eSentire Threat Intelligence Malware Analysis: Aurora Stealer

By eSentire Threat Response Unit (TRU)

Archived: 2026-04-06 01:04:42 UTC

Since December 2022, the eSentire Threat Response Unit (TRU) has observed Aurora Stealer malware infections in the manufacturing industry. It's distributed via fake Google Ads for Notepad++ installer. Aurora Stealer gathers sensitive data, including cookies, autofill information, and encrypted passwords from browsers such as Opera, Brave, Mozilla Firefox, Chrome, etc. However, it is worth noting that the stealer does not collect credentials from Mozilla Firefox.

The malware is priced at $125USD per month, $300USD for 3 months access, and $1,000USD for lifetime access. In order to avoid detection from antivirus scanners, the binary code is filled with junk bytes to increase the file size.

This malware analysis delves deeper into the technical details of how the Aurora Stealer malware operates and our security recommendations to protect your organization from being exploited.

Key Takeaways

- The Aurora Stealer developer is actively working on the Aurora botnet, which includes various modules such as the loader, DDoS (distributed denial-of-service), crypto wallet brute-force, HVNC/HRDP/RDP/VNC, Nmap scanner.
- Aurora Stealer stores its configurations in base64-encoded format.
- The stealer logs are sent to a C2 via a default port 8081 in a GZIP-compressed, base64-encoded, JSON format.
- Aurora Stealer is equipped with grabber and loader modules that allow it to collect specific files and folders, as well as introduce additional malware onto a system.

Case Study Aurora Stealer

Drive-by downloads are becoming increasingly common as attackers find new ways to access and exfiltrate sensitive data.

Since December 2022, the eSentire Threat Response Unit (TRU) has observed several Aurora Stealer infections in the manufacturing industries. The stealer is distributed via Google Ads as a fake Notepad++ installer, TeamViewer, Nvidia Driver, etc. (Figures 1-3)

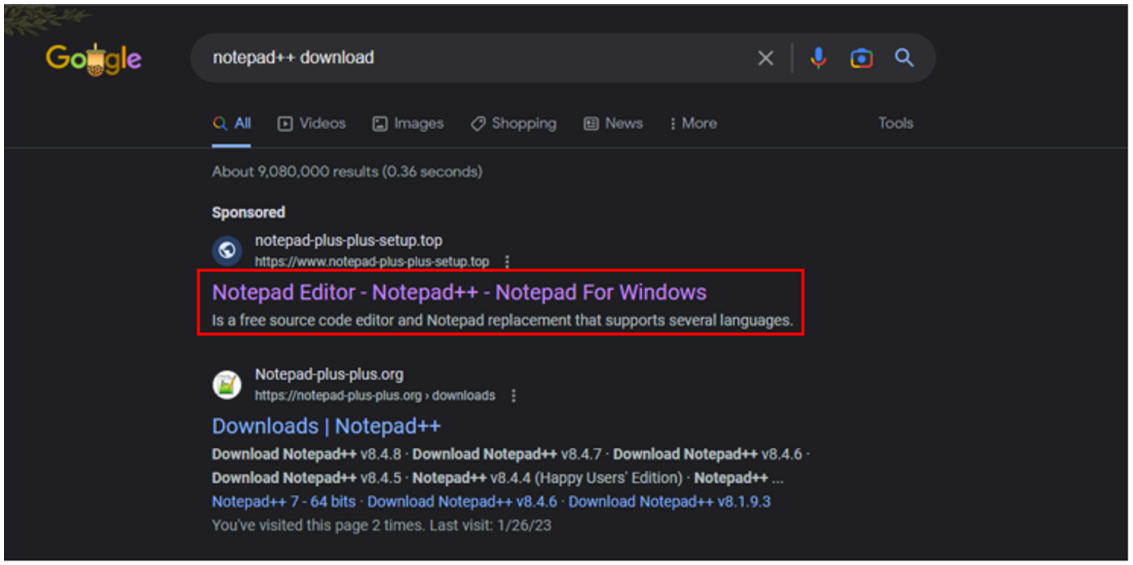


Figure 1: Malicious Google Ads

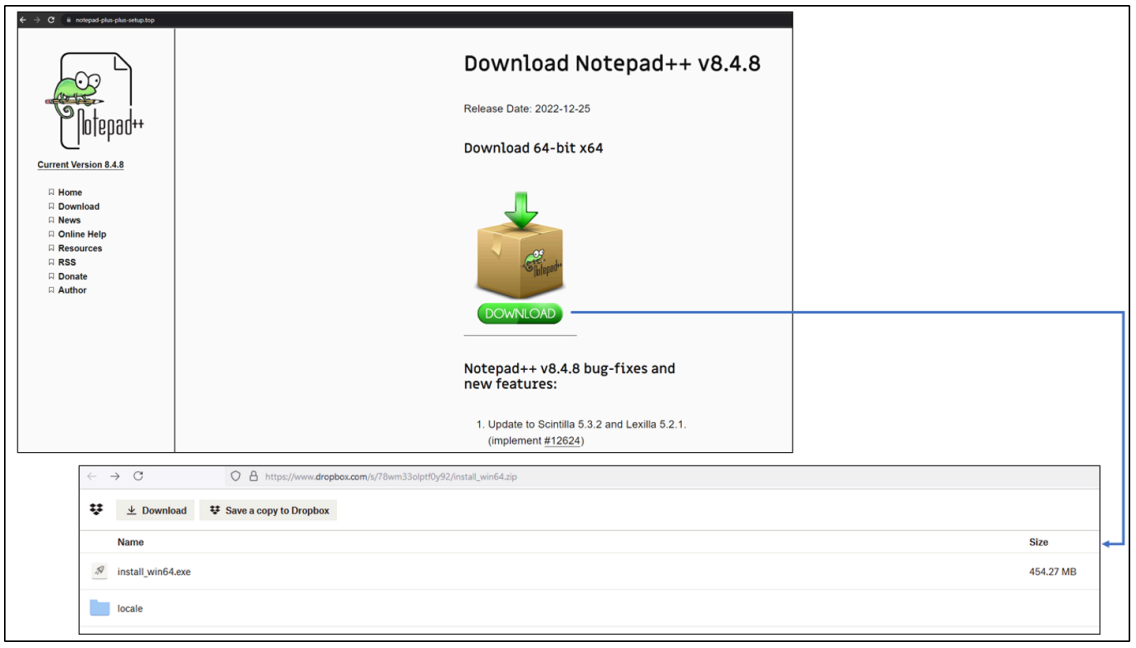


Figure 2: Fake Notepad++ page distributing Aurora Stealer

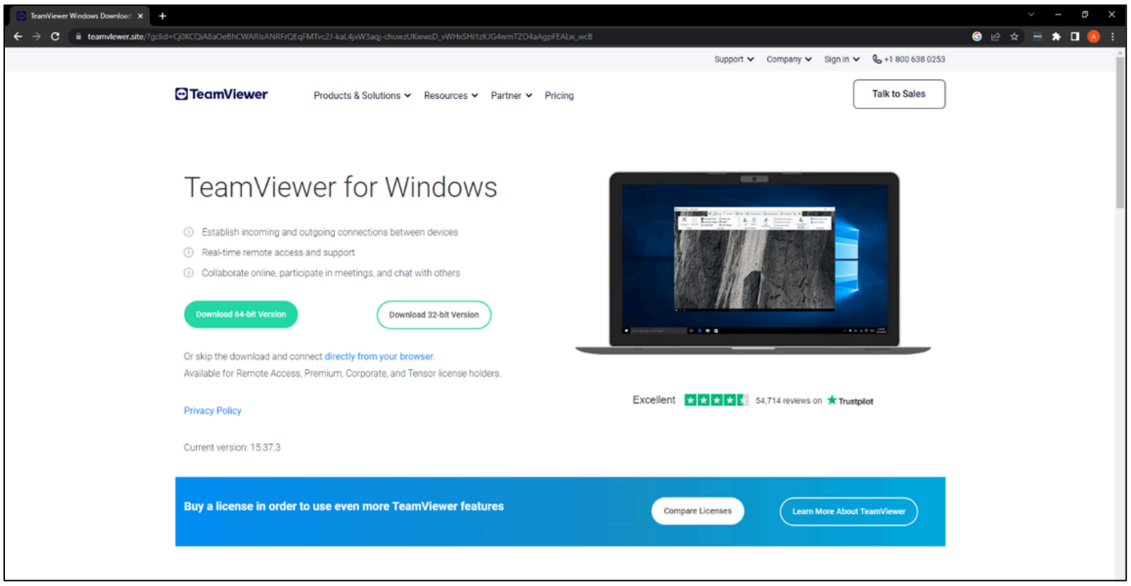


Figure 3: Fake TeamViewer downloader page distributing Aurora Stealer

The stealer uses the Cheshire cat from Alice in Wonderland as its mascot and began appearing for sale on Russian-speaking forums in early 2022. The stealer is written in Golang, capable of stealing over 90 crypto wallets, and has an embedded Loader module that includes the downloader and PowerShell; the developer claims that the stealer does not need any dependencies to function.

The stealer also has a web panel, which does not require the operator to work directly from the dedicated server called “dedik” as slang in Russian hacking forums (Russian: дедик). The dedicated server is the one hosting the stealer and processing the logs. The stealer is priced $125 for one month of access, $300 for 3 months of access, and $1000 for lifetime access. The stealer does not work in Russia and CIS (Commonwealth of Independent States) countries (Figures 4-6).

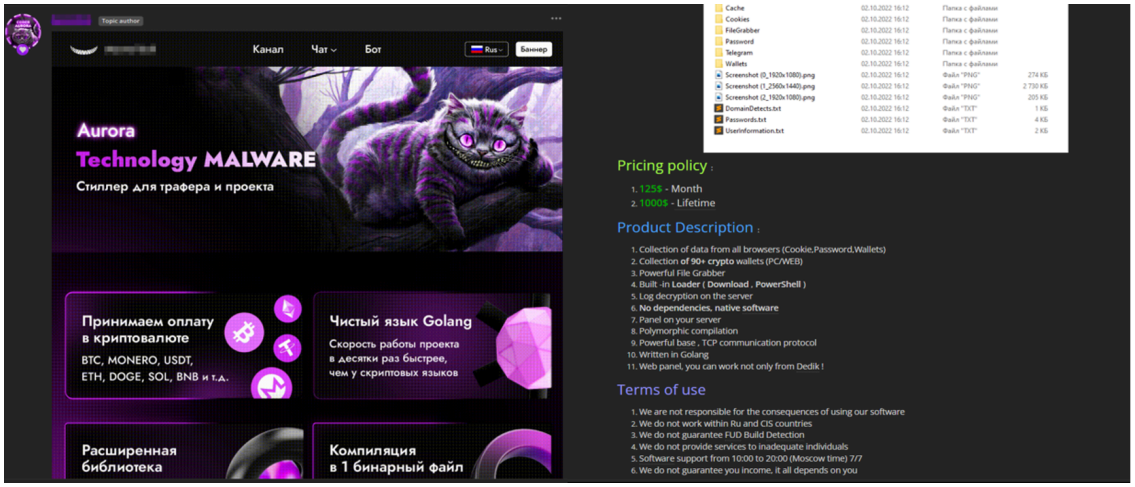


Figure 4: Aurora Stealer seller on the Russian-speaking forum

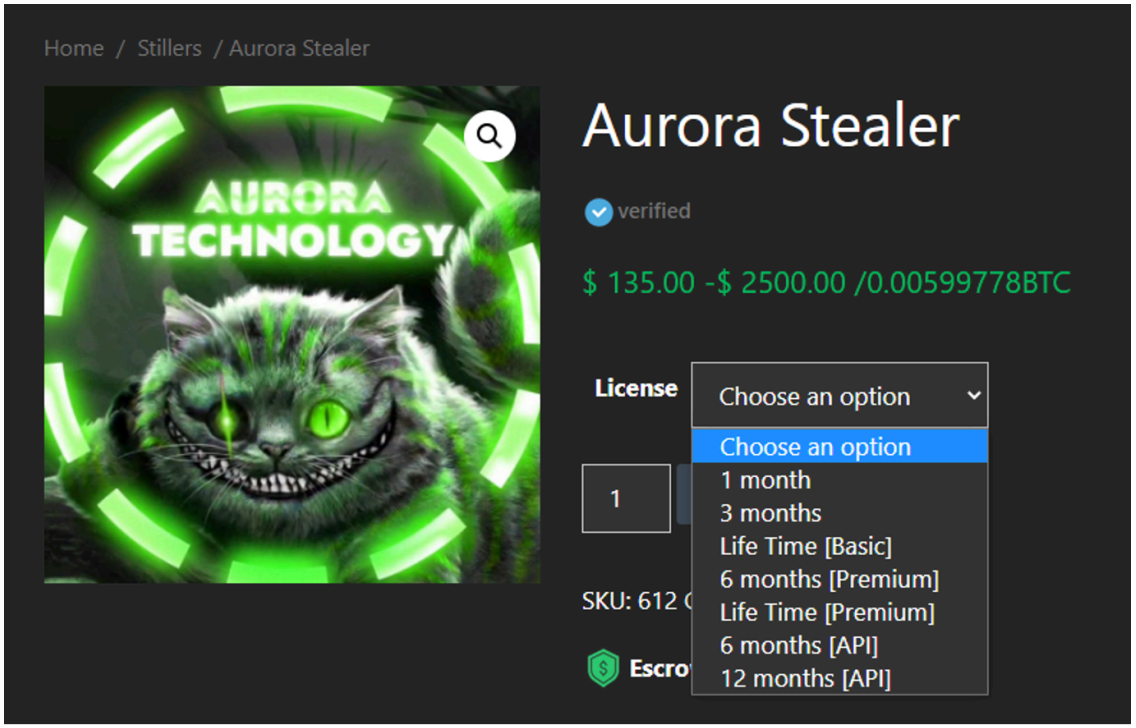


Figure 5: Aurora Stealer reseller

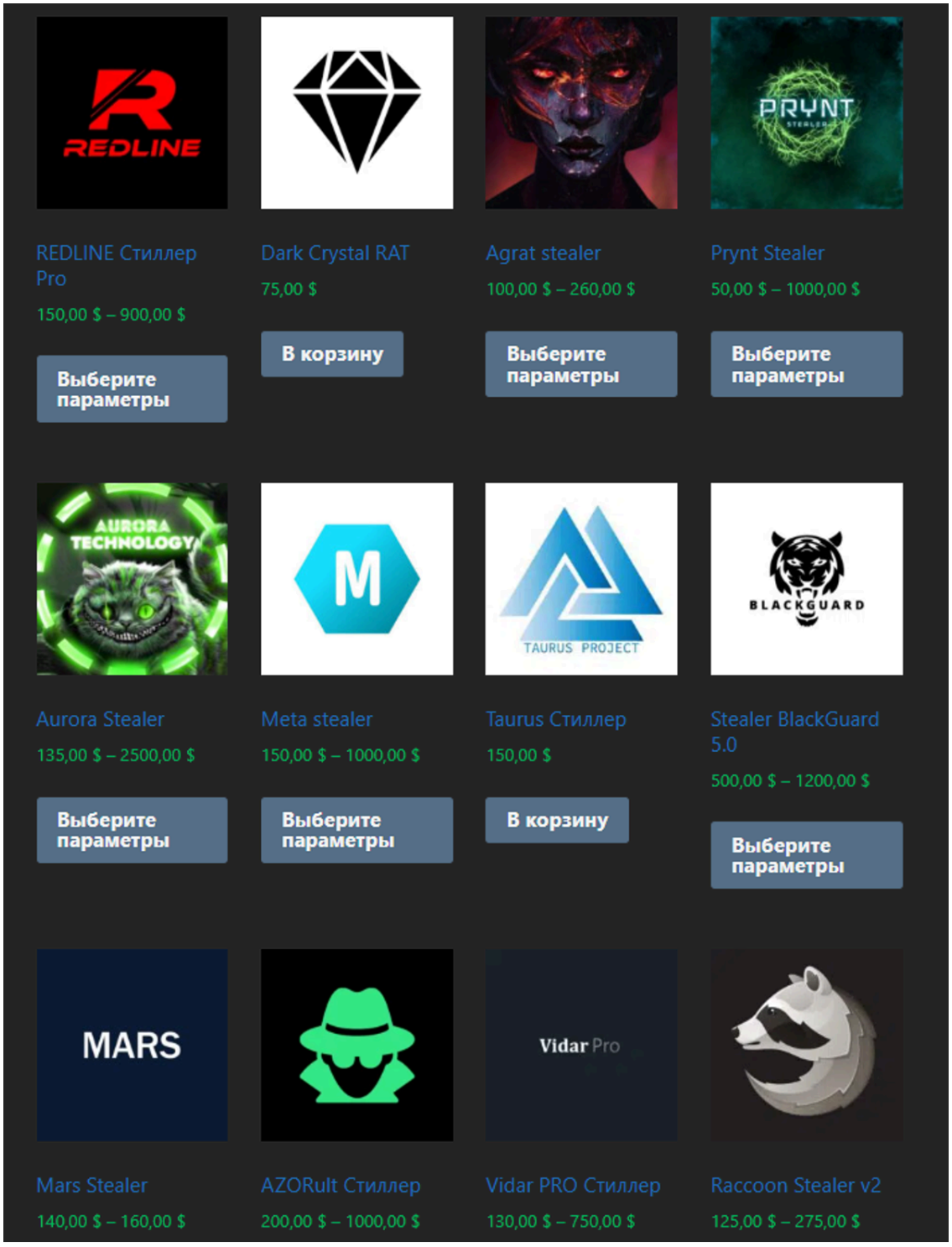


Figure 6: Pricing on Aurora Stealer compared to other stealers on the market

At the time of this writing, the malware developer advertised that the pre-orders come with lifetime access to Aurora Botnet and Aurora Stealer, including all the modules such as DDoS, SiteScanner, Loader, Brute Force, PowerShell/CMD execution, etc. (Figure 7).

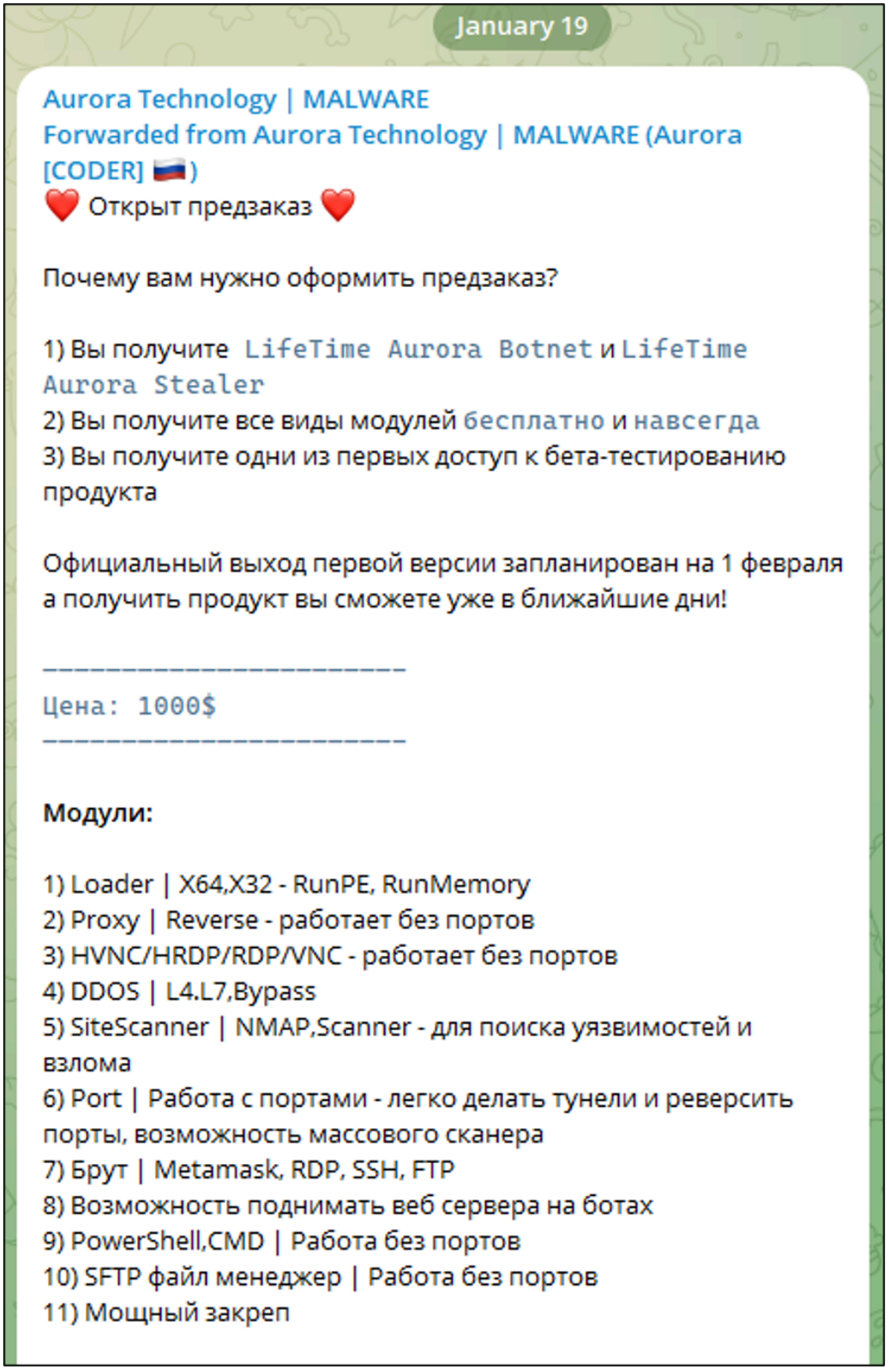


Figure 7: Pre-order advertisement on Aurora Stealer's Telegram channel

The cost for the pre-order is $1000. The botnet is a separate panel that allows an attacker to execute remote commands and perform specific tasks on the hosts, remote in using hVNC/HRDP/RDP/VNC (Figures 8-10).

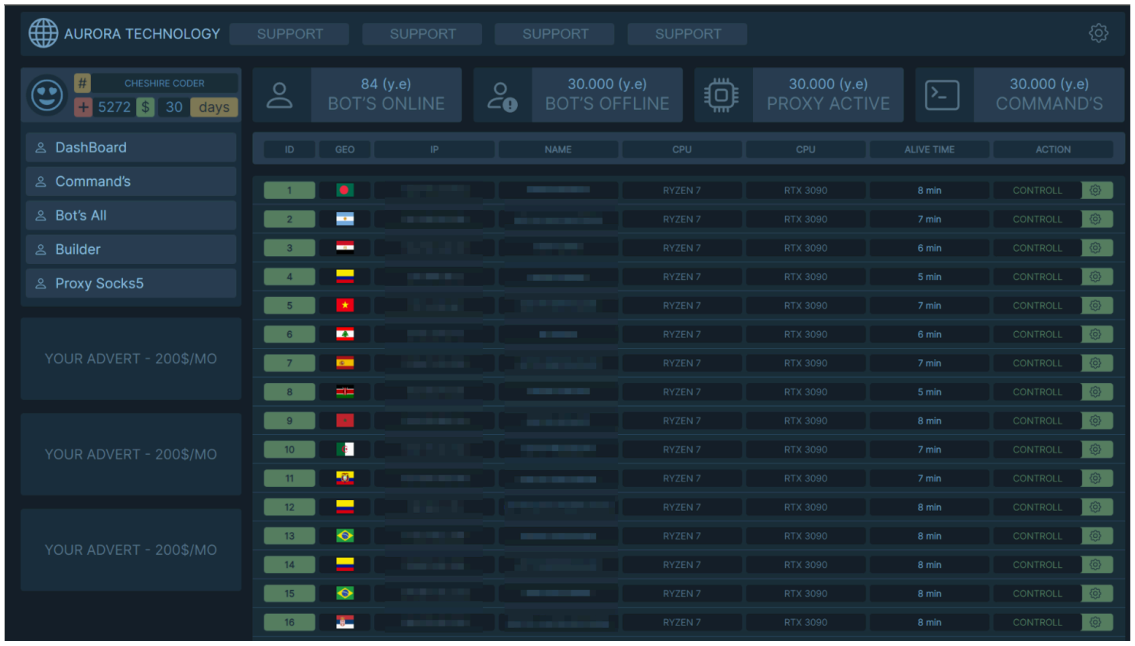


Figure 8: Botnet panel (1)

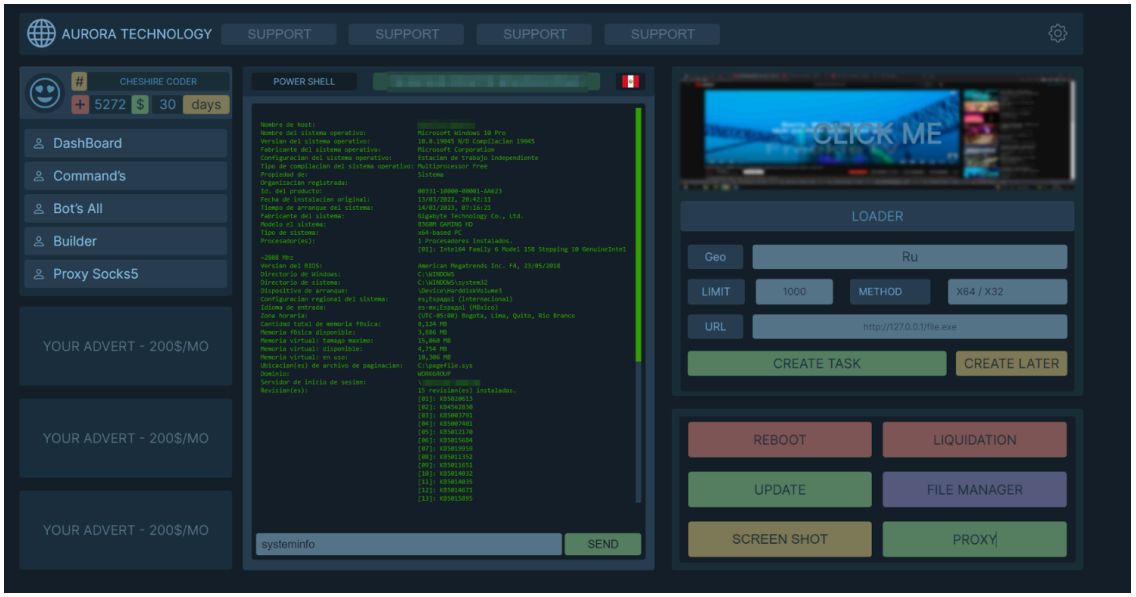


Figure 9: Botnet panel (2)

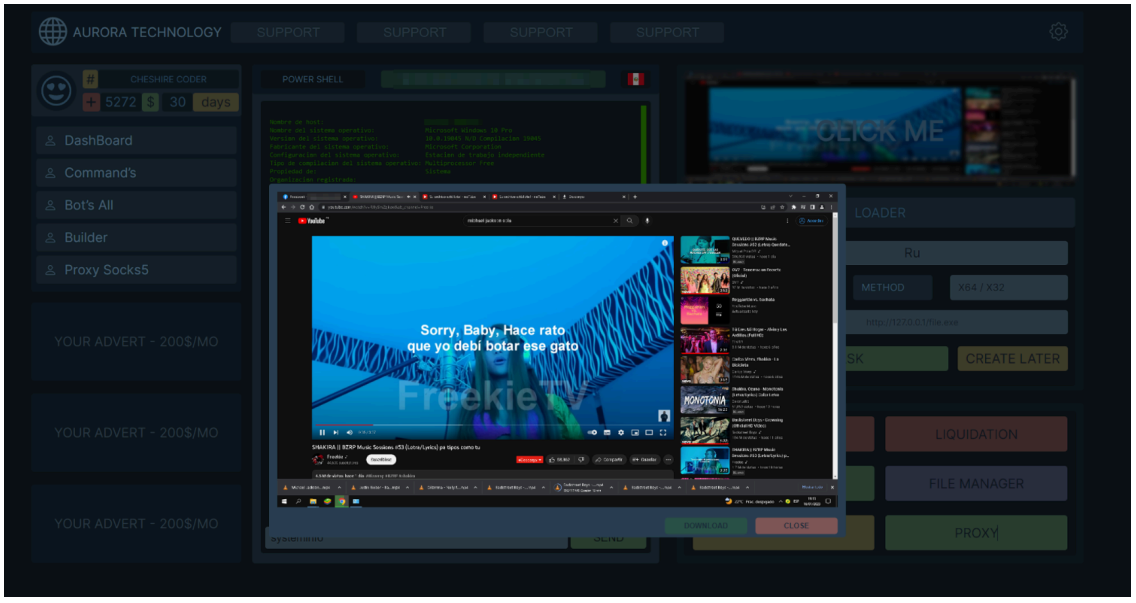


Figure 10: Botnet panel (3)

The Aurora stealer login can also be seen in Figure 11. A snippet of the Aurora manual for setting up and leveraging the malware can be seen in Figure 12.

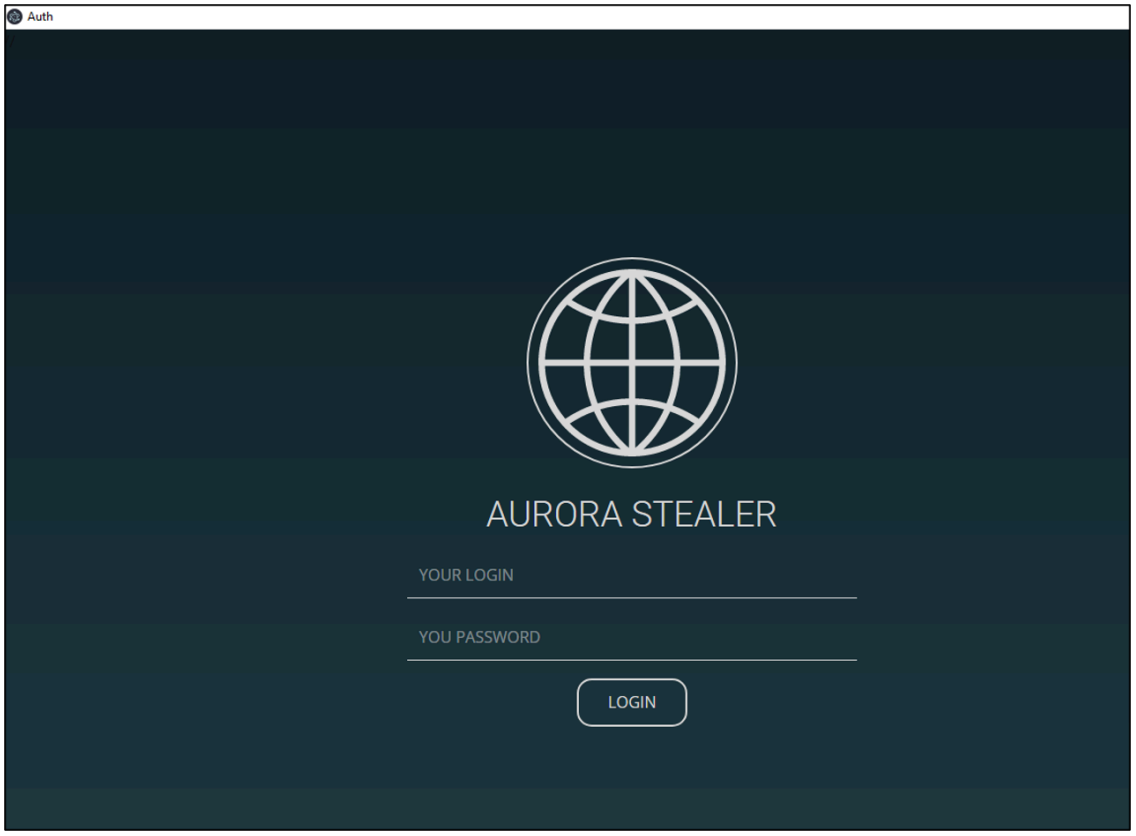


Figure 11: Aurora Stealer authentication panel

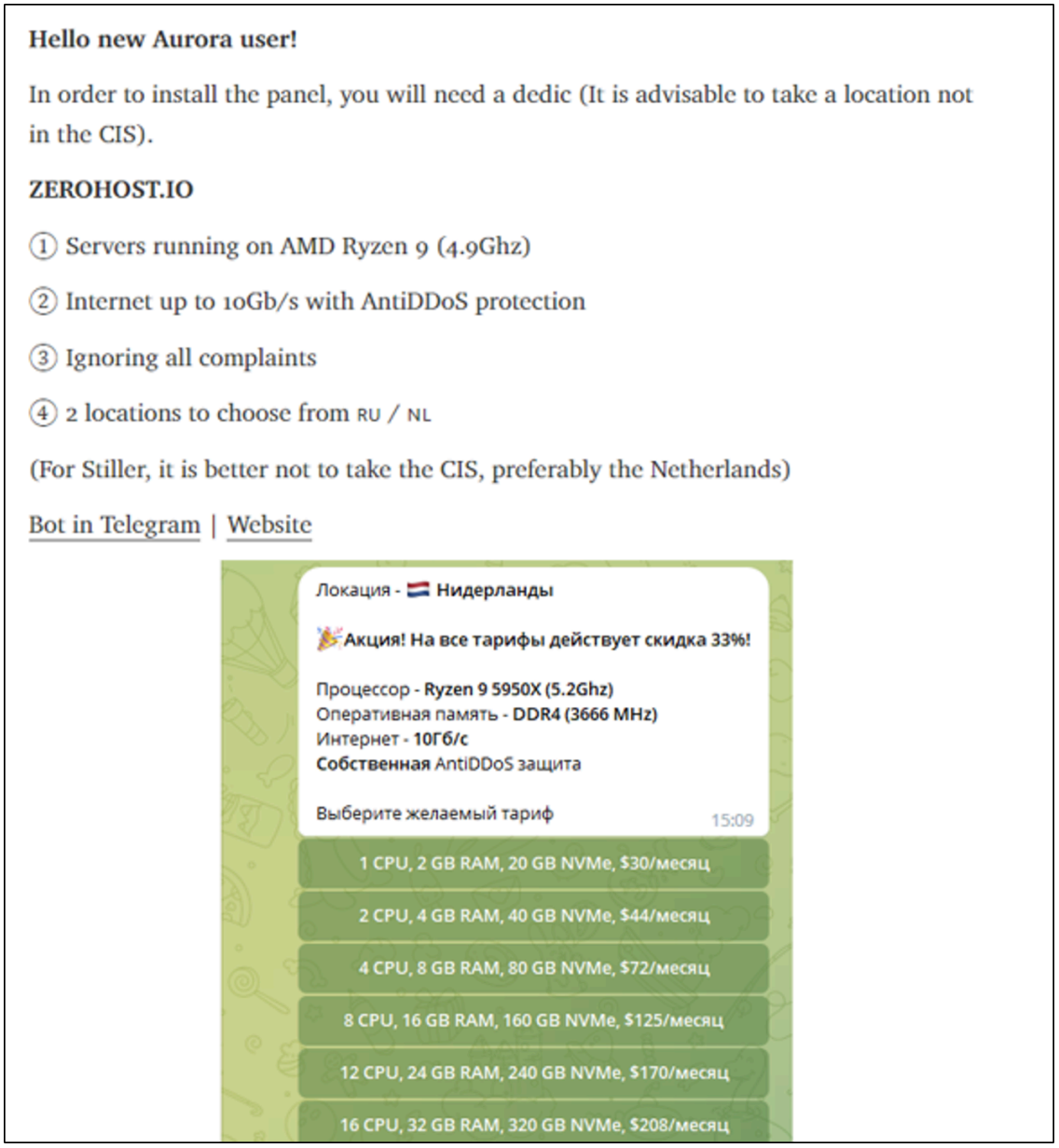


Figure 12: Snippet of the manual on how to set up the stealer

So, how does it spread?

Aurora Stealer is spread via installs (Russian: инсталл), also known as Pay-Per-Install (PPI) services, traffers (Russian: трафферы), or Google Ads. Pay-Per-Install (PPI) is a type of online advertising model where advertisers pay publishers a commission for every installation of their software or application that occurs as a result of the publisher's promotion. The end-user would be redirected to an attacker's landing page (Russian slang: ленд), where they download the malicious stealer payload.

The installs can also spread the stealer via the already infected hosts. The hosts can be infected with other malware families such as RATs (Remote Access Trojans). One of the popular install services that Aurora Stealer uses is InstallLabs (Figure 13).

Traffers are groups of people that are responsible for spreading the stealers via the links to the download pages via social media platforms such as Facebook and YouTube. The worker (Russian: воркер) is the individual within the

traffics group that is responsible for spreading the stealer.

Figure 13: InstallLabs ad on Russian-speaking forum

How can the stealer remain undetected?

To evade antivirus scanners, the attacker(s) usually fill the stealer binary with junk bytes to increase the file size, archive, and password-protect it. Aurora Stealer allows users to pack or add junk bytes into the build (stealer payload) to increase the file size for detection and sandbox evasion (Figure 14).

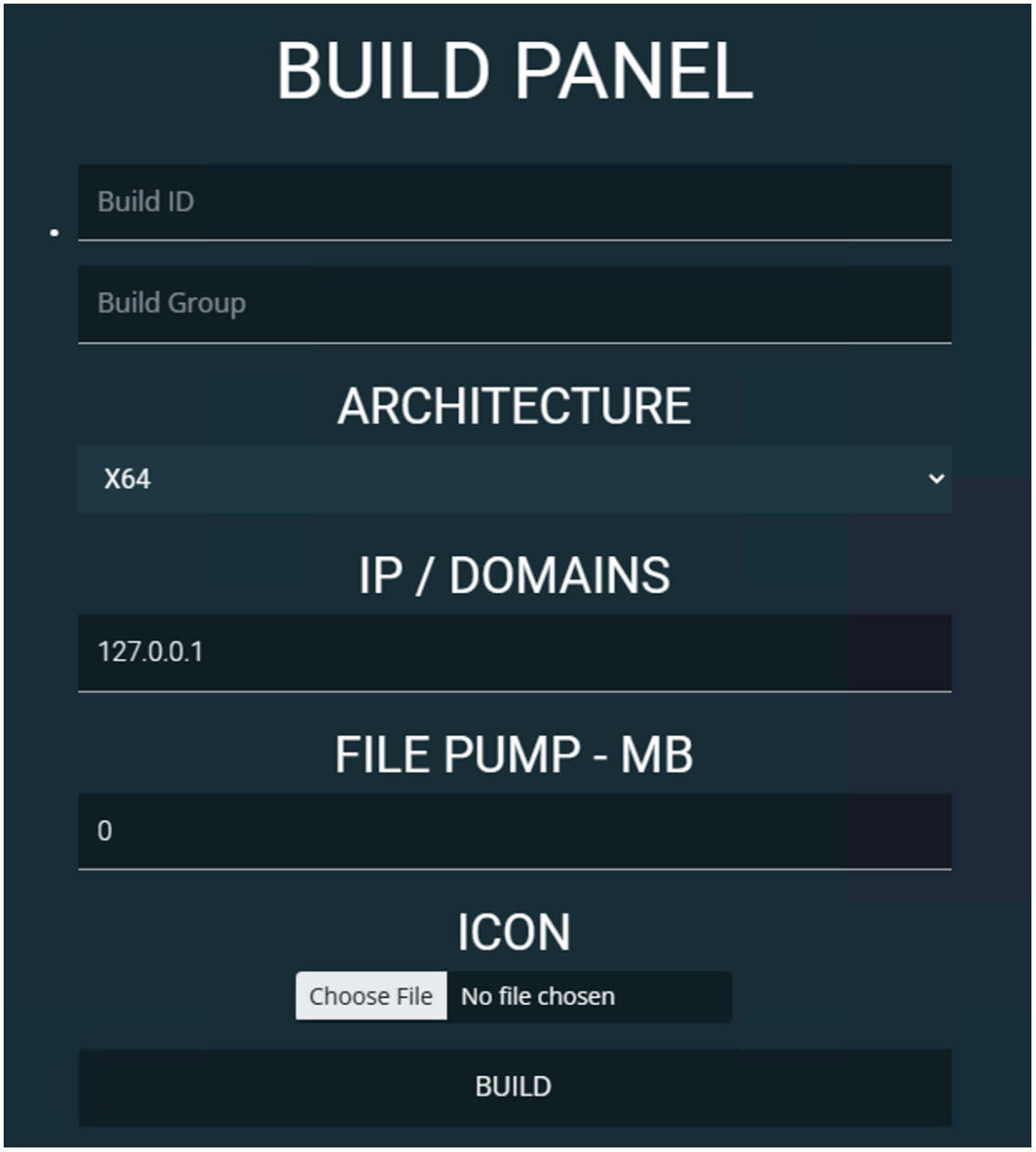


Figure 14: Build panel

The increase in the file size can significantly impact the stealer execution rate (Russian: отстук). The stealer execution rate is used to assess the quality of data transmission from the sender to the server. The better the quality of data transmission, the higher likelihood that the attacker receives all the stolen logs.

The attacker(s) can bypass SmartScreen controls by purchasing an EV certificate. SmartScreen is a security feature in Microsoft Windows that warns users about potentially unsafe websites and downloads. It uses a database of known threats and machine learning algorithms to identify new and suspicious behavior.

An EV (Extended Validation) certificate is a type of digital certificate used to authenticate and secure online communication. It verifies the identity of a website's owner and displays a green address bar in the browser to indicate that the site is trustworthy. Commonly used by financial and e-commerce websites, EV certificates are considered the highest level of validation and can be expensive to purchase (Figure 15).

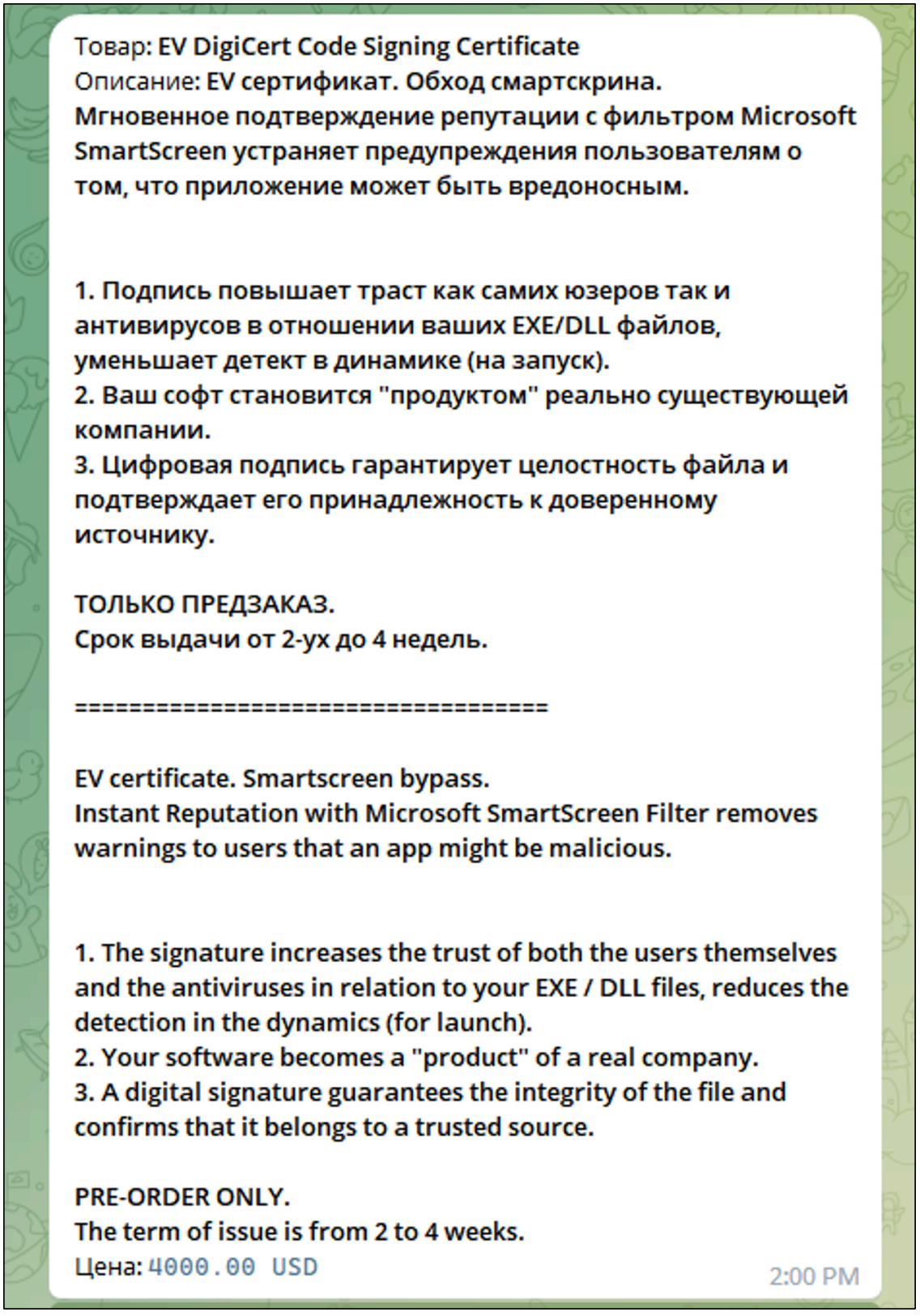


Figure 15: EV Certificate is being sold on Telegram

EV certificates can also be used to bypass User Account Control (UAC) alerts, which is a security feature in Windows operating systems that helps prevent unauthorized changes to a computer. When a user attempts to perform an action that requires elevated permissions, such as installing software or changing system settings, a UAC alert appears on the screen, asking the user to confirm the action.

The Case of a Cheshire Cat

The infection starts with the basic reconnaissance commands spawning from wmic.exe and cmd.exe (Figure 16):

- **wmic os get Caption –** returns the name of the operating system installed on the computer.
- **Cmd/C "wmic cpu get name" –** returns the processor's name on the computer.
- **cmd /C "wmic path win_32_VideoController get name" –** returns the name of the video controller on the computer.

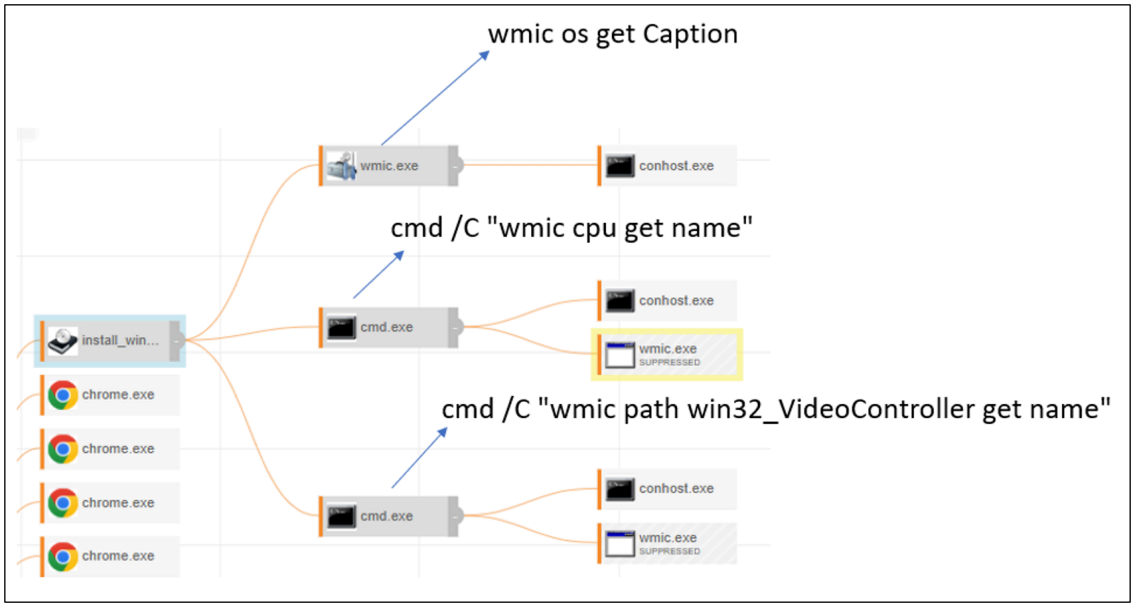


Figure 16: Infection chain

As mentioned before, the stealer binary is written in Go Programming language, the stealer binary without any size pumping and crypting, which involves obfuscating and encrypting the binary, is 2.96 MB in size.

The Aurora developer(s) offer their own crypting service for $40/1 crypt, $300/10 crypts (Figure 17).

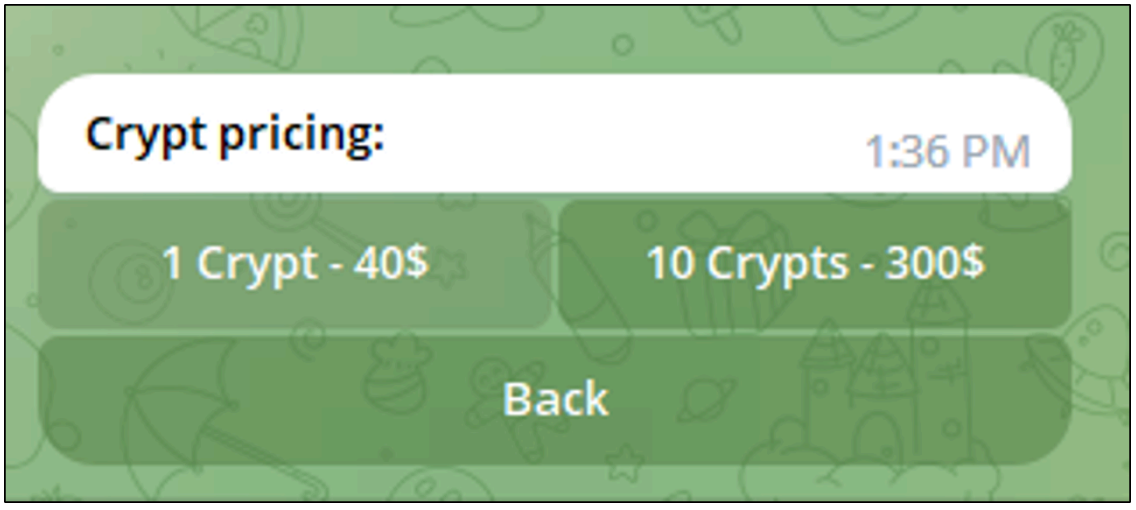


Figure 17: Crypt pricing

The function responsible for enumerating the GPU, CPU, and the caption of the operating system (gets the OS information) is shown in Figures 18-20. The gathered information is then sent to the stealer's panel and is contained in a text file named "UserInformation".

```
45  while ( (unsigned __int64)&v37 <= *(_QWORD *)(v9 + 16) )
46    a1 = runtime_morestack_noctxt(a1, a2, a3, a4, a5, a6, a7, a8, a9);
47  v36[4] = main_getGPU_func1;
48  v36[0] = "/C";
49  v36[1] = 2LL;
50  v36[2] = "wmic path win32_VideoController get name";
51  v36[3] = 40LL;
52  v34 = (_ptr_exec_Cmd)os_exec_Command((unsigned int)"cmd", 3, (unsigned int)v36, 2, 2, a6, a7, a8, a9);
53  p_syscall_SysProcAttr = (syscall_SysProcAttr *)runtime_newobject(&RTYPE_syscall_SysProcAttr);
54  p_syscall_SysProcAttr->HideWindow = 1;
55  v11 = v34;
```

Figure 18: Enumeration function (1)

```
41  while ( (unsigned __int64)&v35 <= *(_QWORD *)(v9 + 16) )
42    a1 = runtime_morestack_noctxt(a1, a2, a3, a4, a5, a6);
43  v34[4] = main_getCPU_func1;
44  v34[0] = "/C";
45  v34[1] = 2LL;
46  v34[2] = "wmic cpu get name";
47  v34[3] = 17LL;
48  v32 = os_exec_Command("cmd", 3uLL, v34, 2LL, 2LL, a6, a7, a8, a9);
```

Figure 19: Enumeration function (2)

```
41  while ( (unsigned __int64)&v35 <= *(_QWORD *)(v9 + 16) )
42    a1 = runtime_morestack_noctxt(a1, a2, a3, a4, a5, a6);
43  v34[4] = main_GetOS_func1;
44  v34[0] = "/C";
45  v34[1] = 2LL;
46  v34[2] = "wmic cpu get name";
47  v34[3] = 17LL;
48  v32 = os_exec_Command("cmd", 3uLL, v34, 2LL, 2LL, a6, a7, a8, a9);
49  p_syscall_SysProcAttr = (syscall_SysProcAttr *)runtime_newobject(&RTYPE_syscall_SysProcAttr);
50  p_syscall_SysProcAttr->HideWindow = 1;
51  v16 = v32;
```

Figure 20: Enumeration function (3)

The stealer mainly uses win, the Windows API package for Go, to perform specific tasks such as taking the screenshot of the host using the APIs such as CompatibleBitmap, CreateCompatibleDC, GetDC, and BitBlt (Figure 21).

```
230  CompatibleDC = github_com_lxn_win_CreateCompatibleDC(v152, 0, v13, a4, a5, v14, v15, v16, v17, v136);
231  if ( !CompatibleDC )
232  {
233    v122 = errors_New((unsigned int)"CreateCompatibleDC failed", 25, v36, a4, a5, v37, v38, v39, v40, v136);
234    v178 = 0LL;
235    *(_QWORD *)&v181 = v122;
236    *((_QWORD *)&v181 + 1) = 25LL;
237    runtime_deferreturn(v122, 25, v123, a4, a5, v124, v125, v126, v127);
238    return v178;
239  }
240  v149 = CompatibleDC;
241  v158[0] = main_Capture_func3;
242  v158[1] = CompatibleDC;
243  v173 = v158;
244  v11 = runtime_deferprocStack((unsigned int)&v172, 0, (unsigned int)v158, a4, a5, v37, v38, v39, v40);
245  if ( v11 )
246  {
247LABEL_28:
248    runtime_deferreturn(v11, a2, v13, a4, a5, v14, v15, v16, v17);
249    return v178;
250  }
251  else
252  {
253    CompatibleBitmap = github_com_lxn_win_CreateCompatibleBitmap(v152, v185, a4, a4, a5, v14, v15, v16, v17, v136);
254    if ( CompatibleBitmap )
255    {
256      v153 = CompatibleBitmap;
257      v156 = main_Capture_func4;
258      v157 = CompatibleBitmap;
259      v171 = &v156;
260      v47 = runtime_deferprocStack((unsigned int)&v170, v185, (unsigned int)&v156, a4, a5, v43, v44, v45, v46);
```

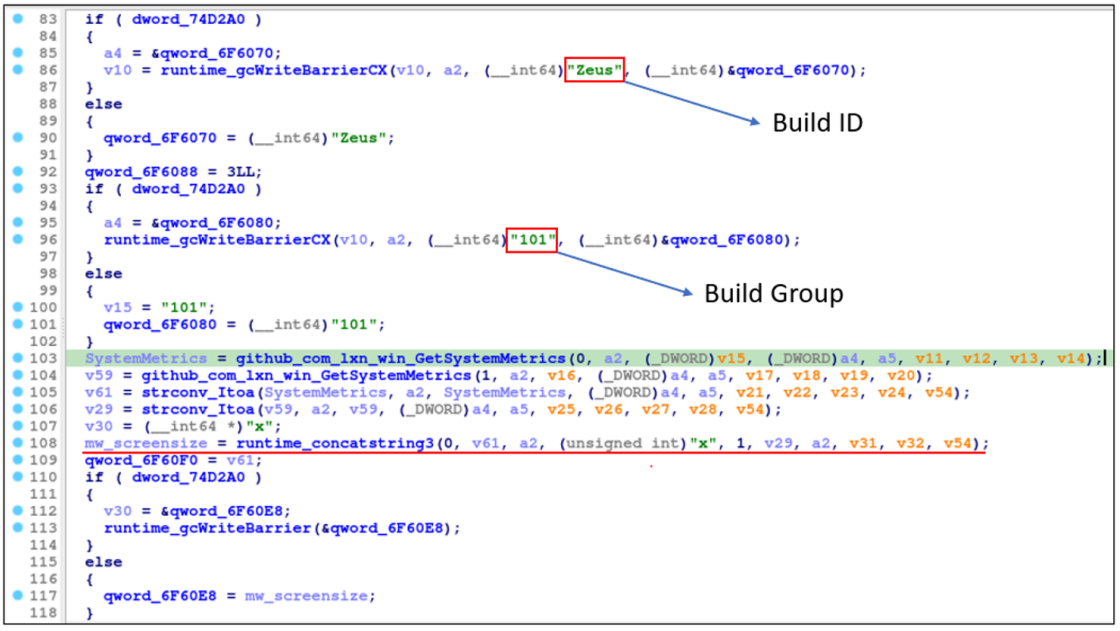
Figure 21: Screenshot capture function

The stealer retrieves the GUID of the infected machine via querying for the MachineGuid parameter under SOFTWARE\Microsoft\Cryptography (Figure 22).

```
38  mw_regopenkey = golang_org_x_sys_windows_registry_OpenKey(
39                    -2147483646,
40                    (unsigned int)"SOFTWARE\\Microsoft\\Cryptography",
41                    31,
42                    257,
43                    a5,
44                    a6,
45                    a7,
46                    a8,
47                    a9);
48  if ( "SOFTWARE\\Microsoft\\Cryptography" )
49  {
50    main_MachineID_func1(
51      mw_regopenkey,
52      (unsigned int)"SOFTWARE\\Microsoft\\Cryptography",
53      v12,
54      257,
55      a5,
56      v13,
57      v14,
58      v15,
59      v16);
60    return v10;
61  }
62  else
63  {
64    v24[0] = main_MachineID_func2;
65    v24[1] = mw_regopenkey;
66    v26 = (__int64 (**)(void))v24;
67    golang_org_x_sys_windows_registry_Key_GetStringValue(
68      mw_regopenkey,
69      (unsigned int)"MachineGuid",
70      11,
71      257,
72      a5,
73      v13,
```

Figure 22: Function responsible for getting the MachineGuid

The functions shown below are responsible for getting the infected machine's screen size and containing the Build ID, Build Group. The collected information is also written in the “UserInformation” text file (Figure 23).




```
83  if ( dword_74D2A0 )
84  {
85    a4 = &qword_6F6070;
86    v10 = runtime_gcWriteBarrierCX(v10, a2, (__int64)"Zeus", (__int64)&qword_6F6070);
87  }
88  else
89  {
90    qword_6F6070 = (__int64)"Zeus";
91  }
92  qword_6F6088 = 3LL;
93  if ( dword_74D2A0 )
94  {
95    a4 = &qword_6F6080;
96    runtime_gcWriteBarrierCX(v10, a2, (__int64)"101", (__int64)&qword_6F6080);
97  }
98  else
99  {
100   v15 = "101";
101   qword_6F6080 = (__int64)"101";
102 }
103 SystemMetrics = github_com_lxn_win_GetSystemMetrics(0, a2, (_DWORD)v15, (_DWORD)a4, a5, v11, v12, v13, v14);
104 v59 = github_com_lxn_win_GetSystemMetrics(1, a2, v16, (_DWORD)a4, a5, v17, v18, v19, v20);
105 v61 = strconv_Itoa(SystemMetrics, a2, SystemMetrics, (_DWORD)a4, a5, v21, v22, v23, v24, v54);
106 v29 = strconv_Itoa(v59, a2, v59, (_DWORD)a4, a5, v25, v26, v27, v28, v54);
107 v30 = (__int64 *)"x";
108 mw_screensize = runtime_concatstring3(0, v61, a2, (unsigned int)"x", 1, v29, a2, v31, v32, v54);
109 qword_6F60F0 = v61;
110 if ( dword_74D2A0 )
111 {
112   v30 = &qword_6F60E8;
113   runtime_gcWriteBarrier(&qword_6F60E8);
114 }
115 else
116 {
117   qword_6F60E8 = mw_screensize;
118 }
```

Figure 23: Function containing the Build Group, Build ID, and functions responsible for getting the screen size

Aurora Stealer gathers sensitive data, including cookies, autofill information, and encrypted passwords from browsers such as Opera, Brave, Mozilla Firefox, Chrome, etc. The gathered information is temporarily stored under the %temp% folder (Figure 24). However, it is worth noting that the stealer does not collect credentials from Mozilla Firefox.

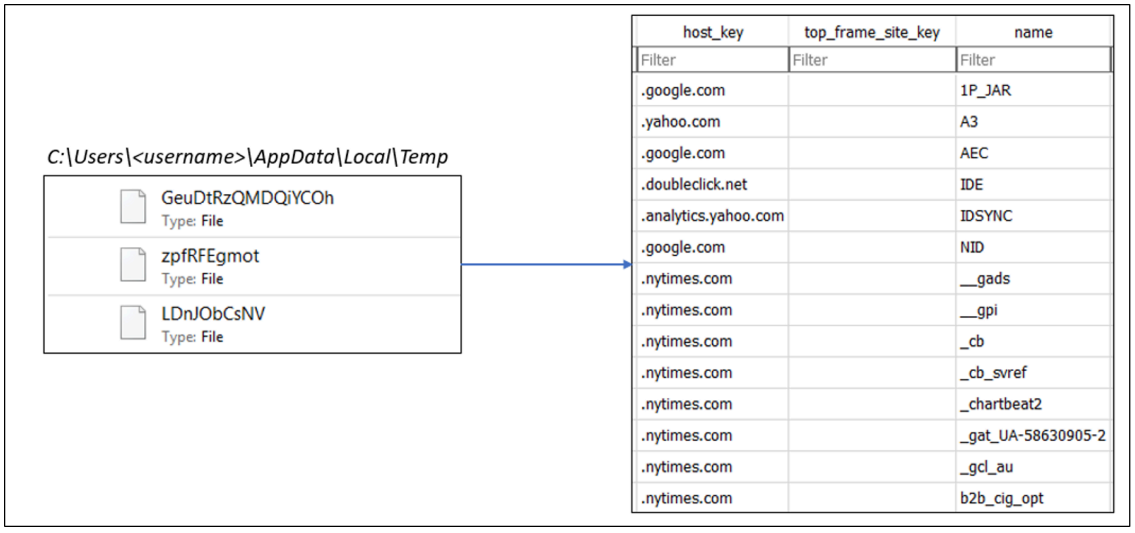


Figure 24: Temporarily stored data under %temp%

Under the function main_getMasterKey, we can see the references to os_crypt, encrypted_key, and DPAPI (Figure 25).

Figure 25: References to os_crypt, encrypted_key, DPAPI

DPAPI (Data Protection Application Programming Interface) is used, for example, to store cookies and password information for Chrome browsers DPAPI uses APIs CryptProtectData and CryptUnprotectData to encrypt and decrypt data accordingly.

Chrome stores the DPAPI-encrypted AES key, which is the Master Key under os_encrypted.encrypted_key in a base64-encoded format. To be able to decrypt the saved credentials and cookies, Aurora Stealer needs to decode the base64-encoded string and call the CryptUnprotectData function, then remove the padding from the master key.

Aurora Stealer has multiple Grabber functions that are responsible for collecting additional data such as crypto wallets, screenshots, files, Telegram, etc. (Figure 26)

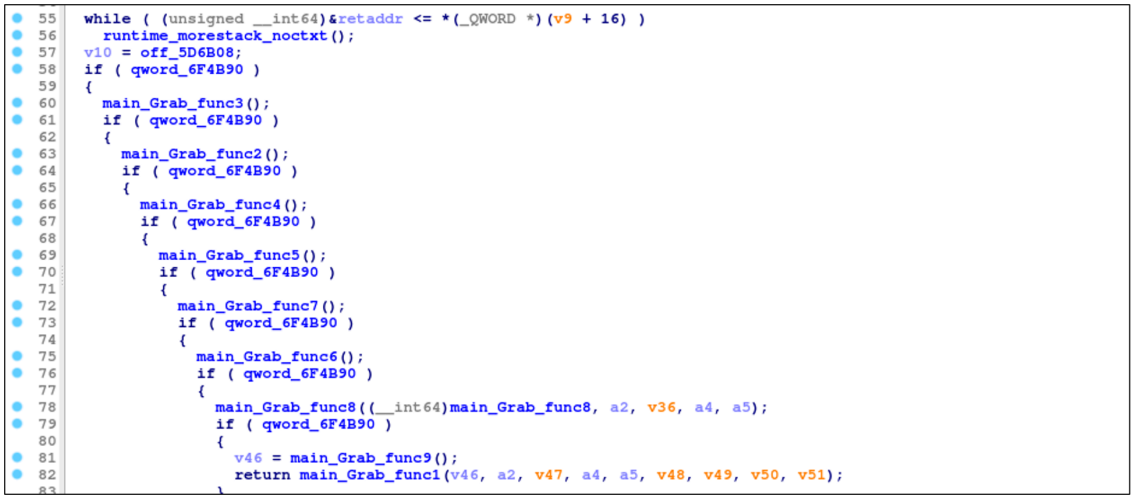


Figure 26: Main grabber functions

The stealer also grabs the files from the folder “Windows.old” which stores the backup copy of the previous Windows installation if applicable (Figure 27).

```
180         a2 = os_Getenv((unsigned int)"USERPROFILE", 11, v38, a4, (_DWORD)a5, a6, a7, a8, a9, v56, v61);
181         v43 = runtime_concatstring2(0, a2, 11, v82, (_DWORD)v83, v39, v40, v41, v42, v59, v64, v66);
182         LODWORD(a4) = 9;
183         a5 = "C:\\Windows.old\\Users\\";
184         v46 = strings_Replace(
185                 v43,
186                 a2,
187                 (unsigned int)"C:\\Users\\",
188                 9,
189                 (unsigned int)"C:\\Windows.old\\Users\\",
190                 21,
191                 -1,
192                 v44,
193                 v45,
194                 v60,
195                 v65,
196                 v67,
197                 v68,
198                 v69,
199                 v70,
200                 v71);
201         v79 = main_Grab_func4_3;
202         v80 = v81;
203         a1 = path_filepath_Walk(
204                 v46,
205                 a2,
206                 (__int64 (__golang **)(_QWORD, _QWORD, _QWORD, _QWORD, _QWORD, _QWORD))&v79,
207                 9LL,
208                 (__int64)"C:\\Windows.old\\Users\\",
209                 v47,
210                 v48,
211                 v49,
212                 v50);
```

Figure 27: Grabber function to collect the backup files from the previous version of Windows

This grabber function searches for crypto wallets under AppData\Roaming (Figure 28), for example, for leveldb files that store the private keys:

- AppData\Roaming\Guarda\Local Storage\leveldb
- AppData\Roaming\atomic\Local Storage\leveldb

```
30  while ( (unsigned __int64)&v25 <= *(_QWORD *)(v9 + 16) )
31    a1 = runtime_morestack_noctxt(a1, a2, a3, a4, a5, a6, a7, a8, a9);
32  v25 = &off_5D6B50;
33  v10 = off_6E9B50;
34  if ( qword_6E9B58 )
35  {
36    v21 = qword_6E9B58;
37    for ( i = 0LL; ; i = v20 + 1 )
38    {
39      v20 = i;
40      v23 = v10;
41      a2 = v10[1];
42      a1 = path_filepath_Walk(                    // Cryptowallet search under AppData\Roaming
43              *v10,
44              a2,
45              (__int64 (__golang **)(_QWORD, _QWORD, _QWORD, _QWORD, _QWORD, _QWORD))&off_5D6B58,
46              a4,
47              (__int64)v10,
48              a6,
49              a7,
50              a8,
51              a9);
```

Figure 28: Grabber function responsible for crypto wallet search

Below is the grabber function for the Telegram tdata folder that would let the attacker authenticate into the victim's Telegram on the Desktop version by placing the tdata folder in the same folder as the Telegram client (Figure 29).

```
108  v67 = v10;
109  v68 = v10;
110  v29 = "\\AppData\\Roaming\\Telegram Desktop\\tdata";
111  v30 = 39LL;
112  *(_QWORD *)&v67 = runtime_concatstring2(
113                      0,
114                      v63,
115                      11,
116                      (unsigned int)"\\AppData\\Roaming\\Telegram Desktop\\tdata",
117                      39,
118                      v31,
119                      v32,
120                      v33,
121                      v34,
122                      v51,
123                      v55,
124                      v59);
```

Figure 29: Snipper of Grabber function that searches for Telegram tdata

Just like other stealers such as Redline, Raccoon Stealer, and Vidar Stealer, Aurora Stealer has two modules: grabber and loader. The grabber module retrieves the files or folders specified by an attacker. The gathered files/folders would then be archived in a zip file named temp.zip, stored under %userprofile% (Figure 30-31).

The “END_PACKET_ALL_SEND” message is likely used for debugging logs.

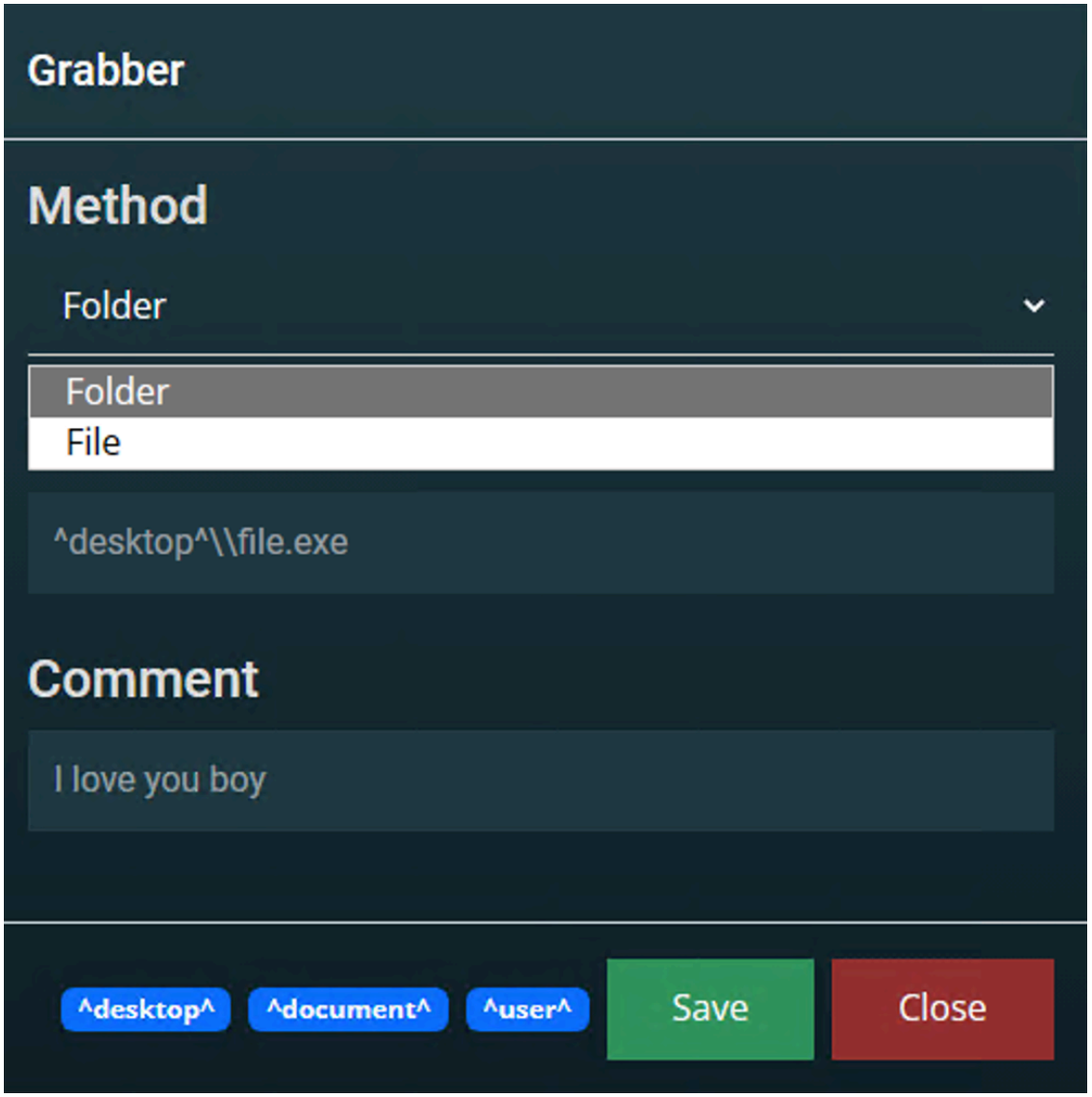


Figure 30: Grabber module (1)

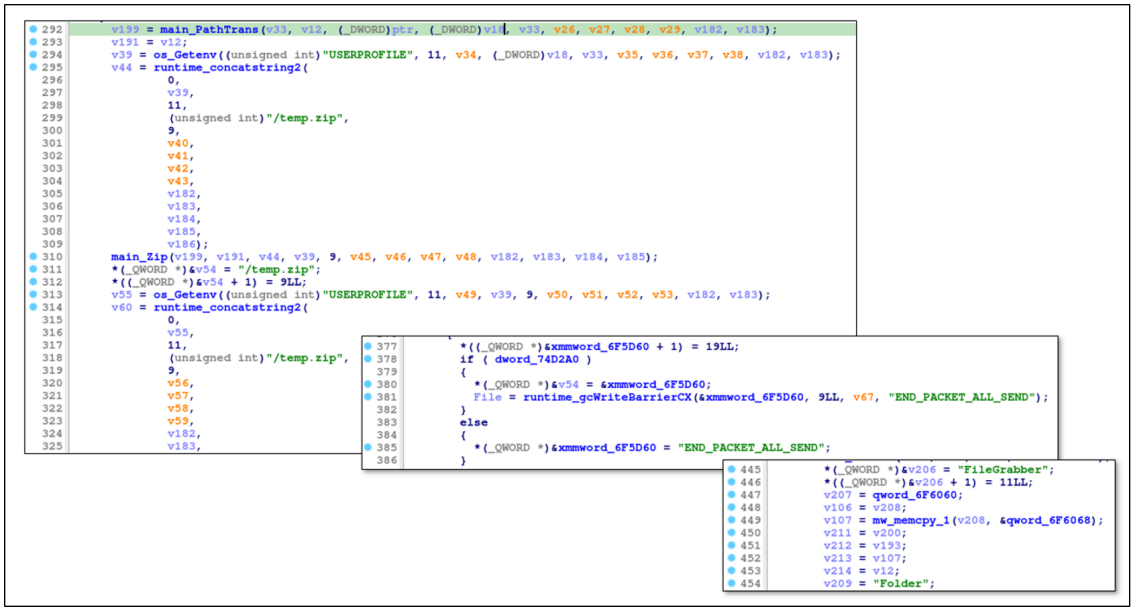


Figure 31: Grabber module (2)

The stealer stores the loader (Figure 32), grabber, and the general configuration information within the build in the base64-encoded form (Figure 33). The loader module has two options:

- **Download and Run –** the attacker specifies the direct download link to the additional payload.
- **PowerShell –** the attacker specifies the PowerShell command to run on the host.

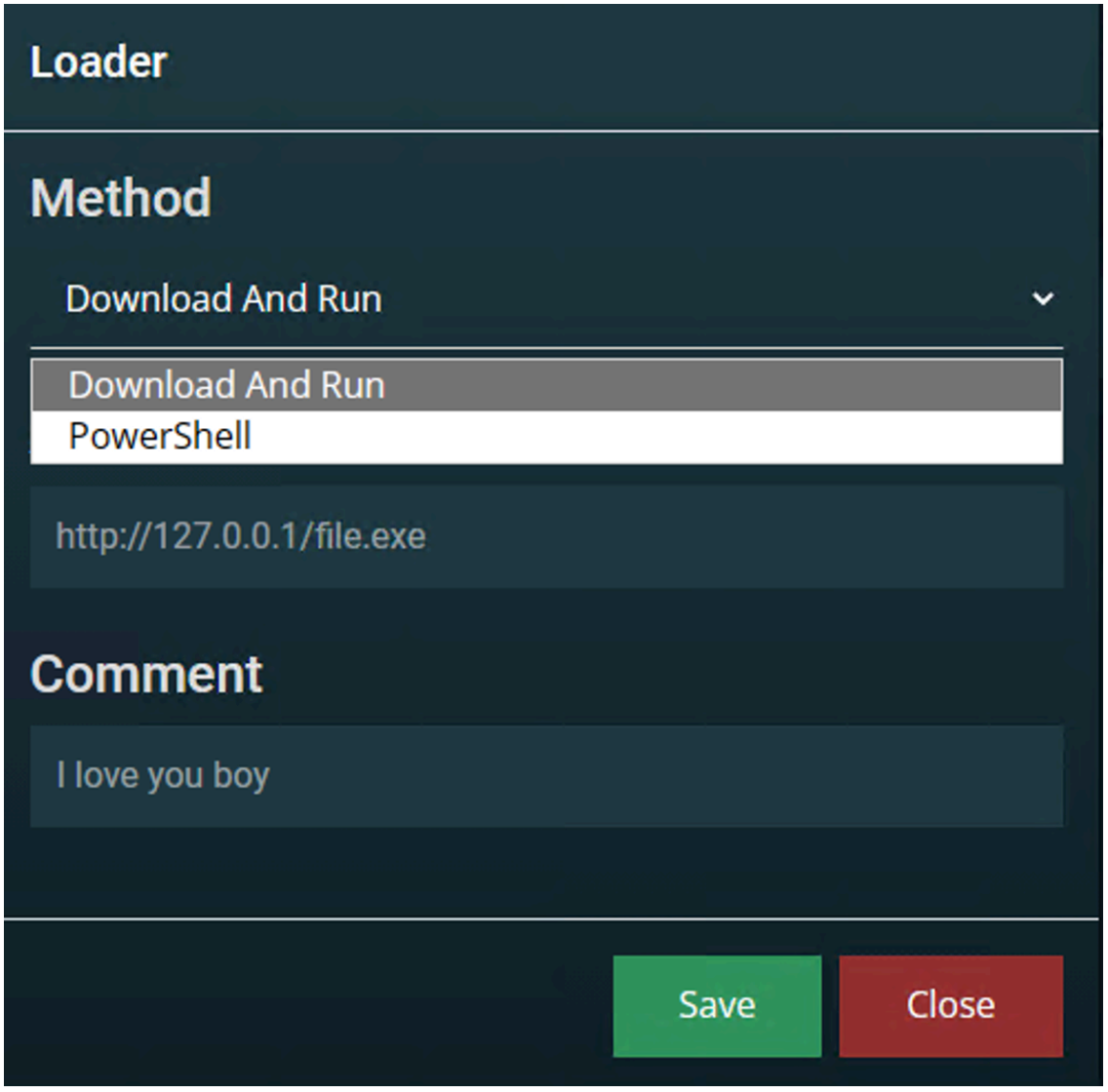


Figure 32: Loader module

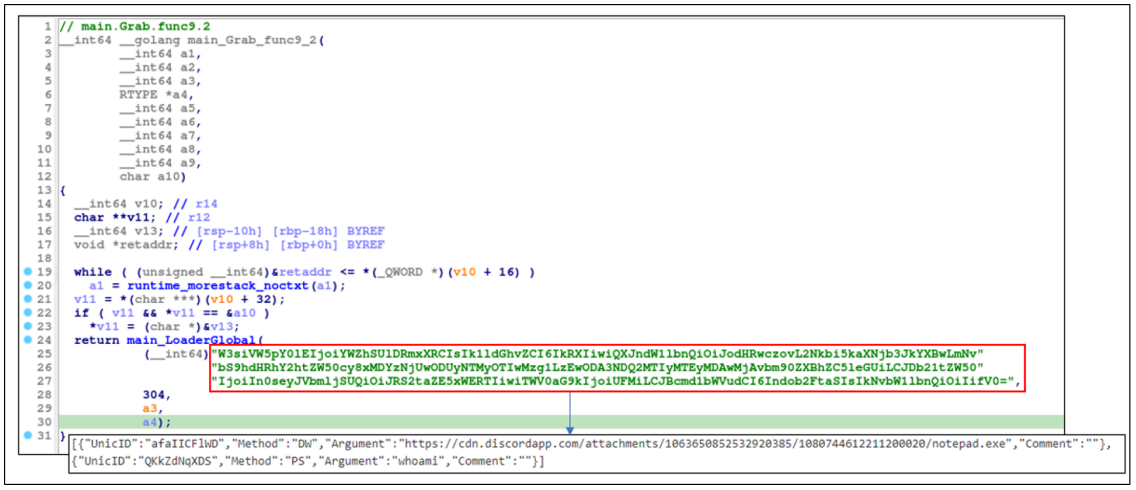


Figure 33: Loader configuration

The Loader module, where:

- DW is the downloader parameter.

- PS is the PowerShell parameter.
- UnicID is the unique identifier.
- Argument contains the loader task.

The loader downloader module pulls an executable from the file hosting server at the end of the stealer execution and places it under the %temp% folder. The stealer executes the secondary payload using “start-process” Powershell cmdlet, as shown in Figure 34.

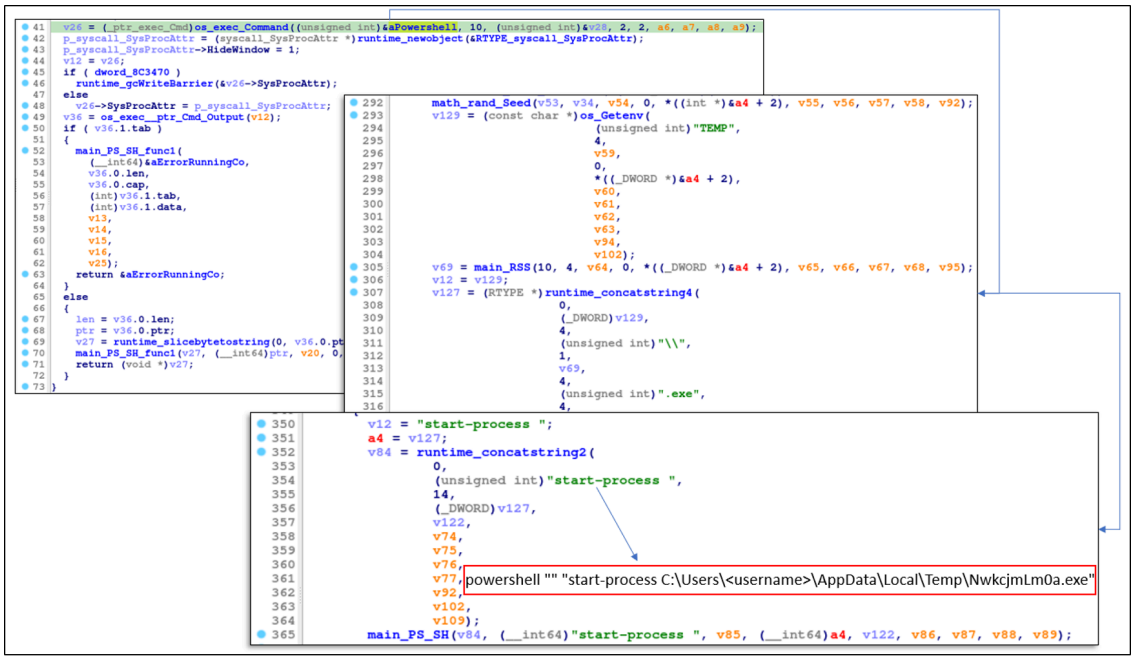


Figure 34: Loader Downloader module

The Grabber module configuration contains the path specified by an attacker to grab certain files/folders from. “FoF” parameter is likely the marker for whether the file folder grabber is specified (Figure 35).

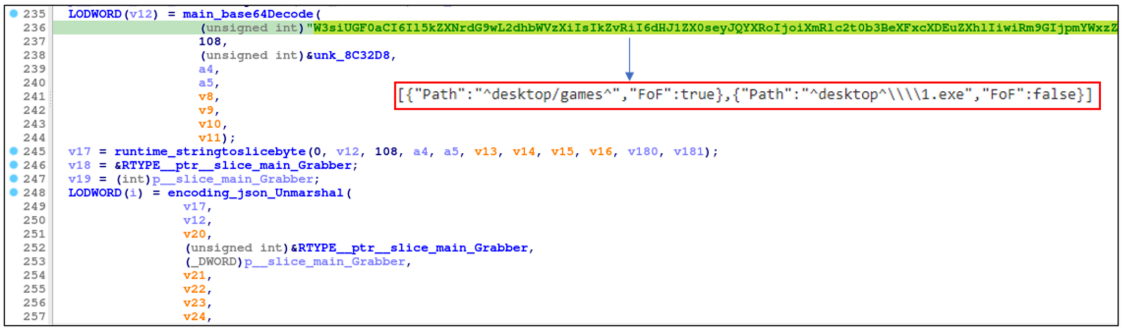


Figure 35: Grabber configuration

Aurora Stealer stores its build configuration at the end of the binary in the base64-encoded format (Figure 36). However, the configuration will likely be stripped if the stealer is encrypted.

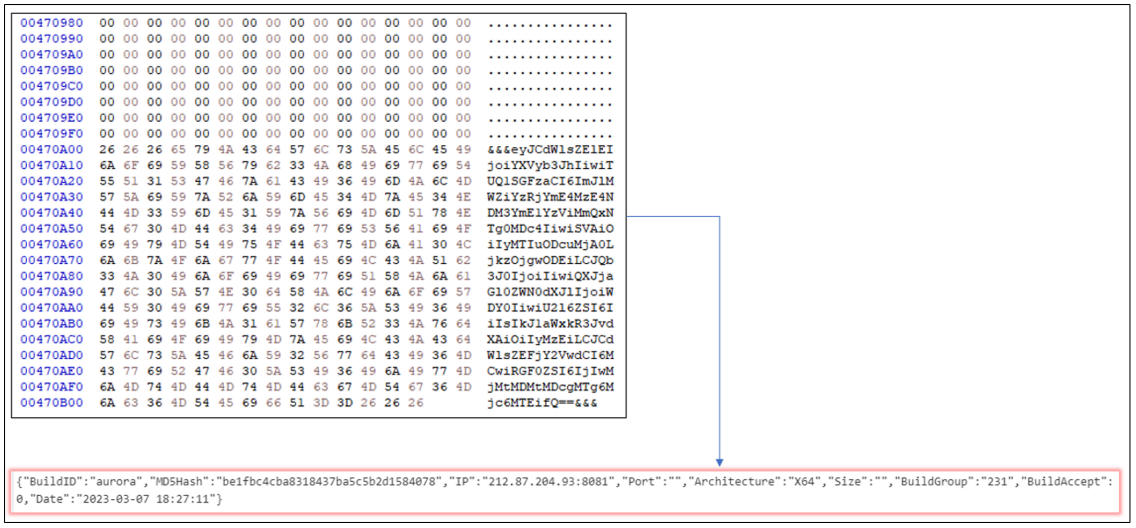


Figure 36: Configuration blob

We wrote the configuration extractor script in Python for Aurora Stealer that looks for base64-encoded patterns within the binary.

The function main_ConnectToServer attempts to connect to the C2 server while printing the log messages, it sleeps after attempting to reconnect for one second and retries if the connection is unsuccessful (Figure 37).

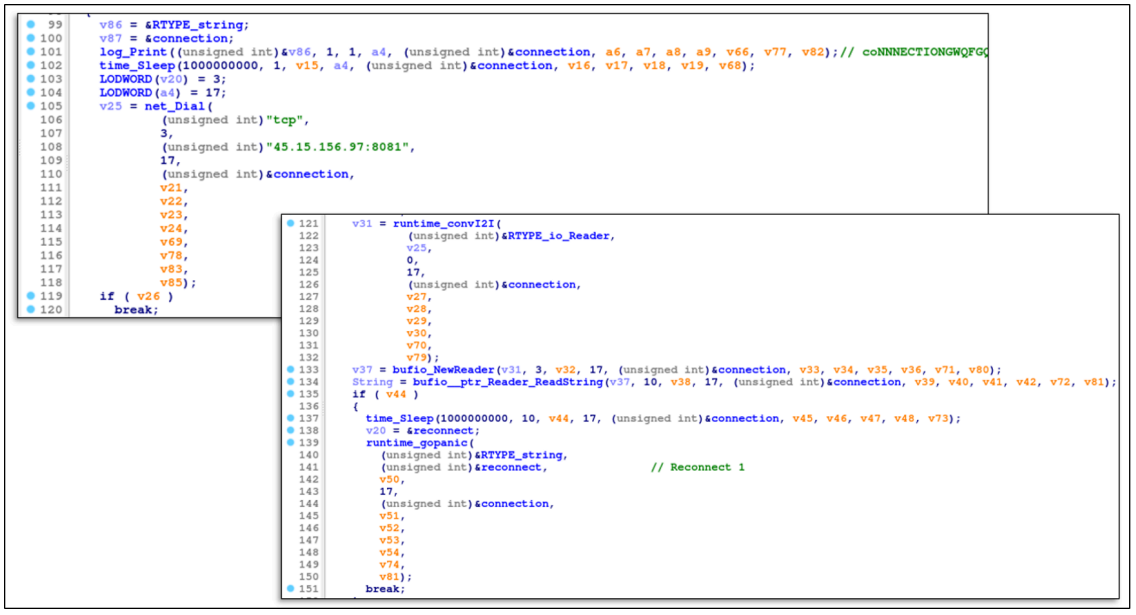


Figure 37: main_ConnectToServer function

If the connection is successful, the function exits with code “666” and log message “BLACK ZONE”.

main_PathTrans function is responsible for replacing the strings such as ^user^, ^document^, and ^desktop^ within the Grabber configuration with the paths of Desktop, Document, and %userprofile% (Figure 38).

```
}
v50 = a1;
v10 = os_Getenv((unsigned int)"USERPROFILE", 11, a3, a4, a5, a6, a7, a8, a9);
v15 = runtime_concatstring2(
        (unsigned int)&v47,
        v10,
        11,
        (unsigned int)"\\Desktop",
        8,
        v11,
        v12,
        v13,
        v14,
        v40,
        v42,
        v44);
v48 = strings_Replace(v50, a2, (unsigned int)"^desktop^", 9, v15, v10, -1, v16, v17);
v23 = os_Getenv((unsigned int)"USERPROFILE", 11, v18, 9, v15, v19, v20, v21, v22);
v28 = runtime_concatstring2(
        (unsigned int)&v46,
        v23,
        11,
        (unsigned int)"\\Documents",
        10,
        v24,
        v25,
        v26,
        v27,
        v41,
        v43,
        v45);
v48 = strings_Replace(v48, a2, (unsigned int)"^document^", 10, v28, v23, -1, v29, v30);
v36 = os_Getenv((unsigned int)"USERPROFILE", 11, v31, 10, v28, v32, v33, v34, v35);
return strings_Replace(v48, a2, (unsigned int)"^user^", 6, v36, 11, -1, v37, v38);
}
```

Figure 38: main_PathTrans function

March 2023 Update

In March 2023, the stealer developer released the first update since October 2022, as shown in Figure 39.

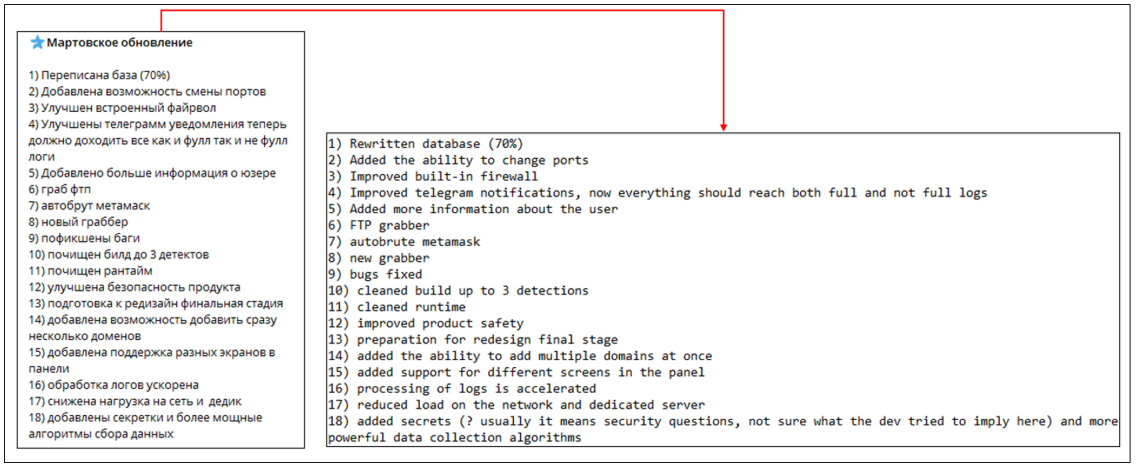


Figure 39: March update

One of the major changes is the stealer's capabilities to grab FTP (FileZilla) and RDP credentials as well as the ability to change the ports to the stealer's panel and C2 communications and specify extensions, disk drives for the grabber module (Figure 40-41).

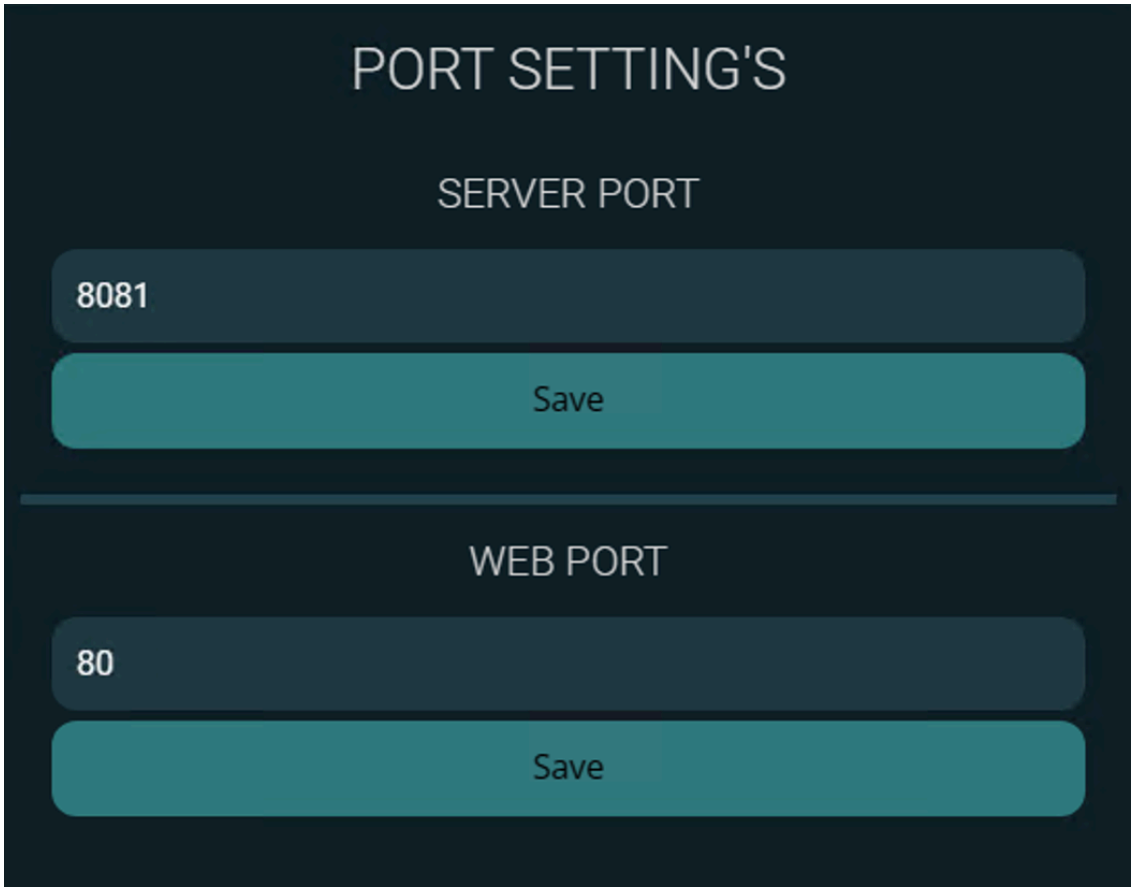


Figure 40: Port settings in the panel

```
.text:0000000000575F73    lea     rdx, aShell     ; "Shell"
.text:0000000000575F7A    mov     qword ptr [rsp+8E0h+var_2C0], rdx
.text:0000000000575F82    mov     qword ptr [rsp+8E0h+var_2C0+8], 5
.text:0000000000575F8E    mov     rsi, cs:qword_70CAC0
.text:0000000000575F95    mov     [rsp+8E0h+var_2B0], rsi
.text:0000000000575F9D    lea     rdi, [rsp+8E0h+var_2A8]
.text:0000000000575FA5    lea     rsi, qword_70CAC8
.text:0000000000575FAC    mov     [rsp+8E0h+var_8F0], rbp
.text:0000000000575FB1    lea     rbp, [rsp+8E0h+var_8F0]
.text:0000000000575FB6    call    sub_464046
.text:0000000000575FBB    mov     rbp, [rbp+0]
.text:0000000000575FBF    lea     rsi, aFilezila  ; "FileZila"
.text:0000000000575FC6    mov     [rsp+8E0h+var_98], rsi
.text:0000000000575FCE    mov     [rsp+8E0h+var_90], 8
.text:0000000000575FDA    mov     ecx, 1
.text:0000000000575FDF    mov     rdi, [rsp+8E0h+var_5A0]
.text:0000000000575FE7    mov     rsi, [rsp+8E0h+var_608]
.text:0000000000575FEF    lea     r8, asc_5BF3D4  ; ")"
.text:0000000000575FF6    mov     r9, rcx
.text:0000000000575FF9    mov     r10, [rsp+8E0h+var_5A8]
.text:0000000000576001    mov     r11, [rsp+8E0h+var_610]
.text:0000000000576009    xor     eax, eax
```

Figure 41: Snippet of the FTP grabber

Besides the WMIC commands mentioned at the beginning of this report, the stealer developer added two new commands to run upon the execution of the malware:

- **cmd.exe /c "wmic csproduct get uuid":** the command retrieves the universally unique identifier (UUID) of the computer's system product
- **systeminfo:** The command is used to display detailed information about the operating system, hardware, and software components of a Windows computer system

Upon the execution of the stealer, PowerShell processes are spawned to copy the browsing data such as cookies, history, and credentials to *AppData\Local\Temp* directory under a randomly named folder, the example command:

- powershell "" "copy \"C:\Users\<username>\AppData\Local\Google\Chrome\User Data\Default\Login Data\" \"C:\Users\<username>\AppData\Local\Temp\<random_folder>\""

C2 Communication & Stealer Logs

Aurora Stealer uses port 8081 for default communication with the C2 server, so prior to the stealer installation on the attacker's server, it's required to enable port 8081 through the firewall for the incoming traffic (Figure 42).

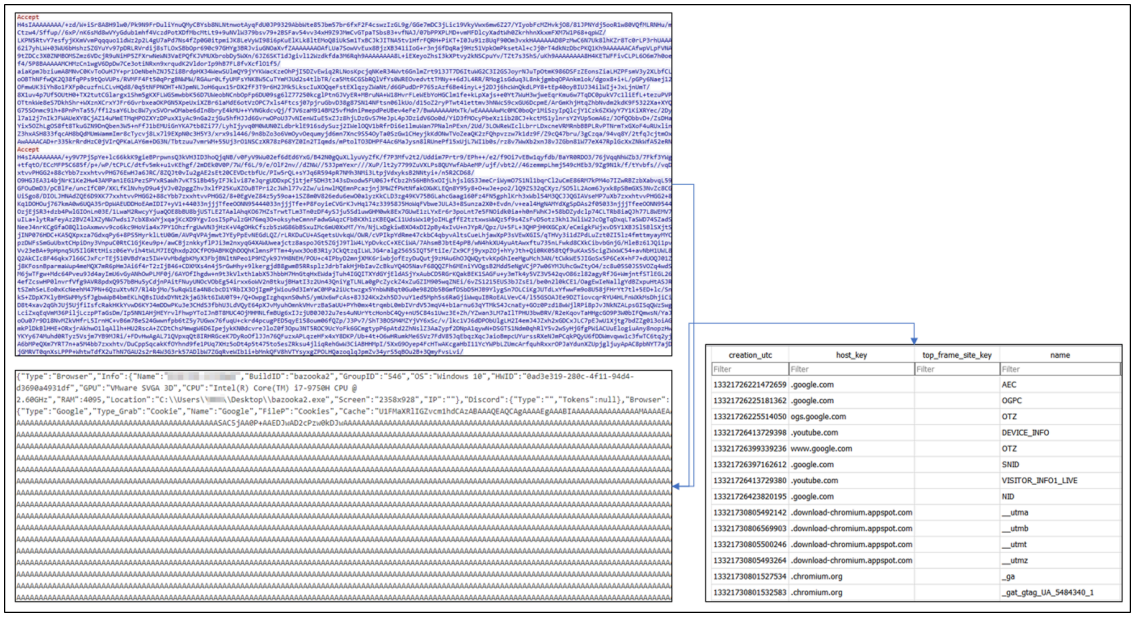


Figure 42: Stealer logs sent to the C2 server

The stealer logs are sent to the C2 server in JSON format, GZIP-compressed and base64-encoded. The stealer logs are stored in the Aurora build folder in the format [Country]HWID_BuildID (Figure 43-44).

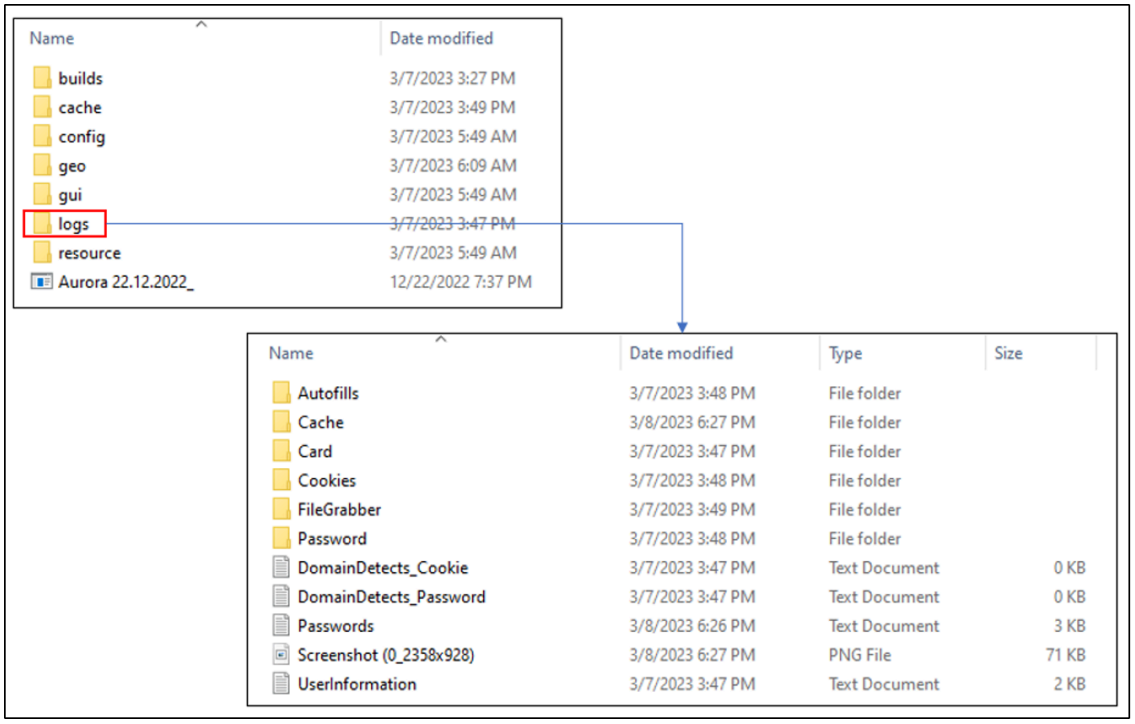


Figure 43: Stealer logs

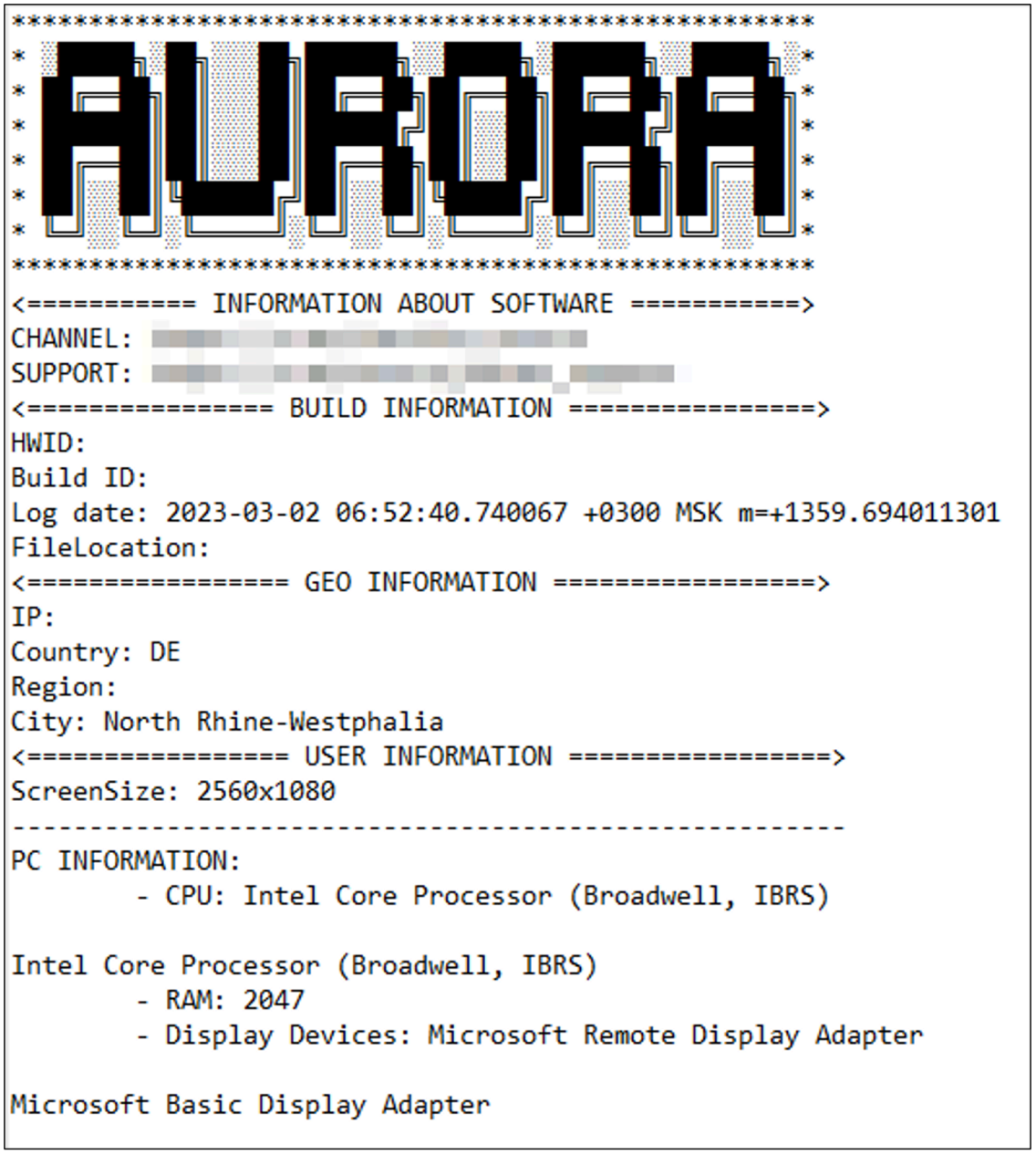
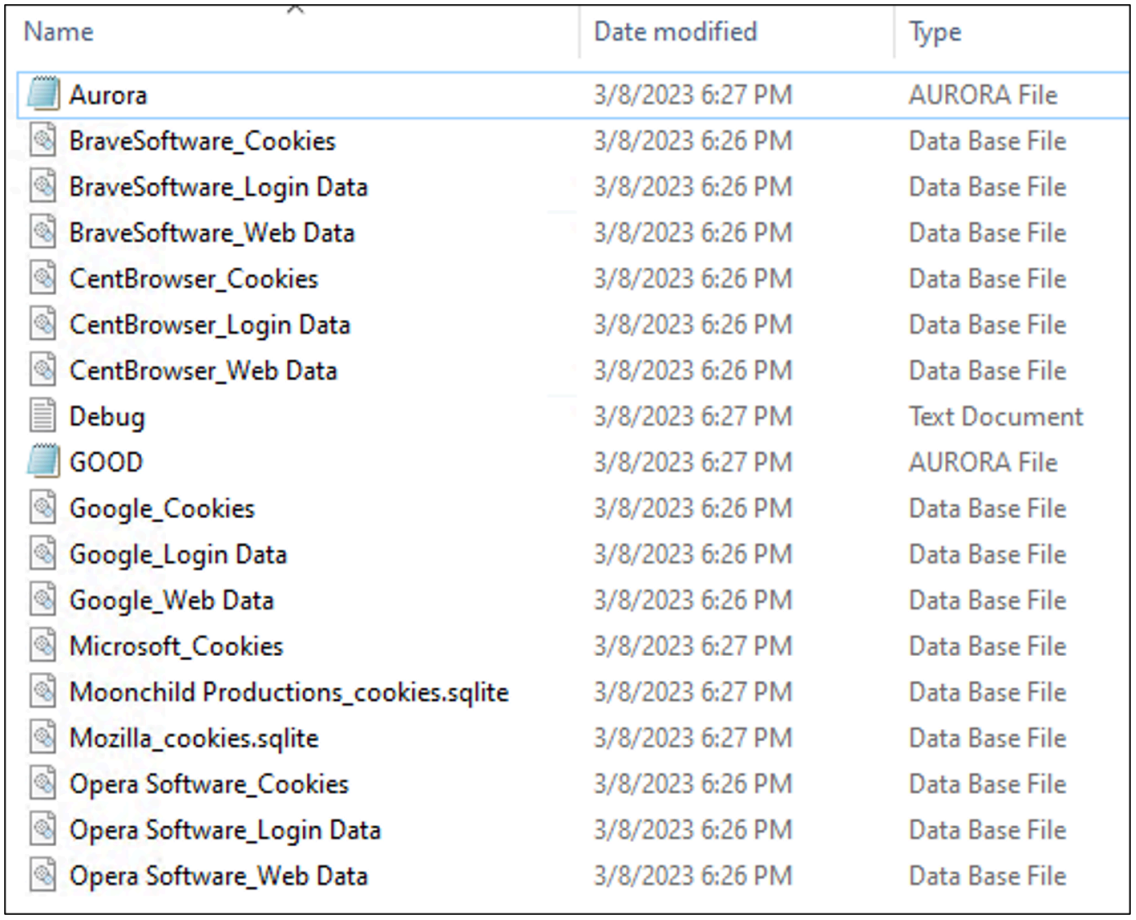


Figure 44: Stealer logs (UserInformation file)

The cache folder contains the database files extracted from the infected host with cookies and credentials in the encrypted format as well as debug logs (Figure 45).



Name	Date modified	Type
Aurora	3/8/2023 6:27 PM	AURORA File
BraveSoftware_Cookies	3/8/2023 6:26 PM	Data Base File
BraveSoftware_Login Data	3/8/2023 6:26 PM	Data Base File
BraveSoftware_Web Data	3/8/2023 6:26 PM	Data Base File
CentBrowser_Cookies	3/8/2023 6:26 PM	Data Base File
CentBrowser_Login Data	3/8/2023 6:26 PM	Data Base File
CentBrowser_Web Data	3/8/2023 6:26 PM	Data Base File
Debug	3/8/2023 6:27 PM	Text Document
GOOD	3/8/2023 6:27 PM	AURORA File
Google_Cookies	3/8/2023 6:26 PM	Data Base File
Google_Login Data	3/8/2023 6:26 PM	Data Base File
Google_Web Data	3/8/2023 6:26 PM	Data Base File
Microsoft_Cookies	3/8/2023 6:27 PM	Data Base File
Moonchild Productions_cookies.sqlite	3/8/2023 6:27 PM	Data Base File
Mozilla_cookies.sqlite	3/8/2023 6:27 PM	Data Base File
Opera Software_Cookies	3/8/2023 6:26 PM	Data Base File
Opera Software_Login Data	3/8/2023 6:26 PM	Data Base File
Opera Software_Web Data	3/8/2023 6:26 PM	Data Base File

Figure 45: Cache folder

The stealer can also be configured to send stealer logs via Telegram where CDD is the “Cookies Detected” and PDD is the “Passwords detected”.

The attacker(s) can also configure to receive the stealer logs via Telegram (Figure 46).

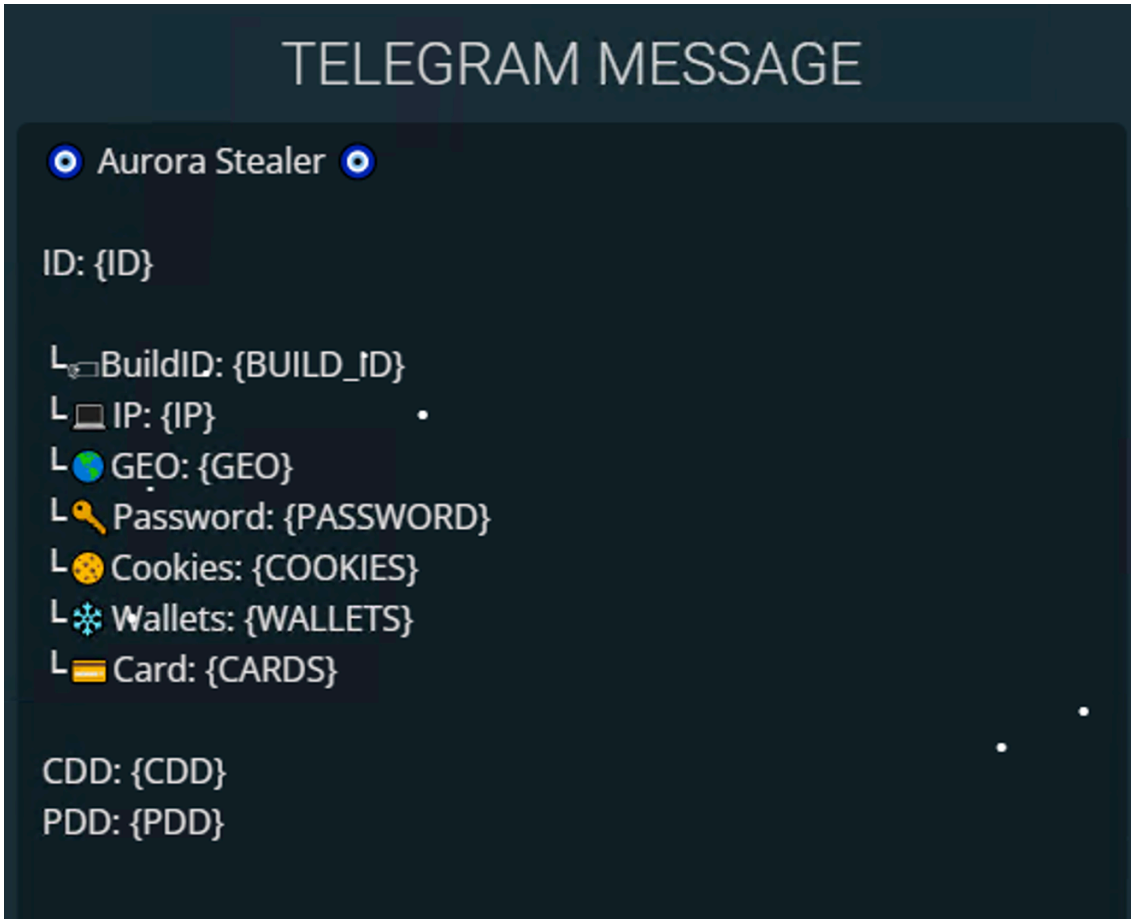


Figure 46: Telegram notification settings

How eSentire is Responding

Our Threat Response Unit (TRU) combines threat intelligence gained from research and security incidents to create practical outcomes for our customers. We are taking a comprehensive response approach to combat modern cybersecurity threats by deploying countermeasures, such as:

- Performing global threat hunts for indicators associated with Aurora Stealer.
- Implementing threat detections to identify malicious command execution and ensure that eSentire has visibility and detections are in place across eSentire MDR for Endpoint.

Our detection content is supported by investigation runbooks, ensuring our SOC (Security Operations Center) analysts respond rapidly to any intrusion attempts related to known malware Tactics, Techniques, and Procedures. In addition, TRU closely monitors the threat landscape, constantly addresses capability gaps, and conducts retroactive threat hunts to assess customer impact.

Recommendations from eSentire's Threat Response Unit (TRU)

We recommend implementing the following controls to help secure your organization against Aurora Stealer malware:

- Confirm that all devices are protected with Endpoint Detection and Response (EDR) solutions

- Implement a Phishing and Security Awareness Training (PSAT) Program that educates and informs your employees on emerging threats in the threat landscape.
- Encourage your employees to use password managers instead of using the password storage feature provided by web browsers. Use master passwords where it's applicable.

While the TTPs used by threat actor(s) grow in sophistication, they lead to a certain level of difficulties at which critical business decisions must be made. Preventing the various attack technique and tactics utilized by the modern threat actor requires actively monitoring the threat landscape, developing and deploying endpoint detections, and the ability to investigate logs & network data during active intrusions.

eSentire's TRU is a world-class team of threat researchers who develop new detections enriched by original threat intelligence and leverage new machine learning models that correlate multi-signal data and automate rapid response to advanced threats.

If you are not currently engaged with an MDR provider, eSentire MDR can help you reclaim the advantage and put your business ahead of disruption.

Learn what it means to have an elite team of Threat Hunters and Researchers that works for you. Connect with an eSentire Security Specialist.

Yara rule

```
rule  AuroraStealer {
    meta:
        author = "eSentire Threat Intelligence"
        description = "Detects the Build/Group IDs if present / detects an unobfuscated AuroraSteale
        date = "3/24/2023"

    strings:
        $b1 = { 48 8D 0D ?? ?? 04 00 E8 ?? ?? EF FF }
        $b2 = { 48 8D 0D ?? ?? 05 00 E8 ?? ?? EF FF }
        $ftp = "FOUND FTP"
        $go = "Go build ID"
        $machineid = "MachineGuid"

    condition:
        3 of them
}
```

MITRE ATT&CK

MITRE ATT&CK Tactic	ID	MITRE ATT&CK Technique	Description
MITRE ATT&CK Tactic Reconnaissance	ID T1592	MITRE ATT&CK Technique Gather Victim Host Information	Description During initial execution, Aurora Stealer gathers the information on the OS, processor name and video controller
MITRE ATT&CK Tactic Initial Access	ID T1189	MITRE ATT&CK Technique Drive-by Compromise	Description Aurora Stealer is delivered via a website hosting a fake software installer
MITRE ATT&CK Tactic Defense Evasion	ID T1027.001	MITRE ATT&CK Technique Binary Padding	Description Aurora Stealer contains the file pump feature upon creating the build to add null bytes to the stealer payload
MITRE ATT&CK Tactic Credential Access	ID T1555 T1555.003	MITRE ATT&CK Technique Credentials from Web Browsers	Description Aurora Stealer steals sensitive data from browsers including credentials, cookies and saved credit cards as well as FTP and RDP credentials
MITRE ATT&CK Tactic Discovery	ID T1082	MITRE ATT&CK Technique System Information Discovery	Description The stealer enumerates the host for hardware and geographical information as well as the screen size
MITRE ATT&CK Tactic	ID	MITRE ATT&CK Technique	Description

Collection	T1113	Screen Capture	The stealer takes the screenshot from the infected machine and sends it to the C2
MITRE ATT&CK Tactic Exfiltration	**ID** T1020	**MITRE ATT&CK Technique** Automated Exfiltration	**Description** The stealer automatically exfiltrates the gathered files to C2. File grabbing options can be customized by an attacker

Indicators of Compromise

Name	Indicators
Aurora Stealer	306fc85ff1c7e06f631c37d60d4ad98b
Aurora Stealer	da1548613d5fa9520931952675f92ca9
Aurora Stealer	16b349b80ef9e6d6a86e768b4e01fc4c
Aurora Stealer	aa349ad45bb48e85b5cd1b55308ae835353859219f28ece9685c8ae552e8e63a
C2	212.87.204.93:8081
C2	185.106.93.245:8081
C2	185.106.93.135:8081
C2	195.123.218.52:8081

Appendix

- https://www.esentire.com/security-advisories/increased-activity-in-google-ads-distributing-information-stealers
- https://twitter.com/1ZRR4H/status/1618136958596960256?s=20&t=UWEJ4jIxIg4XXv384Ibwow
- https://www.passcape.com/index.php?section=docsys&cmd=details&id=28#14
- https://learn.microsoft.com/en-us/windows/win32/api/dpapi/nf-dpapi-cryptprotectdata
- https://learn.microsoft.com/en-us/windows/win32/api/dpapi/nf-dpapi-cryptunprotectdata
- https://unit42.paloaltonetworks.com/credential-gathering-third-party-software/
- https://github.com/RussianPanda95/Configuration_extractors/blob/main/aurora_config_extractor.py

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.

Source: https://www.esentire.com/blog/esentire-threat-intelligence-malware-analysis-aurora-stealer