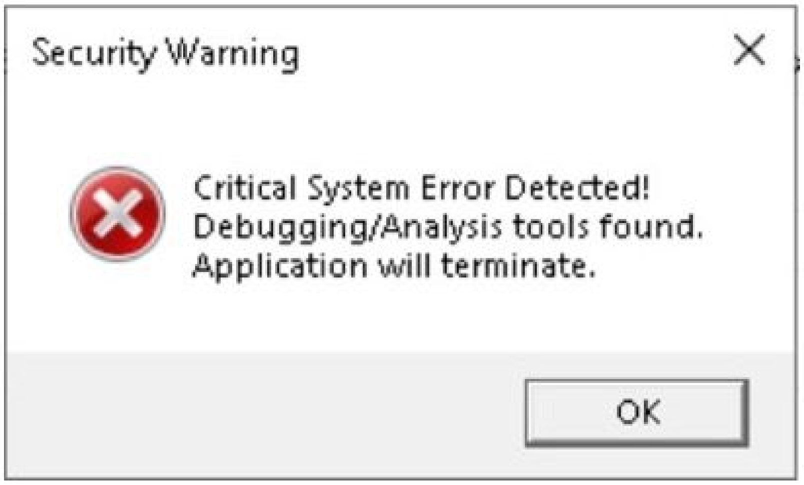
These checks are skipped if the running process’s path and process name are on its whitelist. See Appendix: Installer – Process Path Whitelist and Appendix: Installer – Process Name Whitelist

- **Network Ports** - Runs netstat –an to list all active and listening ports in the system. If the result contains any port in ahardcoded blacklist, it determines it’s being analyzed. In the case of the samples we encountered, this list is empty, practically making this check pointless.
- **Registry Analysis** – Checks if any of the following registry locations contain data related to debugger applications (See Appendix: Installer - Blacklisted Debugger Registry Values)
  - HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AeDebug (value: Debugger)
  - HKLM\SOFTWARE\Classes\exefile\shell\Debug (value: Default)
  - HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run (value: Default)
- **DLL Injection** – Runs the tasklist /m /fi "PID eq {malware process ID} command to list all DLLs that are loaded by the malware process and checks if any of them are related to analysis applications (See Appendix: Installer - Blacklisted Loaded DLLs)
- **Parent Process** – Checks if the process information of its parent process contains one of the blacklisted keywords related to analysis applications (See Appendix: Installer - Blacklisted Parent Process Keywords).

It retrieves the parent process information by running the following command:

wmic process where ProcessId={malware process ID} get Name,CommandLine,ExecutablePath /format:csv

If any of these checks reveal that the malware is being analyzed, it will terminate with the following error message box:



Security Warning

Critical System Error Detected!
Debugging/Analysis tools found.
Application will terminate.

Figure 9: Message box displayed if the anti-analysis check fails

Once it passes through the anti-analysis checks, it proceeds to the actual installation of malware components.

It first writes a base64-encoded authentication key in %temp%\cache.json. This 12-character alphanumeric key is used to authenticate with its C2. This is the same key used by subscribers of the malware service to log in to their dashboards, where they are likely to monitor and control their victims.

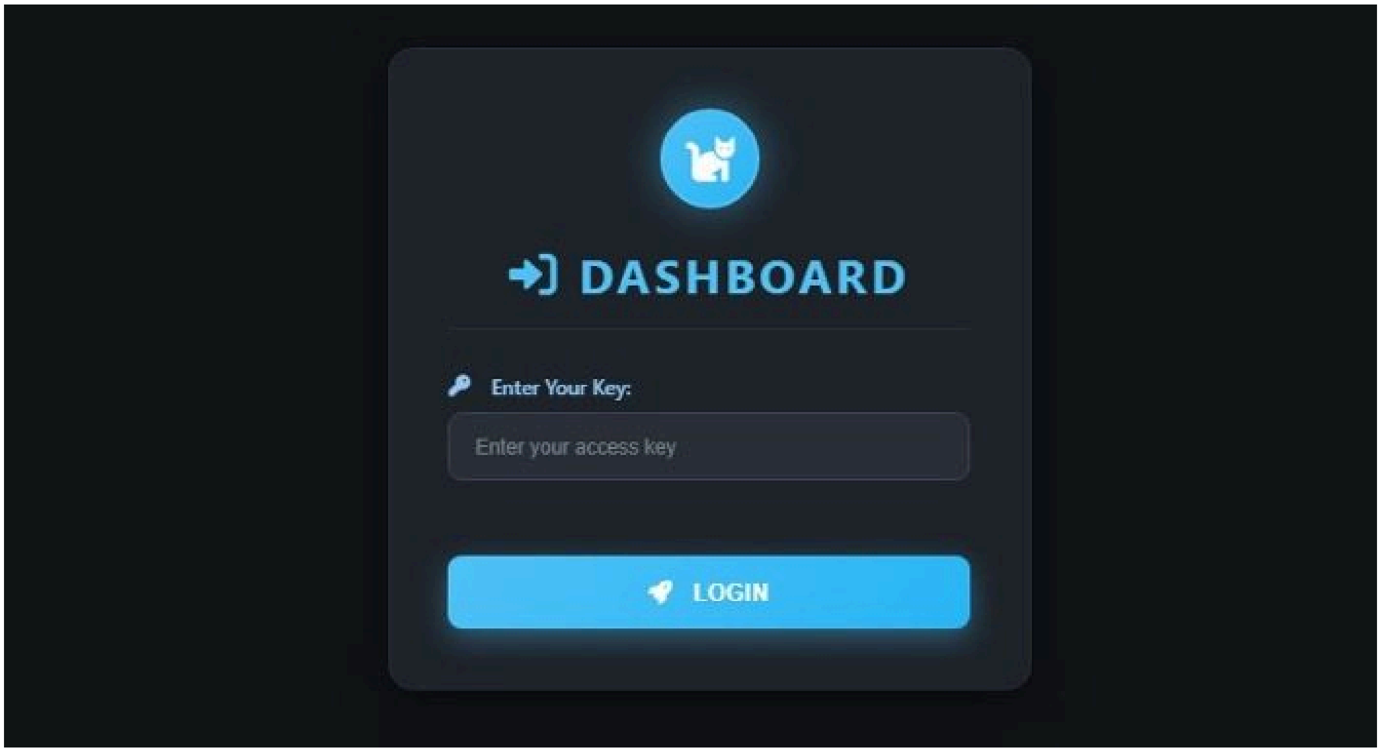


Figure 10: Stealit dashboard login page requiring the authentication key

It then proceeds to download the initial components from the following URLs:

- https[:]//root.iloveanimals[.]shop/download/save_data
- https[:]//root.iloveanimals[.]shop/download/stats_db
- https[:]//root.iloveanimals[.]shop/download/game_cache

The downloaded data are saved to %UserProfile%\Appdata\Local\*{word list}**{random alphanumeric}**{filename}*.exe.br where:

- {word list} - randomly chosen from the following word list:
    - SystemCache
    - WindowsCache
    - AppCache
    - SystemData
    - AppData
    - CacheData
    - SystemTemp
    - AppTemp
    - DataCache
    - TempData
- {*random alphanumeric*} - 2-6 random alphanumeric characters (e.g., vfur1k)
- {filename} - filename from the URL

To prevent Windows Defender from scanning the downloaded files, all the created directories are added to the exemption list by executing the following PowerShell command:

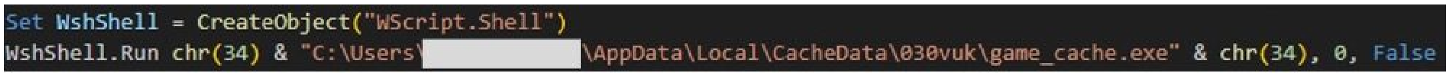
powershell -Command "Add-MpPreference -ExclusionPath {directory}'

The downloaded data are decompressed using the Brotli algorithm and saved in the same directory, following their current filenames without the .br extension. For example, save_data.exe.br is decompressed and saved as save_data.exe.

Similar to the installer, these components are also Node.js scripts packaged as executables. However, while the installer uses Node.js' official SEA to achieve this, these are packaged using an open-sourced project called Pkg. This tool uses a different implementation that involves actual patching of the official Node.js node.exe to support the execution of the embedded NodeJS scripts.

The files are executed with specific arguments immediately after decompression. Analysis of these components are discussed in the next sections.

Once "game_cache.exe" is downloaded and executed, it creates a Visual Basic script, startup.vbs, which launches it. That script is then placed in the Windows startup folder (i.e., %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\startup.vbs) to ensure that the file is automatically executed upon system startup.



```
Set WshShell = CreateObject("WScript.Shell")
WshShell.Run chr(34) & "C:\Users\[redacted]\AppData\Local\CacheData\030vuk\game_cache.exe" & chr(34), 0, False
```

Figure 11: startup.vbs executes game_cache.exe

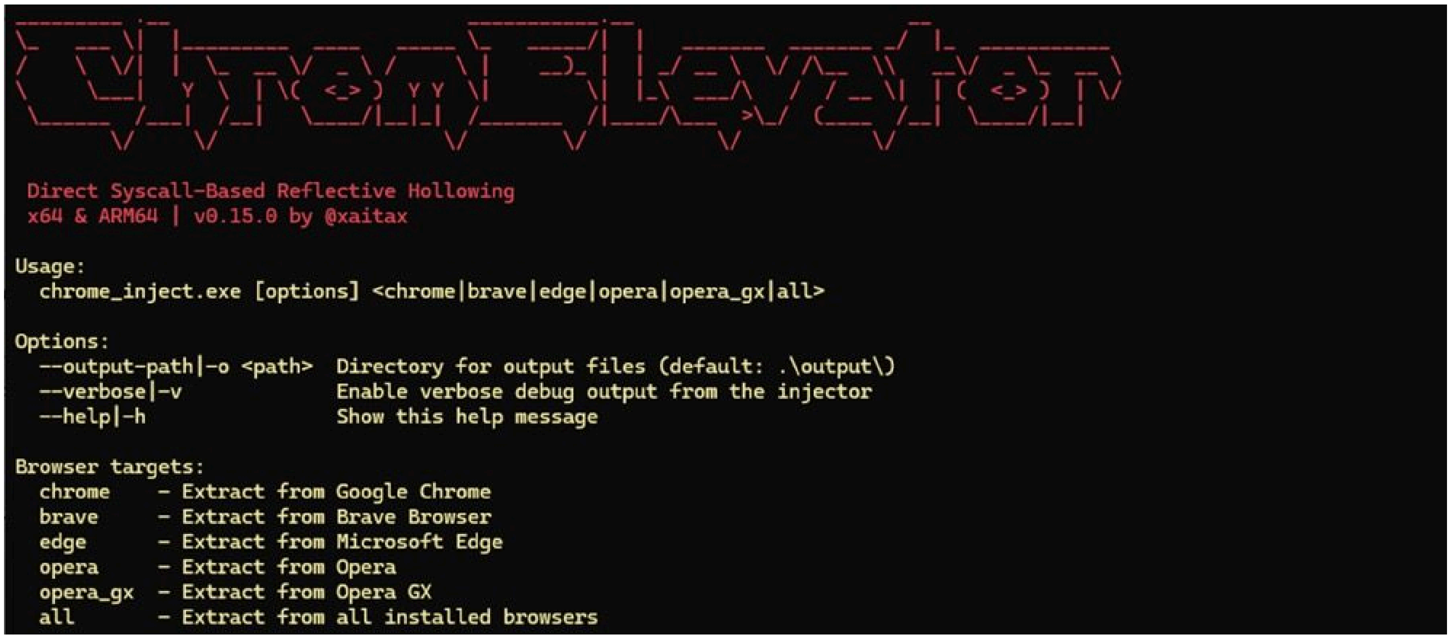
save_data.exe

This component is only downloaded and executed if the malware has been executed with high privileges.

Once this file executes, it drops a file named cache.exe in the same directory and is executed with the following command line:

cache.exe all -o {current directory}\output -v --no-kill

cache.exe is a tool developed from the open-source project ChromElevator, which aims to extract information specifically from Chromium-based browsers. It claims to bypass security features implemented by these browsers, thereby further increasing its chances of success.



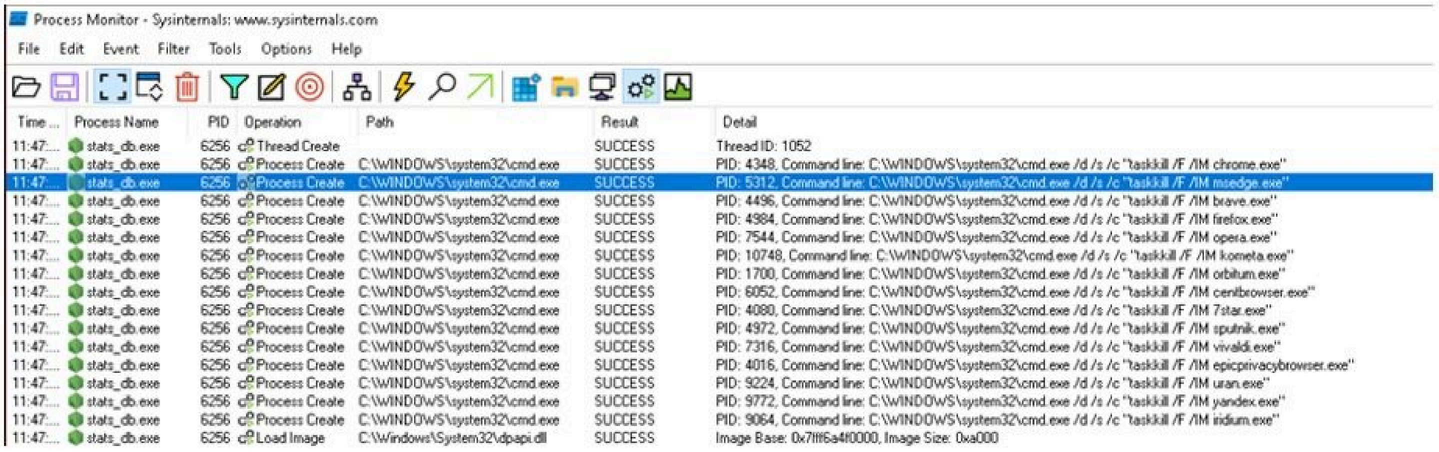
```
 Direct Syscall-Based Reflective Hollowing
 x64 & ARM64 | v0.15.0 by @xaitax

Usage:
  chrome_inject.exe [options] <chrome|brave|edge|opera|opera_gx|all>

Options:
  --output-path|-o <path>  Directory for output files (default: .\output\)
  --verbose|-v             Enable verbose debug output from the injector
  --help|-h                Show this help message

Browser targets:
  chrome     - Extract from Google Chrome
  brave      - Extract from Brave Browser
  edge       - Extract from Microsoft Edge
  opera      - Extract from Opera
  opera_gx   - Extract from Opera GX
  all        - Extract from all installed browsers
```

Figure 12: ChromElevator usage information

stats_db.exe

This component extracts information from a long list of applications.

Before it begins gathering information from target applications, it tries to terminate related processes.



```
Process Monitor - Sysinternals: www.sysinternals.com
File  Edit  Event  Filter  Tools  Options  Help

Time ...  Process Name  PID   Operation       Path                              Result   Detail
11:47:... stats_db.exe  6256  Thread Create                                     SUCCESS  Thread ID: 1052
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 4348, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM chrome.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 5312, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM msedge.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 4496, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM brave.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 4984, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM firefox.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 7544, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM opera.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 10748, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM kometa.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 1700, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM orbitum.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 6052, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM centbrowser.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 4080, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM 7star.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 4972, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM sputnik.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 7316, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM vivaldi.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 4016, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM epicprivacybrowser.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 9224, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM uran.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 9772, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM yandex.exe"
11:47:... stats_db.exe  6256  Process Create  C:\WINDOWS\system32\cmd.exe       SUCCESS  PID: 9064, Command line: C:\WINDOWS\system32\cmd.exe /d /s /c "taskkill /F /IM iridium.exe"
11:47:... stats_db.exe  6256  Load Image      C:\Windows\System32\dpapi.dll     SUCCESS  Image Base: 0x7fff6a4f0000, Image Size: 0xa000
```

Figure 13: Procmon logs show malware trying to terminate targeted applications

It then proceeds to extract information from various browsers, including Google Chrome and Microsoft Edge. For the complete list of target browsers, please see Appendix: Stats_db.exe - Browser Targets**.** The extracted information is stored in the %Temp%/BrowserData directory.

Aside from browsers, it also extracts information from a variety of applications, including:

- **Game-related**
  - Steam
  - Minecraft
  - GrowTopia
  - Epic Games Launcher

- **Messengers**
  - WhatsApp
  - Telegram

- **Cryptocurrency wallets**
  - Atomic
  - Exodus
  - Wallets installed as browser extensions (See Appendix: Stats_db.exe - Browser Target Extension Wallets)

game_cache.exe

This client is responsible for communicating with the command-and-control (C2) server and executing instructions issued by threat actors. It runs using an authentication key passed as an argument.

Initially, this client drops additional components in the victim system in the %Temp% directory, including:

- Encrypted_files.txt: This is likely related to the ransomware module of the malware, in which case, this might contain the list of files that were encrypted
- ScreenCapture_1.3.2.bat**,** app.manifest: These files are used to build a VB.NET application screenCapture_1.3.2.exe inside the victim system. It can be used to remotely view the victim's screen.
- Stealit{random characters}.txt: This contains the authentication key in plaintext.

It initially connects to the C2 by sending the following victim information via an HTTP POST request to https[:]//root[.]iloveanimals[.]shop/panelping in JSON format:

- Username
- HWID (Hardware ID)
- Key (for authentication)

Example:

{"pcName":"[*username*]","hwid":"[*UUID*]","key":"[*12-character key*]"}

From here on, it will be waiting for commands from the C2 server.

Based on the functionalities advertised on Stealit's website, this component can be controlled by the threat actors to perform the following:

Feature	Description
Live Screen View	Stream victim's screen in real-time
Live Webcam Access	View camera feed from victim's device
System Management	Remote shutdown, restart, or control system behavior
Ransom Chat Panel	Communicate directly with the victim
Fake Alert Message	Push custom fake system alerts to the victim
Log Refresh	Retrieve updated logs instantly without needing re-injection
CMD Executor	Send and execute terminal commands live
Remote Audio Player	Play any sound or music on victim's device
EXE Installer + Startup Binder	Upload, execute, and persist any payload
File Grabber	Collect files from Desktop, Documents, Downloads, and other critical paths
Wallpaper Changer	Remotely set any image as victim's desktop wallpaper

Table 1: Stealit features

Return to Electron Framework

Within weeks, new Stealit samples reverted to the Electron framework, this time encrypting bundled Node.js scripts with AES-256-GCM. Functionally, they operate the same as the SEA variant described earlier.

Conclusion

This new Stealit campaign leverages the experimental Node.js Single Executable Application (SEA) feature, which is still under active development, to conveniently distribute malicious scripts to systems without Node.js installed. Threat actors behind this may be exploiting the feature's novelty, relying on the element of surprise, and hoping to catch security applications and malware analysts off guard.

Furthermore, it employs heavy obfuscation and numerous anti-analysis techniques to evade detection and complicate analysis. Once installed, it is capable of controlling the victim's system and extracting information, including login credentials and cryptocurrency wallets, from a wide variety of applications.

This is an active campaign, and in the course of writing this article, we observed shifts in tactics, including switching back to delivering their payload via the Electron framework and relocating their panels to new domains.

FortiGuard will continue monitoring this threat.

Fortinet Protections

FortiGuard AntiVirus detects the malicious files identified in this report as:

- W64/Litseat.A!tr

- W64/Litseat.B!tr
- W64/Litseat.C!tr
- W64/Litseat.D!tr

The FortiGuard Web Filtering Service detects the C2 URLs cited in this report as Malicious and blocks them.

Due to the ease of disruption, damage to daily operations, potential impact on an organization's reputation, and the unwanted destruction or release of PII, among other concerns, it is essential to keep all AV and IPS signatures up to date.

We also recommend that organizations have their end users participate in our free NSE training, NSE 1 – Information Security Awareness. It includes a module on internet threats designed to help end users learn how to identify and protect themselves from various types of phishing attacks.

If you believe these or any other cybersecurity threats have impacted your organization, please contact our Global FortiGuard Incident Response Team.

IOCS

Files

554b318790ad91e330dced927c92974d6c77364ceddfb8c2a2c830d8b58e203c
aa8f0988f1416f6e449b036d5bd1624b793b71d62889afdc4983ee21a1e7ca87
5ea27a10c63d0bbd04dbea5ec08fe0524e794c74d89f92ac6694cfd8df786b1f
083c4e0ffdc9edf0d93655ee4d665c838d2a5431b8064242d93a545bd9ad761b
432b8414113a8c14c0305a562a93ed926e77de351bac235552a59cc02e1e5627
8e1cf254d23e2b94c77294079336339ececf33a3e7ee1a3621ee4e0df0695ce5
919a2107ac27e49cdaa60610706e05edfc99bd3f2e9ca75da4feb6a5f2517c27
e004f8e39e489dec74a13d99836ee5693bd509047ecf49f3fc14efc143a161b5
818350a4fb4146072a25f0467c5c99571c854d58bec30330e7db343bceca008b
8814db9e125d0c2b7489f8c7c3e95adf41f992d4397ed718bda8573cb8fb0e83
24b3def3f374c5f17ec9f1a347c71d9c921155c878ab36e48dd096da418bf782
c38130d7cb43cf3da4858247a751d7b9a3804183db8c4c571b6eede0590474da

URLs

https[:]//iloveanimals[.]shop/
https[:]//iloveanimals[.]shop/user/login
https[:]//root[.]iloveanimals[.]shop/download/save_data
https[:]//root[.]iloveanimals[.]shop/download/stats_db
https[:]//root[.]iloveanimals[.]shop/download/game_cache
https[:]//root[.]iloveanimals[.]shop/panelping
https[:]//root[.]stealituptaded[.]lol/download/save_data
https[:]//root[.]stealituptaded[.]lol/download/stats_db

https[:]//root[.]stealituptaded[.]lol/download/game_cache
https[:]//cdn[.]discordapp[.]com/attachments/1395171942494896190/1413957011837816915/VrchatPlugin.rar?ex=68bdd195&is=68bc8015&hm=b9f359a7f75b84d1b860d2aa4dd92f8adad3a2feef5d82832f49d664a256ff7b&
https[:]//www[.]mediafire[.]com/file/9ni7pgjxuw8pc6h/ShaderSetup.rar/file
Https[:]//download1529[.]mediafire[.]com/8006s55pduvgtQ0THBMZxcLtIrh20a5BnfF18n8YfGUB8P7M5U3mEQb-UYYDCrMHsSG0aWvnyy_LIMg2OnTc4kuNYmWzjWLQwOds-qSfhdO03NOQFAAaYCPiOvB8nU7mBEHe-3a5gDSufW6upPbFXyGlbzBTdtpcrVPXokNKOYZ9/c4zbp39q02jvrn8/Aykadia.rar

Appendix

Installer - Virtual Environment Hostname Blacklist

- vm
- virtual
- sandbox
- malware
- test
- analysis
- vbox
- vmware

Installer - Virtual Environment Username Blacklist

- malware
- virus
- sandbox
- analyst
- test
- vm
- user
- admin

Installer – Virtual Environment Path Blacklist

- C:\Windows\System32\drivers\VBoxMouse.sys
- C:\Windows\System32\drivers\VBoxGuest.sys
- C:\Windows\System32\drivers\vmci.sys
- C:\Windows\System32\drivers\vmmouse.sys
- C:\Program Files\VMware\VMware Tools\
- C:\Program Files\Oracle\VirtualBox\
- C:\Program Files (x86)\VMware\VMware Tools\
- C:\Program Files (x86)\Oracle\VirtualBox\

Installer - Commandline Blacklist

- wireshark
- tcpdump
- tshark
- microsoft network monitor
- ettercap
- fiddler
- capsa
- smartsniff
- omnipeek
- colasoft capsa
- networkminer
- ntop
- ntopng
- packetcapture
- windump
- networx
- glasswire
- x64dbg
- x32dbg
- ollydbg
- immunity debugger
- immunity
- windbg
- visual studio
- ida pro
- ida64
- ida32
- idaw
- idaw64
- radare2
- ghidra
- binary ninja
- binaryninja
- cheat engine
- cheatengine
- artmoney
- game hacker
- gamehacker
- memory scanner
- memoryscanner
- trainer

- burpsuite
- burp suite
- burp-suite
- zaproxy
- zap proxy
- owasp zap
- postman
- insomnia
- curl.exe
- resourcehacker
- resource hacker
- pe-bear
- pebear
- cff explorer
- cffexplorer
- hex editor
- hexeditor
- hxd
- hex workshop
- process monitor
- procmon
- process explorer
- procexp
- api monitor
- apimonitor
- detours
- easyhook
- minhook
- madchook

Installer - Process Path Whitelist

- c:\windows\system32\
- c:\windows\syswow64\
- c:\windows\winsxs\
- c:\program files\windows
- c:\program files (x86)\windows
- c:\program files\microsoft
- c:\program files (x86)\microsoft
- microsoft corporation
- windows defender
- microsoft\
- \windowsapps\

- c:\windows\servicing\
- c:\windows\ccm\
- c:\windows\ccmcache\

Installer – Process Name Whitelist

- svchost.exe
- explorer.exe
- dwm.exe
- winlogon.exe
- csrss.exe
- lsass.exe
- services.exe
- spoolsv.exe
- taskhost.exe
- taskhostw.exe
- conhost.exe
- dllhost.exe
- rundll32.exe
- msiexec.exe
- wininit.exe
- smss.exe
- wermgr.exe
- searchindexer.exe
- audiodg.exe

Installer – Analysis Application Path Blacklist

- \wireshark\
- \fiddler\
- \burp\
- \ida\
- \ghidra\
- \ollydbg\
- \x64dbg\
- \x32dbg\
- \cheat engine\
- \cheatengine\
- \artmoney\
- \processhacker\
- \process hacker\
- \procmon\
- \apimonitor\

- \api monitor\
- \detours\
- \immunity\
- \radare2\
- \binary ninja\
- \binaryninja\
- \resourcehacker\
- \resource hacker\
- \hex workshop\
- \hxd\
- \zaproxy\
- \owasp\\zap\
- \postman\
- \insomnia\

Installer - Process Argument Blacklist

- --debug-port
- --remote-debugging
- --debug-brk
- --inspect
- --trace-warnings
- --trace-uncaught
- --prof
- --debug-only
- /debug
- /trace
- /monitor
- /capture
- /dump
- /hook
- -debug
- -trace
- -monitor
- -capture
- -dump
- -hook

Installer - Blacklisted Debugger Registry Values

- ollydbg
- x64dbg
- x32dbg

- windbg
- ida
- immunity
- ghidra
- radare2
- binary ninja
- cheat engine

Installer - Blacklisted Loaded DLLs

- detours.dll
- easyhook
- minhook
- madchook
- injection.dll
- apimonitor
- deviare
- nektra
- apihook
- dllinjector

Installer - Blacklisted Parent Process Keywords

- x64dbg.exe
- x32dbg.exe
- ollydbg.exe
- ida.exe
- ida64.exe
- ghidra.exe
- immunity.exe
- windbg.exe
- cheatengine.exe
- cheat engine.exe
- processhacker.exe
- apimonitor.exe
- wireshark.exe
- Fiddler.exe

stats_db.exe - Browser Target Extension Wallets

Browser Extension ID	Crypto Wallet
egjidjbpglichdcondbcbdnbeeppgdph	Trust
nkbihfbeogaeaoehlefnkodbefgpgknn	Metamask
hnfanknocfeofbddgcijnmhnfnkdnaad	Coinbase

fhbohimaelbohpjbbldcngcnapndodjp	BinanceChain
bfnaelmomeimhlpmgjnjophhpkkoljpa	Phantom
ibnejdfjmmkpcnlpebklmnkoeoihofec	TronLink
fnjhmkhhmkbjkkabndcnnogagogbneec	Ronin
aholpfdialjgjfhomihkjbmgjidlcdno	Exodus
aeachknmefphepccionboohckonoeemg	Coin98
bhghoamapcdpbohphigoooaddinpkbai	Authenticator
afbcbjpbpfadlkmhmclhkeeodmamcflc	MathWallet
ffnbelfdoeiohenkjibnmadjiehjhajb	YoroiWallet
hpglfhgfnhbgpjdenjgmdgoeiappafln	GuardaWallet
cjelfplplebdjjenllpjcblmjkfcffne	JaxxxLiberty
amkmjjmmflddogmhpjloimipbofnfjih	Wombat
cgeeodpfagjceefieflmdfphplkenlfk	EVERWallet
pdadjkfkgcafgbceimcpbkalnfnepbnk	KardiaChain
hmeobnfnfcmdkdcmlblgagmfpfboieaf	XDEFI
lpfcbjknijpeeillifnkikgncikgfhdo	Nami
aiifbnbfobpmeekipheeijimdpnlpgpp	TerraStation
efbglgofoippbgcjepnhiblaibcnclgk	MartianAptos
nphplpgoakhhjchkkhmiggakijnkhfnd	TON
dmkamcknogkgcdfhhbddcghachkejeap	Keplr
hifafgmccdpekplomjjkcfgodnhcellj	CryptoCom
ejjladinnckdgjemekebdpeokbikhfci	PetraAptos
mcohilncbfahbmgdjkbpemcciiolgcge	OKX
fhmfendgdocmcbmfikdcogofphimnkno	Sollet
epapihdplajcdnnkdeiahlgigofloibg	Sender
opcgpfmipidbgpenhmajoajpbobppdil	Sui
khpkpbbcccdmmclmpigdgddabeilkdpd	SuietSui
jnlgamecbpmbajjfhmmmlhejkemejdma	Braavos
ebfidpplhabeedpnhjnobghokpiioolj	FewchaMove
mcbigmjiafegjnnogedioegffbooigli	EthosSui
dlcobpjiigpikoobohmabehhmhfoodbb	ArgentX
jbdaocneiiinmjbjlgalhcelgbejmnid	NiftyWallet
odbfpeeihdkbihmopkbjmoonfanlbfcl	BraveWallet
blnieiiffboillknjnepogjhkgnoapac	EqualWallet
fihkakfobkmkjojpchpfgcmhfjnmnfpi	BitAppWallet
kncchdigobghenbbaddojjnnaogfppfj	iWallet
fhilaheimglignddkjgofkcbgekhenbh	AtomicWallet
nlbmnnijcnlegkjjpcfjclmcfggfefdm	MewCx
nanjmdknhkinifnkgdcggcfnhdaammmj	GuildWallet
nkddgncdjgjfcddamfgcmfnlhccnimig	SaturnWallet
fnnegphlobjdpkhecapkijjdkgcjhkib	HarmonyWallet
mgffkfbidihjpoaomajlbgchddlicgpn	PaliWallet
aodkkagnadcbobfpggfnjeongemjbjca	BoltX
kpfopkelmapcoipemfendmdcghnegimn	LiqualityWallet
dngmlblcodfobpdpecaadgfbcggfjfnm	MaiarDeFiWallet
ookjlbkiijinhpmnjffcofjonbfbgaoc	TempleWallet
ejbalbakoplchlghecdalmeeeajnimhm	Metamask
kjmoohlgokccodicjjfebfomlbljgfhk	Ronin
akoiaibnepcedcplijmiamnaigbepmcb	Yoroi
ocglkepbibnalbgmbachknglpdipeoio	Authenticator
djclckkglechooblngghdinmeemkbgci	MetaMask

stats_db.exe - Browser Targets

- Brave-Browser
- Google Chrome
- Microsoft Edge
- Opera
- VIvaldi
- Yandex
- Amigo
- Toch
- Kometa
- Orbitum
- CentBrowser
- 7Star
- Sputnik
- Epic Privacy Browser
- UCozMedia Uran
- Iridium
- Mozilla Firefox
- Chromium
- Thorium
- Google Chrome Canary
- ChromePlus
- Chedot
- Elements Browser
- Sleipnir
- Citrio
- Coowon
- LieBao
- QIP Surf
- Comodo Dragon
- 360Browser
- Maxthon3
- K-Melon
- CocCoc
- UR Browser
- Slimjet
- DCBrowser