Meduza

TECHNICAL ANALYSIS REPORT

ZAYOTEM

ZARARLI YAZILIM ÖNLEME VE TERSİNE MÜHENDİSLİK

Contents

Summary

Meduza Stealer is a malicious program designed to target Windows users and entities. It is origin is unknown. Currently, it has only been reported to be active outside of ten specific countries. The main objective of Meduza Stealer is to steal comprehensive data. It collects various browser-related data by intercepting users' browsing activity. This data includes critical login information, valuable browsing history records, and carefully selected bookmarks. Crypto wallet extensions, password managers, and two-factor authentication applications are vulnerable to this threat.

This malware;

- Credentials saved in web browsers,
- Crypto wallet information saved in web browsers,
- Cookie information stored in web browsers,
- Password manager apps,
- Two-factor authentication applications,
- Information about registered Outlook accounts,
- System information on the computer,
- Credentials held by some applications on the computer,
- Computer documents,

Provides access.

Meduza.exe Analysis

Name	meduza.exe
MD5	C6068C2C575E85EB94E2299FC05CBF64
SHA256	0d0a4622c58f3f17d16fb5cbd0aa5403bc614ca58847b4a725f432d202a55454
File Type	PE64 / EXE

Overview

Meduza Stealer uses the **IsDebuggerPresent** API as an anti-debug technique before starting it is malicious actions. The purpose of this is to make the analyst's job harder. It then checks the **country codes** and the **Windows operating system version**. If the checks are satisfied, the functions that perform the actual malicious operations are called. The purpose of the malware includes collecting system information, browser data, password manager details, mining-related registry information, and details about installed applications. Once all of this detailed information is collected, it is packaged. It is ready to be uploaded to the attacker's command and control server. Once complete, the program deletes itself in the background, ending it is malicious activities.

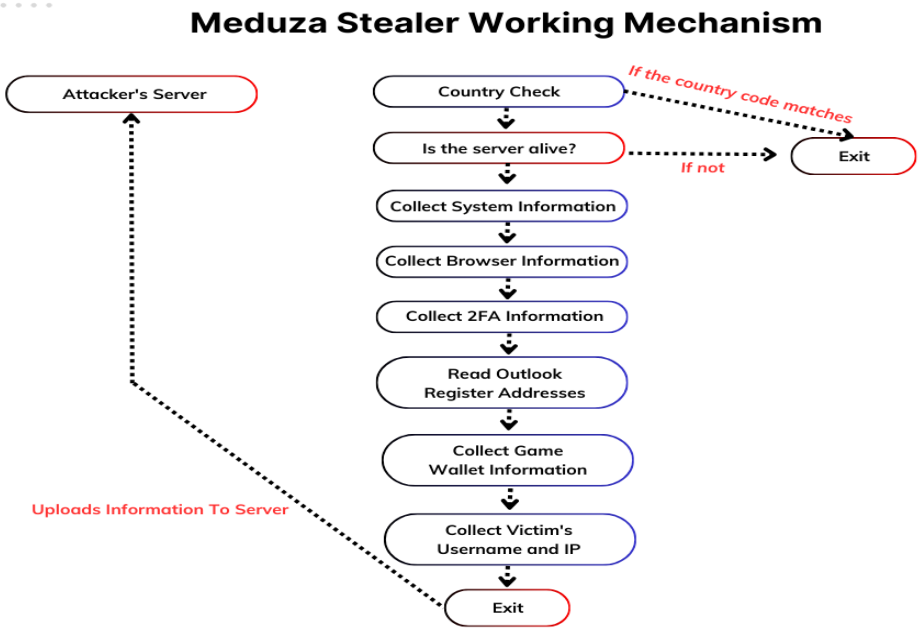

Image 1 – How The Malware Works

The following countries have implemented language control to prevent the malware from running:

Russia	Armenia	Belarus
Kazakhstan	Uzbekistan	Tajikistan
Moldova	Kyrgyzstan	Turkmenistan
Georgia		

Image 2 – Table of Countries with Language Control

The malware targets password managers and two-factor authentication applications:

GAuthAuthenticator	Authenticator	SafePal	Guarda
EOS Authenticator	BrowserPass	KeePassXC	1Password
Trezor Password Manager	Dashlane	Bitwarden	LastPass
Keeper	Nordpass	RoboFrom	Splikity
MYKI	Zoho Vault	Authy	

Image 3 – Password Managers and Two-factor authentication List Targeted by the Malware

Desktop applications targeted by the malware:

Discord	Telegram	Jaxx_Liberty

Image 4 – List of Desktop Applications Targeted by the Malware

Browsers targeted by the malware:

Microfost Edge	Mozilla Firefox	Pale Moon	Suhba	RockMelt
Google Chrome	Chromium	Amigo	QQBrowser	Vivaldi
CrytoTab Browser	TorBro Browser	Cent Browser	Opera	Brave Old
Chedot Browser	Torch	7Star	Tencent	OperaGX
Privacy Browser	Yandex Browser	360 Browser	Orbitum	Xpom
Comodo Dragon Epic	Opera Browser	SalamWeb	Kinza	Xvast
Nichrome	Slim Browser	Chromodo	Go Browser	Maxthon
Mail.Ru Atom	CocCoc Browser	Coowon		

Image 5 – List of Browsers Targeted by the Malware

Crypto wallets targeted by the malware:

Electrum	Electrum-LTC	Exodus	ElektronCash	MultiDoge
Jaxx_Desktop_Old	Atomic	Binance	Coinomi	Monero
TronLink	MetaMask	Wasabi Wallet	Yoroi	DashCore
Niftywallet	Mathwallet	Coinbase	Guarda	EQUALWallet
JaxxLiberty	BitAppWallet	iWallet	Wombat	MeWCx
Guidwallet	RoninWallet	Neoline	CloverWallet	Liqualitywallet
Terra Station	Keplr	Sollet	AuroWallet	PolymeshWallet
ICONex	Harmony	Coin98	EVER Wallet	KardiaChain
Rabby	Phantom	BraveWallet	Atomic	Paliwallet
Boltx	Xdefiwallet	NamiWallet	MaiarDeFiWallet	Goby
Solflare	Cyanowallet	TezBox	Temple	
BinanceChainWallet	Blockstream Green	Daedalus	Waveskeepe	

Image 6 – List of Crypto Wallets Targeted by the Malware

System details collected by the malware:

System Build Details	**Computer Name**
CPU Details	**Execute Path**
Geo	**GPU**
Hardware ID Details	**Public Ip**
OS Details	**RAM Details**
Screen Resolution Details	**Screenshot**
Time	**Time Zone**

Image – 7 Malware Collected System Details

Dynamic Analysis

Before executing any malicious activity, the malware uses the **IsProcessorFeaturePresent** API to determine if the device is running on Windows 7 or an older version.

```
BOOL __fastcall sub_13F09CE8C(DWORD64 a1)
{
  DWORD64 retaddr; // [rsp+38h] [rbp+0h]
  DWORD64 v3; // [rsp+40h] [rbp+8h] BYREF

  v3 = a1;
  if ( IsProcessorFeaturePresent(0x17u) )
    __fastfail(2u);
  capture_previous_context(&ContextRecord);
  ContextRecord.Rip = retaddr;
  ContextRecord.Rsp = (DWORD64)&v3;
  qword_13F0DA790 = retaddr;
  ContextRecord.Rcx = v3;
  dword_13F0DA780 = -1073740791;
  dword_13F0DA784 = 1;
  dword_13F0DA798 = 1;
  unk_13F0DA7A0 = 2i64;
  return _raise_securityfailure((struct _EXCEPTION_POINTERS *)&ExceptionInfo);
}
```

Image 8 – Detecting the Version of the Operating System

The malware first acquires the computer's processor and architecture information.

```
00000013FB3EC04  C5FE6F52 20          vmovdqu ymm2,yword ptr ds:[rdx+20]      rdx+20:L"OM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC"
00000013FB3EC09  C5FE6F5A 40          vmovdqu ymm3,yword ptr ds:[rdx+40]      rdx+40:L"D;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC"
00000013FB3EC0E  C5FE6F62 60          vmovdqu ymm4,yword ptr ds:[rdx+60]      rdx+60:L".JSE;.WSF;.WSH;.MSC"
00000013FB3EC13  C5FD7F09             vmovdqa yword ptr ds:[rcx],ymm1
00000013FB3EC17  C5FD7F51 20          vmovdqa yword ptr ds:[rcx+20],ymm2
00000013FB3EC1C  C5FD7F59 40          vmovdqa yword ptr ds:[rcx+40],ymm3      rcx+40:L"`1"
00000013FB3EC21  C5FD7F61 60          vmovdqa yword ptr ds:[rcx+60],ymm4
00000013FB3EC26  C5FE6F8A 80000000    vmovdqu ymm1,yword ptr ds:[rdx+80]      rdx+80:L"MSC"
00000013FB3EC2E  C5FE6F92 A0000000    vmovdqu ymm2,yword ptr ds:[rdx+A0]      rdx+A0:L"CHITECTURE=AMD64"
00000013FB3EC36  C5FE6F9A C0000000    vmovdqu ymm3,yword ptr ds:[rdx+C0]
00000013FB3EC3E  C5FE6FA2 E0000000    vmovdqu ymm4,yword ptr ds:[rdx+E0]      rdx+E0:L"IFIER=Intel64 Family 6 Model 165 Stepping 2, GenuineIntel"
00000013FB3EC46  C5FD7F89 80000000    vmovdqa yword ptr ds:[rcx+80],ymm1
00000013FB3EC4E  C5FD7F91 A0000000    vmovdqa yword ptr ds:[rcx+A0],ymm2
```

Image 9 – Getting Information about Processor and Architecture

The first step Meduza Stealer performs when it successfully infiltrates a machine is to check the victim's geolocation against a predefined list (see Image 2). If the victim's geolocation is included, the malware does not run.

```
00000013FB2E761  74 1D                je medusa.13FB2E780
00000013FB2E763  48:8BD7              mov rdx,rdi                             rdx:"RU", rdi:"RU"
00000013FB2E766  48:8BCB              mov rcx,rbx                             rcx:"RU"
00000013FB2E769  E8 72070000          call <medusa.country_code_check_list>
00000013FB2E76E  48:83C3 20           add rbx,20
00000013FB2E772  48:895C24 30         mov qword ptr ss:[rsp+30],rbx           [rsp+30]:"RU"
00000013FB2E777  48:83C7 20           add rdi,20                              rdi:"RU"
00000013FB2E77B  48:3BFD              cmp rdi,rbp
00000013FB2E77E  75 E3                jne medusa.13FB2E763
00000013FB2E780  48:895C24 28         mov qword ptr ss:[rsp+28],rbx           [rsp+28]:"RU"
00000013FB2E785  48:8BD3              mov rdx,rbx                             rdx:"RU"
00000013FB2E788  48:8BCB              mov rcx,rbx                             rcx:"RU"
00000013FB2E78B  E8 60320000          call medusa.13FB319F0
```

Image 10 – Acquiring Country Controls for Specific Countries using Country Codes

The malware checks for server availability (as shown in image 11). If the server is up, it proceeds with malicious operations. If not, it terminates them.

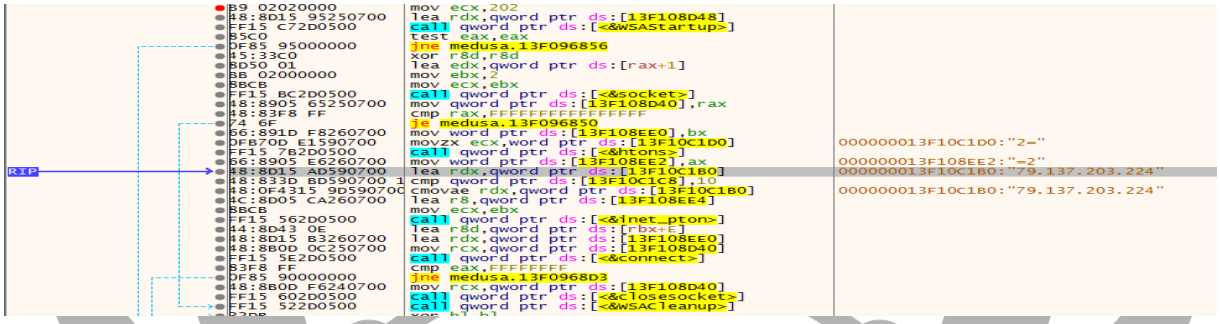

Image 11 – Getting the Connection to the IP Address Requested by the Malware

At the following address, the panel redirects the request to **/auth/login**.

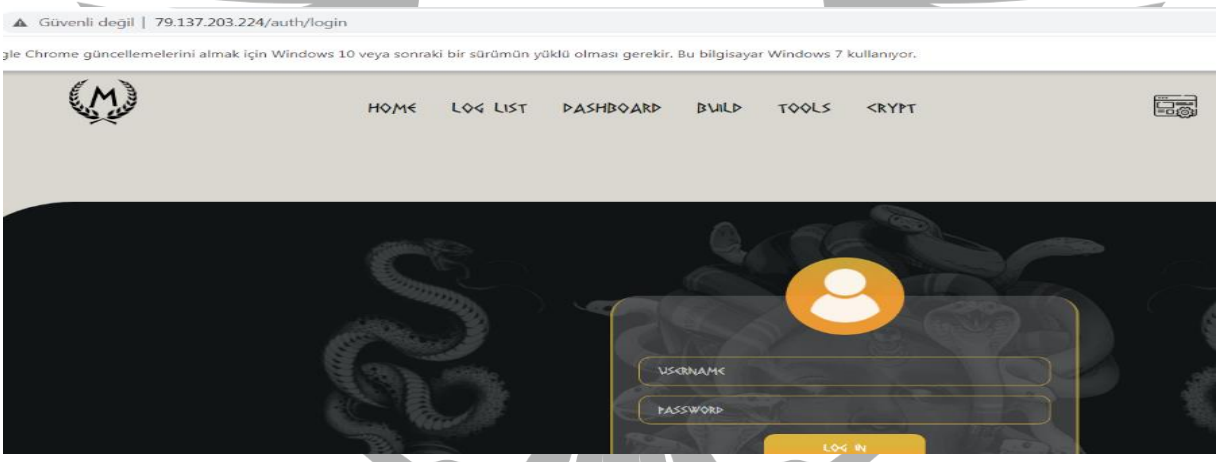

Image 12 – Find the Web Panel Address Used by the Malware to Communicate

The malware uses **EnumDisplayDevices** API which is a malicious code that retrieves data on the display devices being used in the current session.

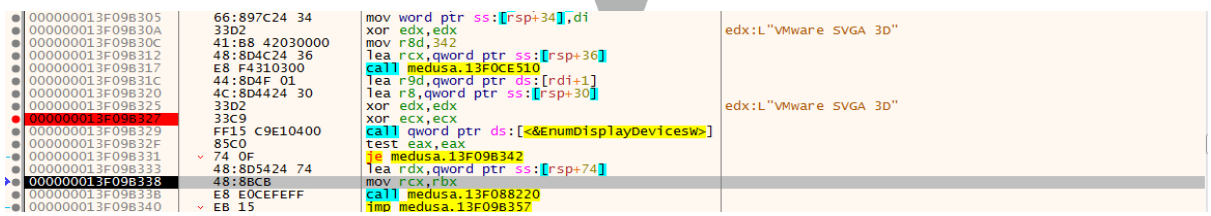

Image 13 – Getting Information about Imaging Devices

The malware that targets crypto assets starts by identifying the specific crypto wallet, which can be either a browser plugin or a hardware device. It then searches for the target coin within the wallet and retrieves the name of the cryptocurrency, the wallet name and the name of the file associated with the wallet as parameters.

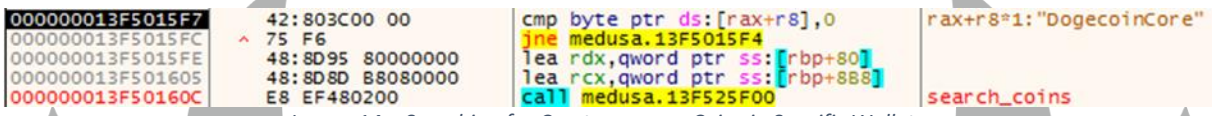

Image 14 – Searching for Cryptocurrency Coins in Specific Wallets

To create a new folder named "coin" inside the "crypto" folder, the **CreateDirectoryA** API is utilized. The directory of the APPDATA folder is obtained using the **SHGetFolderPathA** API, and the wallet name is added as a directory using **lstrcatA**. This results in the absolute directory of the wallet, where all the wallet data is stored. All the cryptocurrency-related data is then copied into the "crypto" folder.

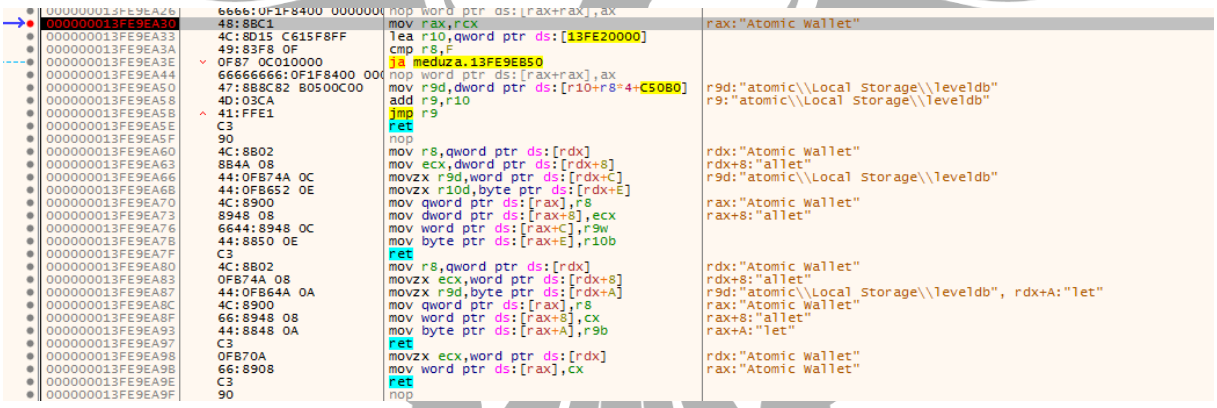

Image 15 – Get Crypto Wallets Targeted by the Malware

The malware acquires cookies from all the plugins installed in the browser.

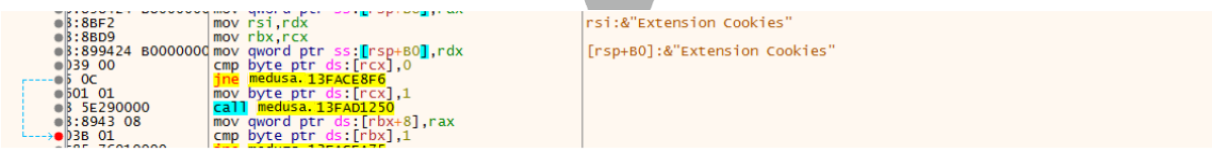

Image 16 – Getting Cokkies From Browser Plugins

In addition, the malware accesses network cookies stored in the Chrome browser.

Image 17 – Getting Cokkies Stored from Chrome Browser

After completing the scanner operations, the malware program targets Outlook data. At this stage, it attempts to obtain handles for registry directories that are specified as parameters with **KEY_READ** permission for **HKEY_CURRENT_USER** by using the **RegOpenKeyExA** API in Outlook registry addresses found in the Windows Registry. If the returned value is ERROR_SUCCESS, the **RegEnumValueA** API is called, with the handle and char array variable provided for writing the data in addition to the default. This call is set in the while loop in Imagede, which will be the value of the return value. It will continue to work as long as the returned value is ERROR_SUCCESS.

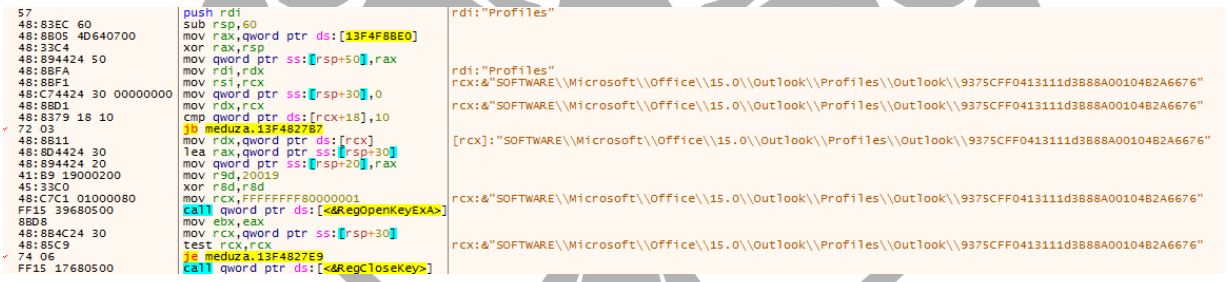

Image 18 – Getting the Address of the Location Where the Malware Intends to Take the Handles

- SOFTWARE\\Microsoft\\Windows Messaging Subsystem\\Profiles\\9375CFF0413111d3B88A00104B2A6676
- SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Windows Messaging Subsystem\\Profiles\\Outlook\\9375CFF0413111d3B88A00104B2A6676

Image 19 – Registry Addresses Targeted by the Malware

After reading the registration addresses, the malware contacts the C2 server. But before that, it sends a request to api[.]ipify[.]org with the **InternetOpenUrlA** API to return the victim's public IP.

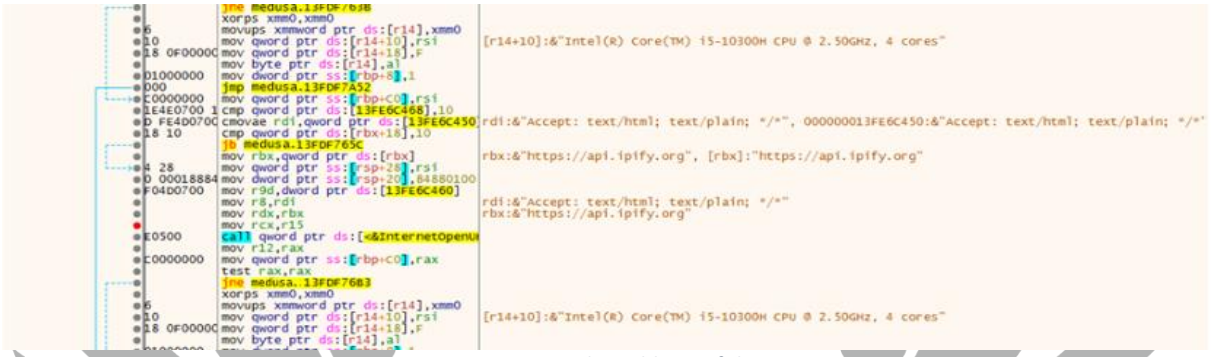

Image 20 – Getting the Public IP of the Device

The malware uses the **RtlGetVersion** and **GetNativeSystemInfo** APIs to obtain information about the system and it is version.

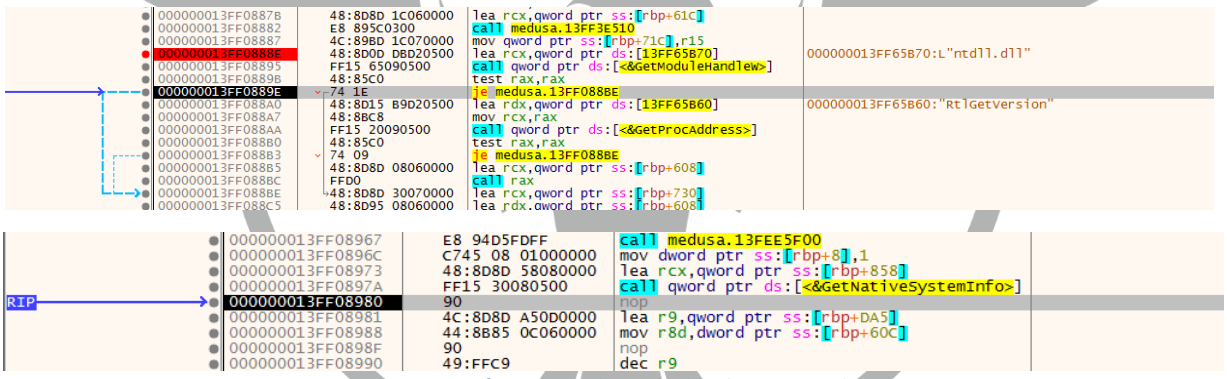

Image 21 – Acquiring information about the local system and it is version.

The malware obtains the name of the victim's machine using the **GetComputerName** API.

Image 22 – Getting the Victim Device Name

In addition, the malware collects GPU, RAM and other system information in Image-7.

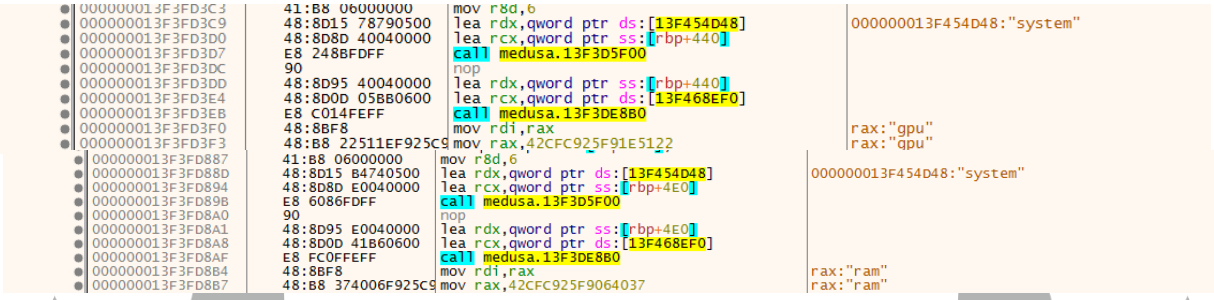

Image 23 – Getting System Information

The Telegram application and registry key on the target computer are controlled through the InstallLocation value.

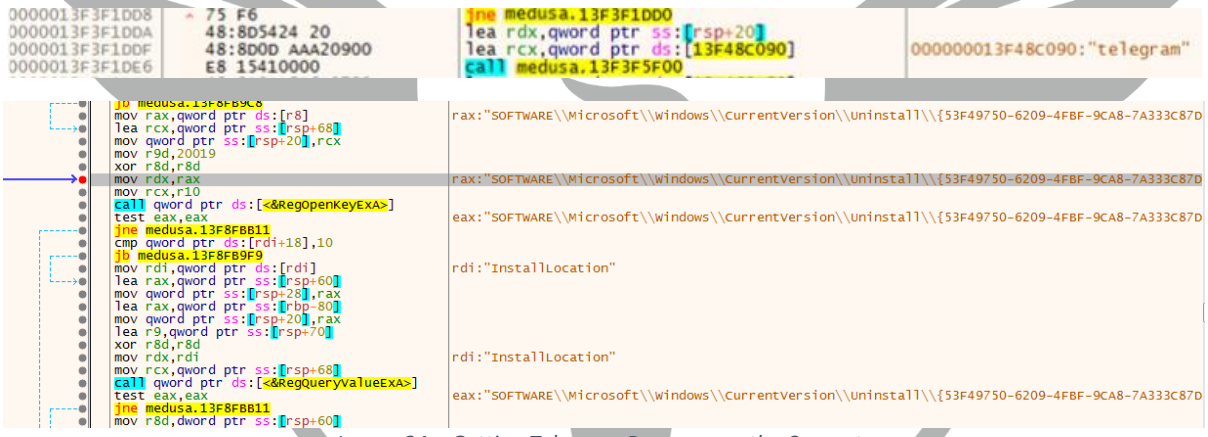

Image 24 – Getting Telegram Presence on the Computer

Time zone information is retrieved by accessing the **SYSTEM\CurrentControlSet\Control\TimeZoneInformation** registry key and calling the **TimeZoneKeyName** API.

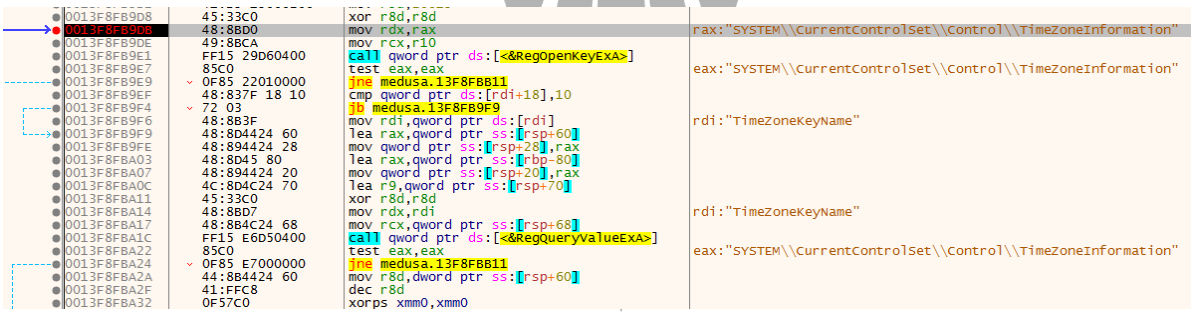

Image 25 – Getting Time Zone Information On Computer

The malware aims to access some subroutines that run specifically for the Discord application and the accounts.xml file where discord user accounts are kept. It also wants to access the database file of the desktop application of the liberty jaxx wallet.

Image 26 – Getting Some Desktop Applications That The Malware Wants To Access

The malware obtains Steam client data by reading the registry key "**\SOFTWARE\Valve\Steam**". Steam is a digital distribution platform created by Valve Corporation and used primarily for video games. This registry key stores user-specific settings, game information, login data, session information, and other configuration data associated with the Steam client.

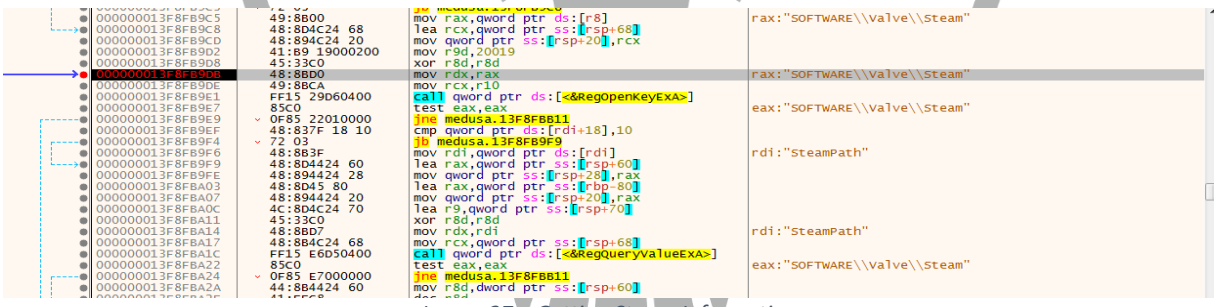

Image 27 – Getting Steam Information

Then, the malware collects users' profile photos in the Chrome browser.

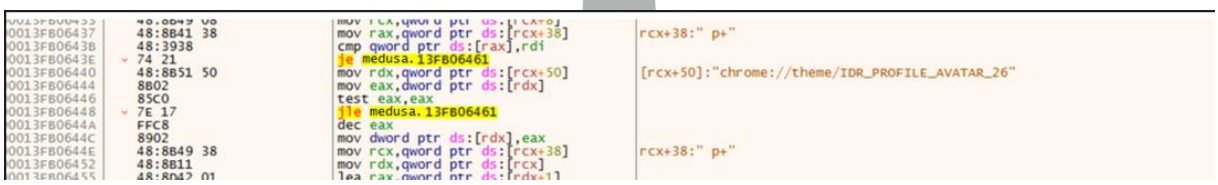

Image 28 – Getting User Profile Photos

The malware tries to obtain information of locked user profiles

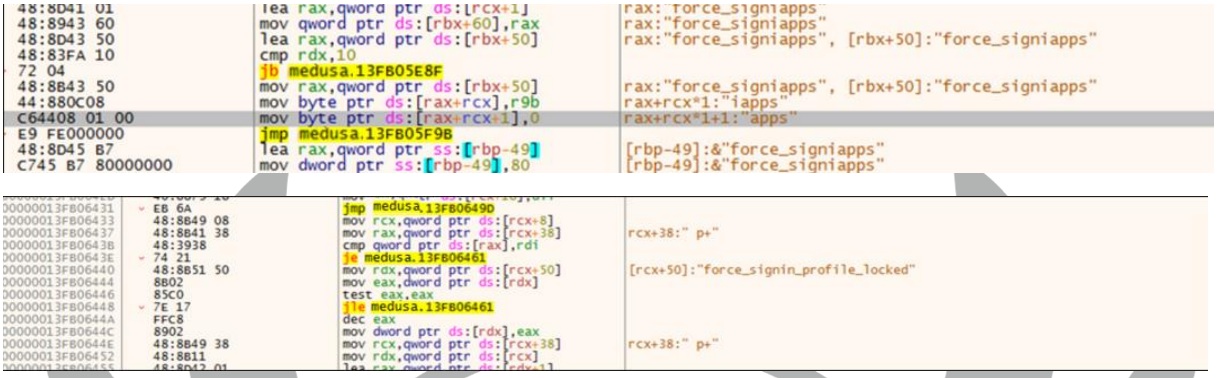

Image 29 – Getting User Profiles

The malware performs a three way handshake via port **15666** as a result of the request sent to IP **79[.]137[.]137[.]203[.]224**. With this communication, the data is uploaded to the server in an encrypted way.

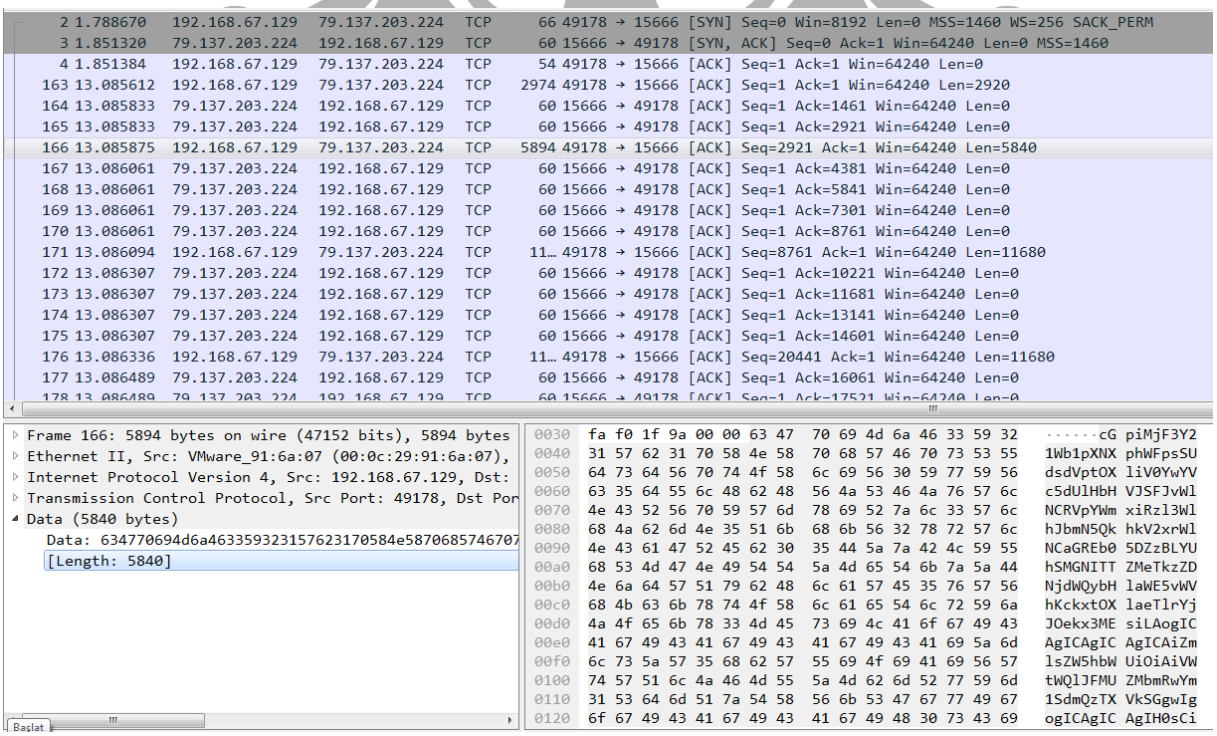

Image 30 – Getting Malware's Connection to the Attacker Server

When the encrypted data of the malware is analyzed, it is observed that it is encrypted again in BASE64 format.


```
            "content": "RGVuZW1lWWF6aXNp",
            "filename": "dGVzdC50eHQ="
        },
        {
            "content": "
S3VsbGFuaWNpIEFkaTphZG1pbgpTaWZyZTphZG1pbgoKS3VsbGFuaWNpIEFkaTpkZXN0cm95ZXIKU2lmcmU6MTIzNDU2Cg==
            "filename": "c2lmcmVsZXIudHh0"
        }
    ],
    "chromium_browsers": {
        "Google Chrome": {
            "Default": {
                "Extension Cookies":
```


Image 31 – Getting Malware's Encrypted Data

When the encrypted data was analyzed again, it was found that the victim computer's data had been compromised.


```
            "content": "DenemeYazisi",
            "filename": "test.txt"
        },
        {
            "content": "
Kullanici Adi:admin
Sifre:admin
Kullanici Adi:destroyer
Sifre:123456
            "filename": "sifreler.txt"
        }
    ],
    "chromium_browsers": {
        "Google Chrome": {
```


Image 32 – Getting Decrypt Data from of Encrypted Hashes

Finally, the malware uses the **GetModuleFileNameA** API to get the location of the given executable. It then opens the command client with the **ShellExecuteA** API and executes the command in Image 34.

Image 33 – The Malware Deletes Itself After Completing It is Process

With the following cmd script, the **Nul** command sends a packet to the IP address 1[.]1[.]1[.]1 without displaying any output on the screen. The script also sets a timeout of 3 seconds, after which the **Del** command is executed. Upon execution, the malware deletes itself, thereby terminating all it is operations.

```
ping 1.1.1.1 -n 1 -w 3000 > Nul & Del /f /q \"C:\\Users\\*\\Desktop\\medusa.exe"
```

Image 34 – The Command the Malware Executes When Deleting Itself

YARA Rule

```
rule Medusa {

        meta:

                description = "MedusaStealer"

        strings:

                $wallet1 = "\\Electrum\\wallets\\"

                $wallet2 = "\\atomic\\Local Storage\\leveldb\\"

                $wallet3 = "\\WalletWasabi\\Client\\Wallets\\"

                $wallet4 = "Coinomi\\Coinomi\\wallets"

                $wallet5 = "\\Exodus\\exodus.wallet\\"

                $wallet6 = "\\com.liberty.jaxx\\IndexedDB\\file__0.indexeddb.leveldb\\"

                $wallet7 = "\\Metamask\\"

                $k1 = "SOFTWARE\\Microsoft\\Windows Messaging
Subsystem\\Profiles\\9375CFF0413111d3B88A00104B2A6676"

                $k2 = "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Windows Messaging
Subsystem\\Profiles\\Outlook\\9375CFF0413111d3B88A00104B2A6676"

                $x1 = "DiscordDevelopment\\accounts.xml"

                $x2 = "Ethereum\\keystore"

                $x3 = "User Data\\Extension Cookies"

                $x4 = "Web Data"

                $x5 = "Login Data"

                $x6 = "DiscordPTB"

                $x7 = "DiscordCanary"

                $ip = "79.137.203.224"
```

```
        $c1 = "Bitcoin"
        $c2 = "Ethereum"
        $c3 = "Armory"
        $c4 = "bytecoin"
        $c5 = "LiteCoin"
        $api1 = "EnumDisplayDevicesA"
        $api2 = "GdipCreateBitmapFromHBITMAP"
        $api3 = "GetUserDefaultLocaleName"
        $api4 = "CryptoMsgDllCNGExportKeyFree"
        $api5 = "GdipSaveImageToStream"
        $api6 = "InternetReadFile"
        $api7 = "WSAStartup"
        $api8 = "InternetOpenUrlA"
        $api9 = "HttpQueryInfoW"
        $api10 = "InternetQueryDataAvailable"
        $api11 = "IsDebuggerPresent"
    condition:
        all of them or
        4 of ($wallet*) and 3 of ($c*) or
        4 of ($wallet*) and 3 of ($api*) or
        2 of ($wallet*) and all of ($k*) and all of ($x*) and $ip
}
```

MITRE ATTACK TABLE

Collection	Execution	Discovery	Defense Evasion	Credential Access	C&C	Exfliration
Data from Local System (T1005)	**Windows Command Shell** (T1059.003)	**File and Directory Discovery** (T1083)	**Debugger Evasion** (T1622)	**Credentials from Web Browsers** (T1555.003)	**Standard Encoding** (T1132.001)	**Exfliration Over C2 Channel** (T1041)
		Query Registry (T1012)	**Deobfuscate/ Decode Filesor Information** (T1140)	**Steal Web Session Cookie** (T1539)		
		System Information Discovery (T1082)				

Security Takeaways

1. An up-to-date antivirus program should be used.
2. Passwords should not be stored in clear text on the computer.
3. Unknown applications should not be run without checking.
4. Two-factor authentication should be used in crypto wallets.
5. More reliable cryptocurrency storage methods such as cold wallets should be preferred.
6. Attached files of unknown e-mails should not be opened.
7. Do not click on links that are not from reliable sources.
8. The applications used should be kept up to date.

PREPARERS

Akif İnan Yiğit	LinkedIn
Halit Düzgün	LinkedIn
Mehmet Özen	LinkedIn
Ömer Faruk Berber	LinkedIn