ECHO

CYBER THREAT INTELLIGENCE

RACCOONSTEALER V2.0

TECHNICAL ANALYSIS REPORT

Contents

Executive Summary 2
File.exe Analysis 3
General Review 3
Stage 2 Analysis 4
DLL Detection 4
Process Detection 5
Computer Name Detection 6
Username Detection 7
InstallUtil.exe Analysis 8
General Review 8
Dynamic Analysis 8
Getting API Funtion Address 8
String Resolving Algorithm 9
Process Access Detection 9
Creation of Request Contents 12
Network Analysis 13
Request Analysis 13
After Respons 15
Gathering Device Information 15
DLL Loading 19
Database Operations 19
File Traversal Algorithm 20
Additional Analysis 21
SQL Queries 21
YARA Rule 22
MITRE ATTACK TABLE 23
Mitigations 23

Executive Summary

Raccoon Stealer V2.0 is an advanced version of the malicious software Raccoon Stealer. This malware is used by hackers to steal sensitive information from target computers and use personal information for malicious purposes.

Features of Raccoon Stealer V2.0 include the ability to steal users' browser histories, cookies, logins, and other sensitive information. The effects of Raccoon Stealer V2.0 can pose a serious threat to organisations. This malware can seriously compromise organisations' data security, damage their reputation and cause financial losses.

Organisations can protect against threats such as Raccoon Stealer V2.0 by adopting a strong security strategy and raising user awareness. Keeping security software up to date, using strong encryption methods and conducting regular security audits can help minimise the impact of this type of malware.

To detect and mitigate Raccoon Stealer V2.0, organisations should seek support from security experts and security software, and be prepared to deal with threats through continuous monitoring and updates.

File.exe Analysis

General Review

SHA 256	1976859574585aac13a24b6696cec26479029a92334c721ec71492094a7edec3
Name	file.exe
File Type	PE32-EXE

Table 1 file.exe file information

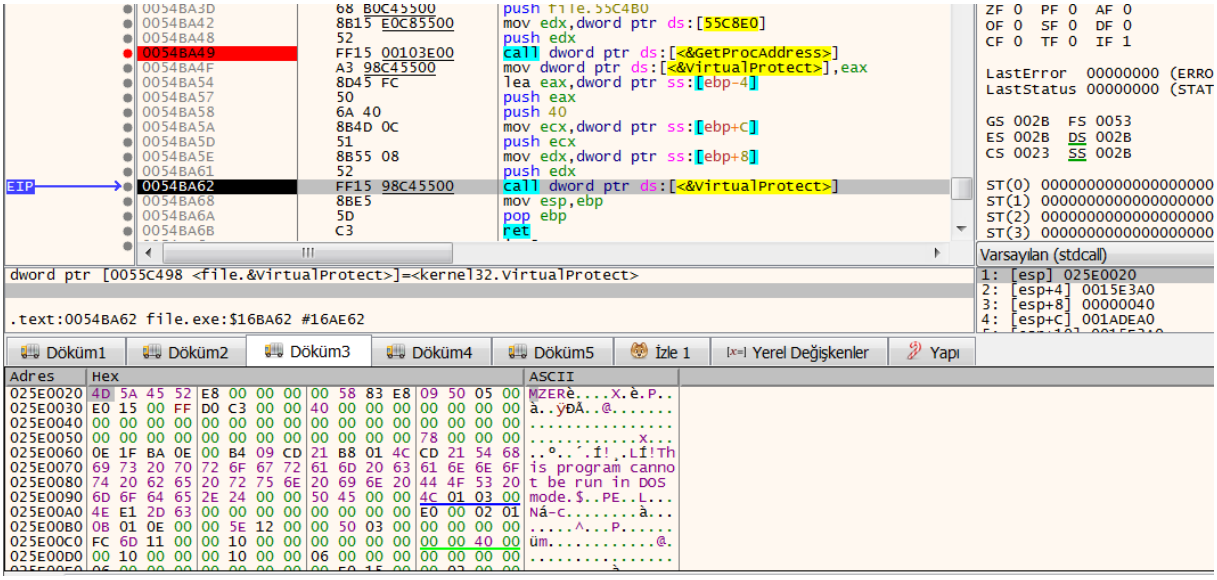

Figure 1 Obtaining executable permission for the area where the analysed code is written

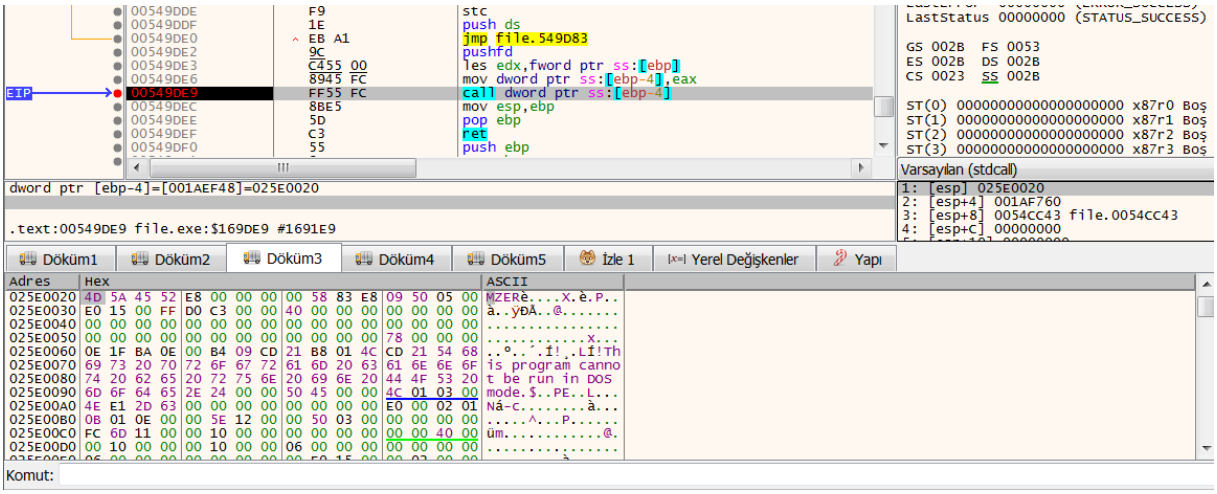

Figure 2 Call to the starting address of the decoded code

Malware has been detected and unpacked.

Stage 2 Analysis

At this stage, it was determined that malware applied analysis detection techniques.

DLL Detection

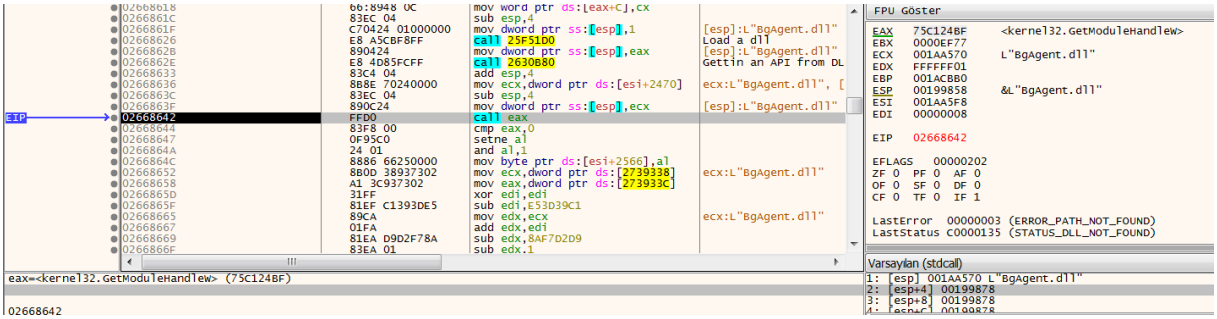

Figure 3 Determination of the existence of the dynamic library names analysed at runtime on the computer

It has been observed that the malware tries to detect DLL files belonging to some security products and systems. The DLLs it tries to detect and the systems they belong to are as follows:

CWSandbox	api_log.dll
	dir_watch.dll
	pstorec.dll
Sandboxie	sbieDll.dll
ThreatExpert	dbghelp.dll
Comodo	cmdvrt32.dll /cmdvrt64.dll
BullGuard	BgAgent.dll

Table 2 Names of the checked dynamic library files and the systems they belong to

Process Detection

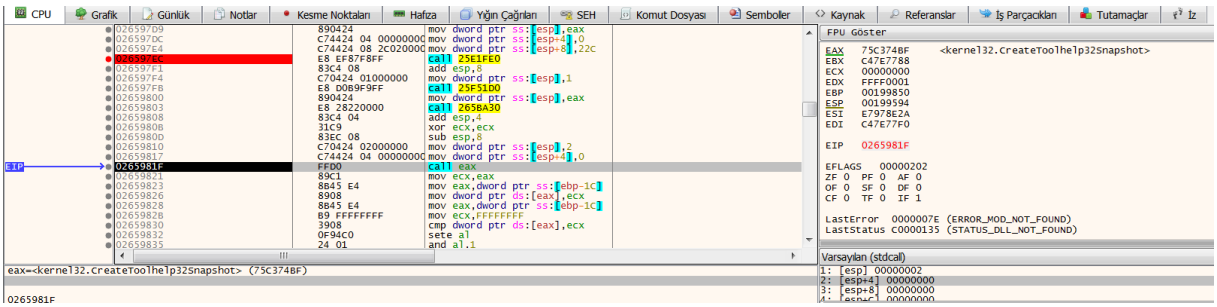

Figure 4 Taking a snapshot of the processes running in the background

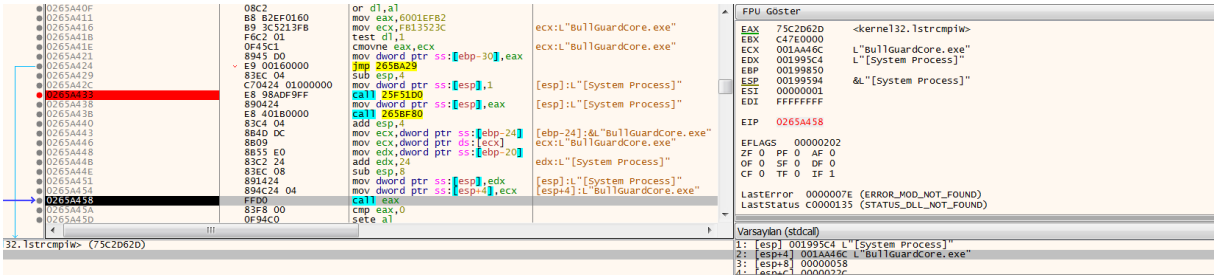

Figure 5 Process Blacklist Control

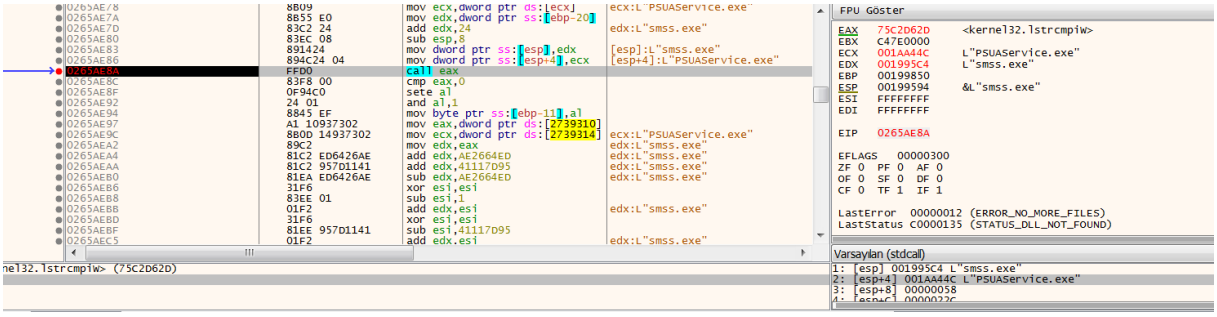

Figure 6 Process Blacklist Control

It was observed that the malware compares the processes running in the background with its own blacklist. The process list compared is as follows:

- fmon.exe
- WRSA.exe
- PSUAService.exe
- BullGuardCore.exe

Computer Name Detection

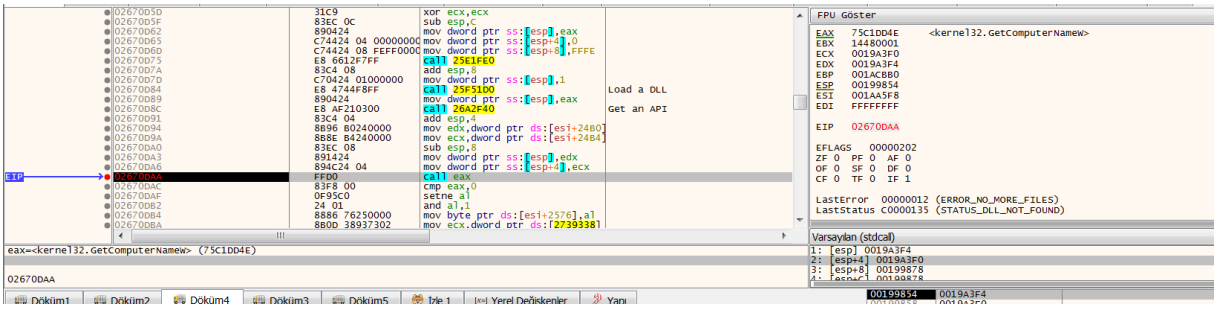

Figure 7 Withdrawing the victim's computer name

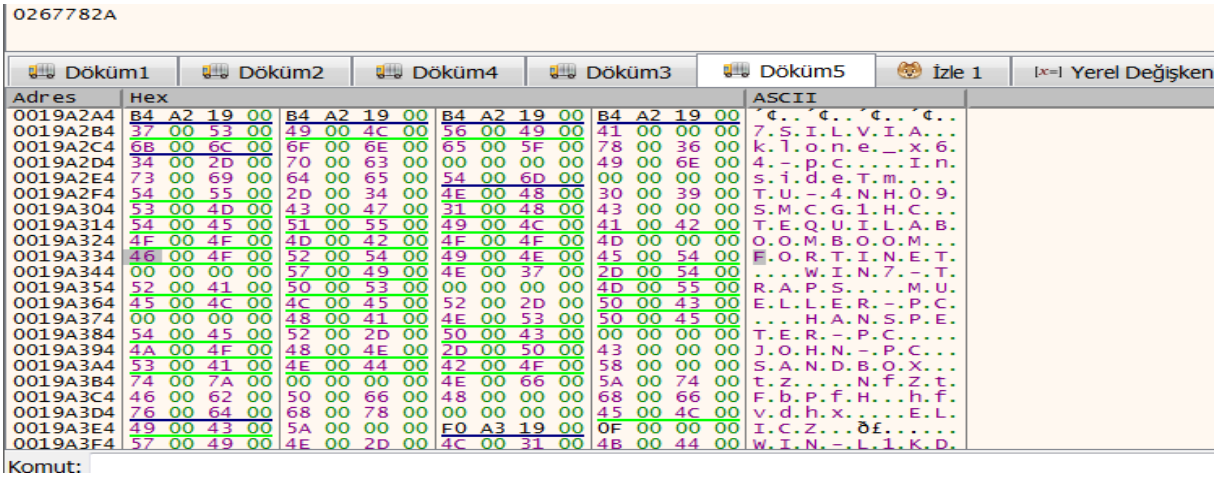

Figure 8 Black list of computer names after analyses

Figure 9 Comparing the computer names in the list with the victim computer name in order

It was observed that malware analyses a list of names and compares it with the name of the computer on which it was found. Analysed computer names:

SANDBOX	JOHN-PC	HANSPETER-PC	MUELLER-PC
WIN7-TRAPS	FORTINET	TEQUILABOOMBOOM	TU-4NH09SMCG1HC
InsideTm	klone_x64-pc	7SILVIA	tz
NfZt	FbPfH	hfvdhx	ELICZ

Table 3 Analysed computer names

Username Detection

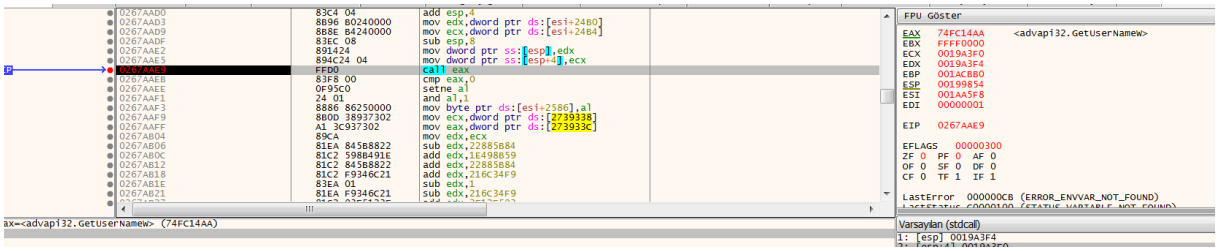

Figure 10 User name retrieval process

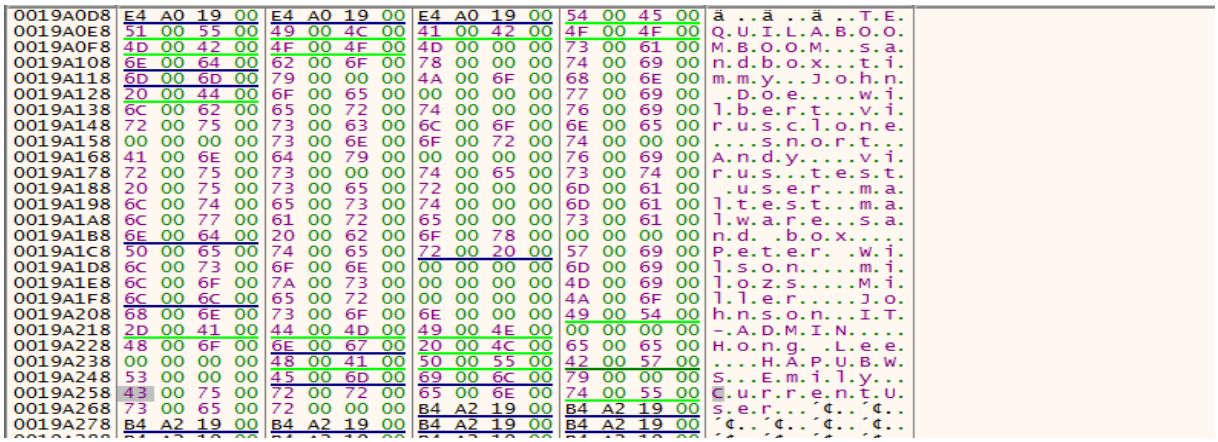

Figure 11 Post-analysis username blacklist

Figure 12 The process of comparing the user names in the list

The malware was also found to analyse the username list and compare it with the username of the computer it was found on. Compared usernames:

CurrentUser	sandbox	Emily	HAPUBWS
Hong Lee	IT-ADMIN	Johnson	Miller
TEQUILABOOMBOOM	milozs	Peter Wilson	sand box
malware	maltest	test user	virus
Andy	snort	virusclone	wilbert
virusClone	John Doe	timmy	

Table 4 Checked user names

It was detected that the malware executes malicious code with the ProcessHollowing technique in the executable file **"C:\Windows\Microsoft.NET\Framework64\v4. 0.30319\InstallUtil.exe"**.

InstallUtil.exe Analysis

General Review

SHA256	6052F7D7832F6EDDF1BA8309F189FCCCB9917128D216FC1C181327B3DEBDEDAC
Name	InstallUtil.exe
File Type	PE32-EXE

Table 5 InstallUtil.exe file information

Dynamic Analysis

Getting API Funtion Address

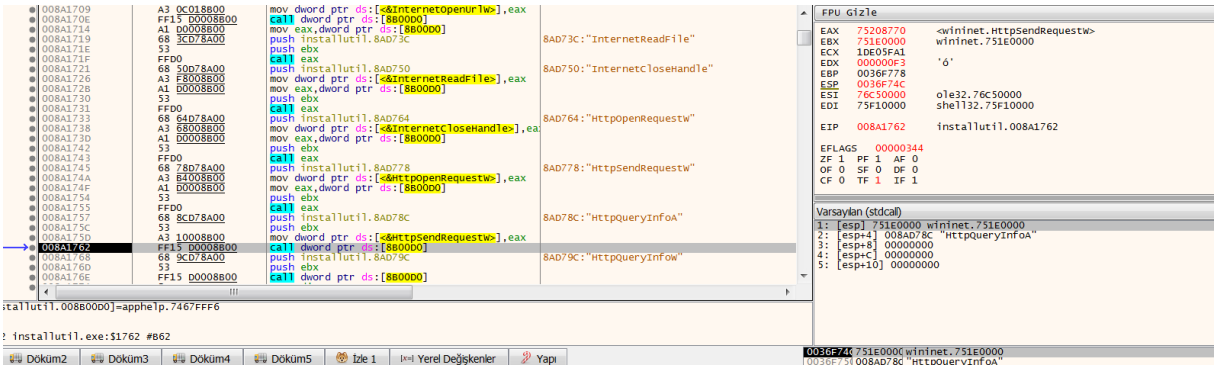

Figure 13 Pulling the addresses of the APIs to be used

It has been determined that it has received the addresses of the API Functions it will use. The functions whose addresses he received are as follows:

GetFileSize	GetDriveType	GetFileSize	GetDriveType
GetModuleFileNameW	GetSystemInfo	GetModuleFileNameW	GetSystemInfo
wideCharToMultiByte	ShellExecuteW	wideCharToMultiByte	ShellExecuteW
PathMatchSpecW	InternetReadFile	PathMatchSpecW	InternetReadFile
HttpSendRequestW	HttpQueryInfoA	HttpSendRequestW	HttpQueryInfoA

Table 6 APIs whose addresses were retrieved

Figure 14 Mutex Creation

It was also detected that malware created a mutex named **"264782971_qJ5tS2bD5fD1nZ5kD2kV"**.

String Resolving Algorithm

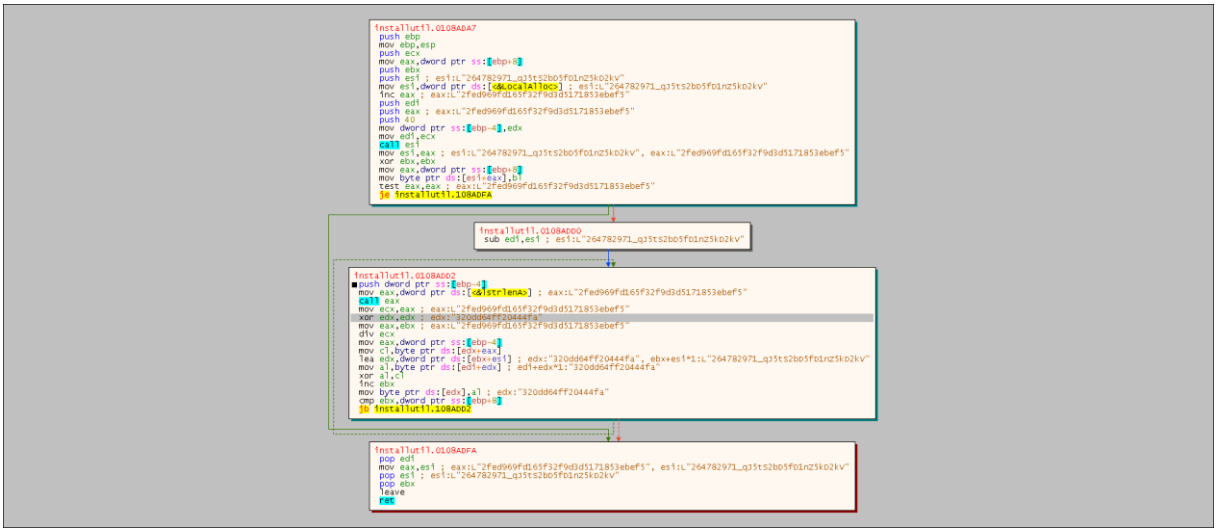

Figure 15 Algorithm for analysing Sitring expressions

Process Access Detection

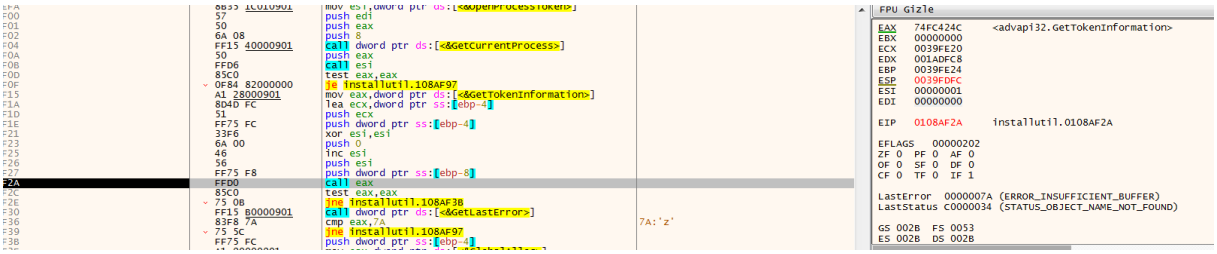

Figure 16 Access token information of the current process

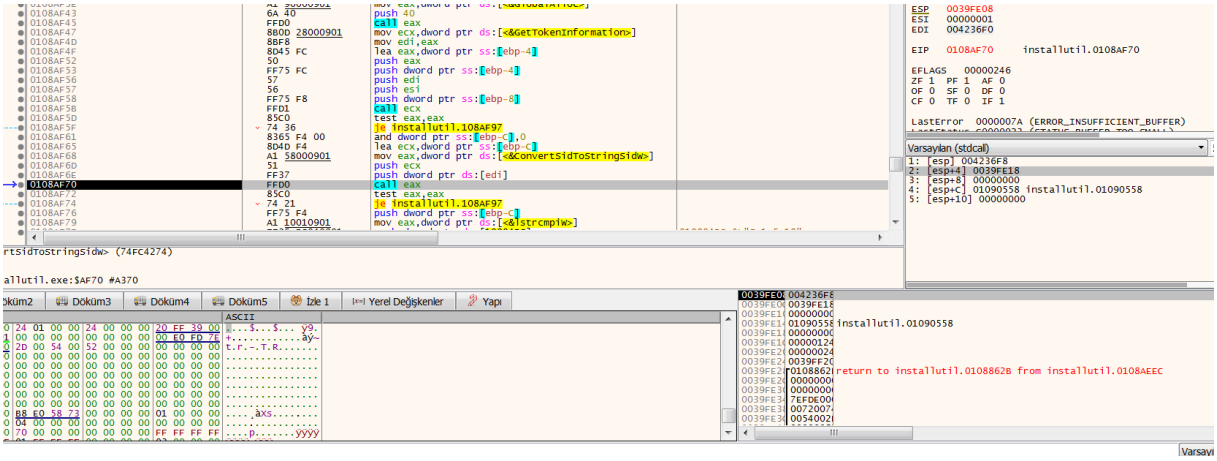

Figure 17 Retrieval of SID information for comparison

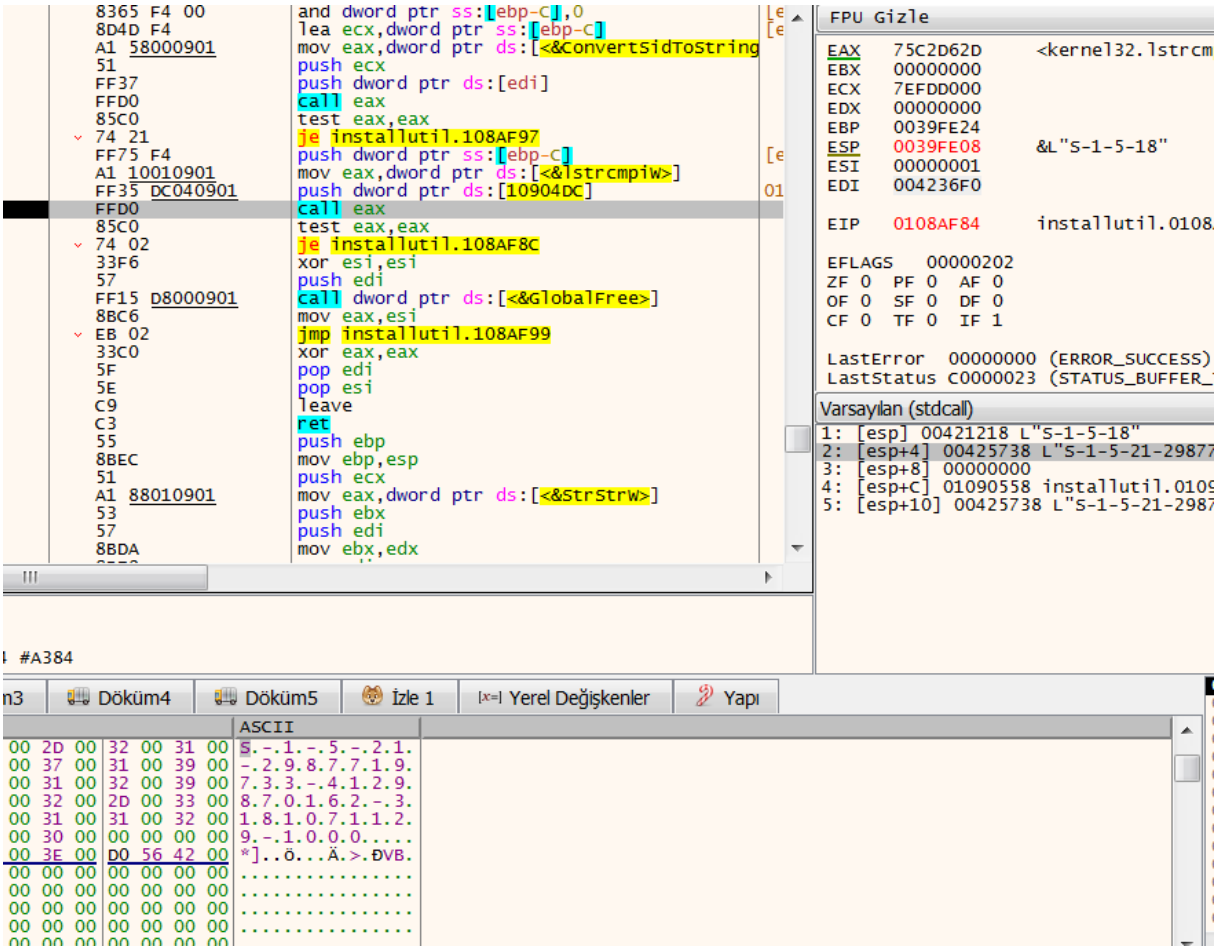

Figure 18 Admin authorisation comparison with SID information

It has been determined that the malware checks whether it has Admin authorisation. If it does not have Admin authorisation, it copies the Access Token of explorer.exe and restarts itself.

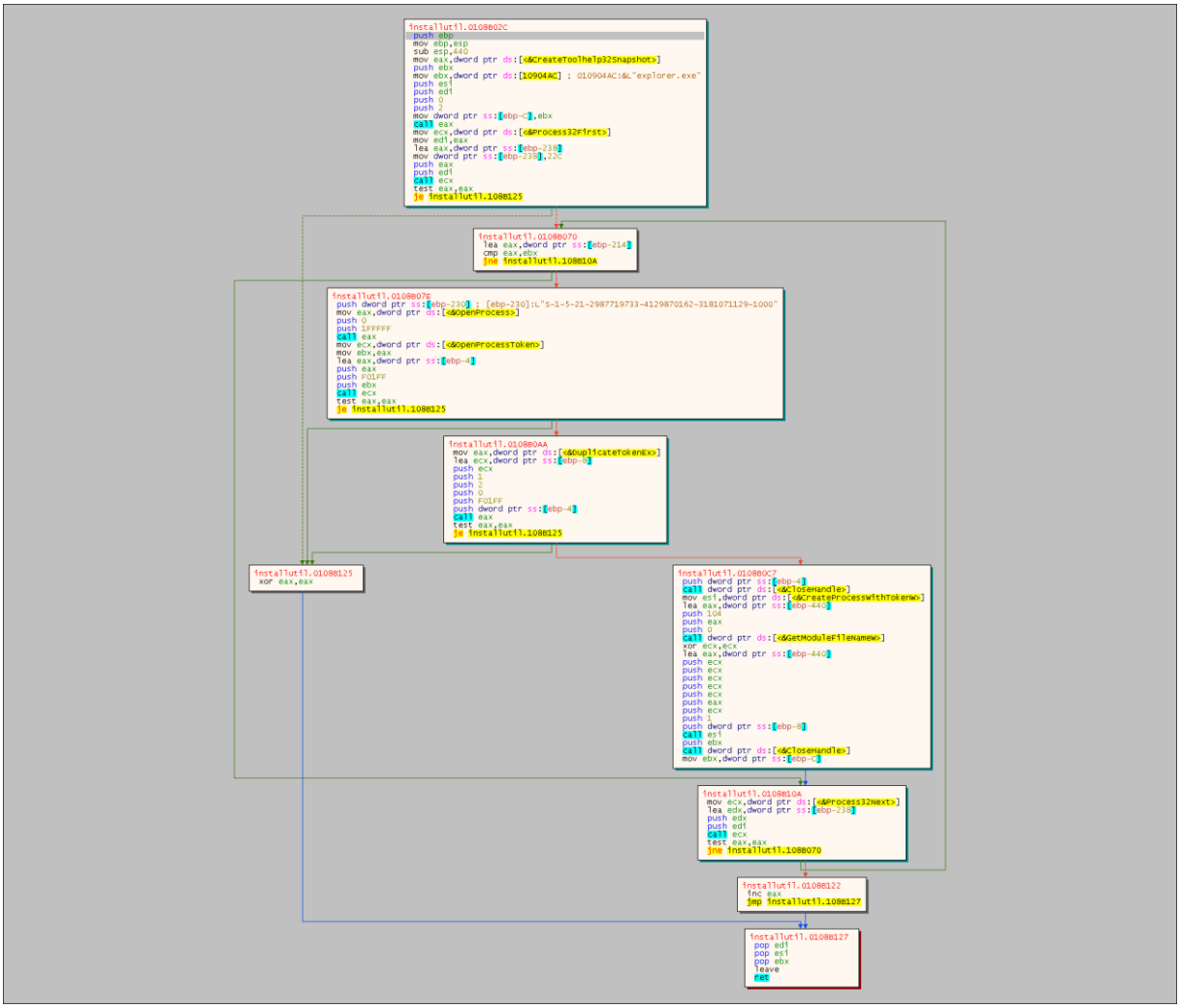

Figure 19 Algorithm to be applied if the process detects that it does not have admin authorisation

Creation of Request Contents

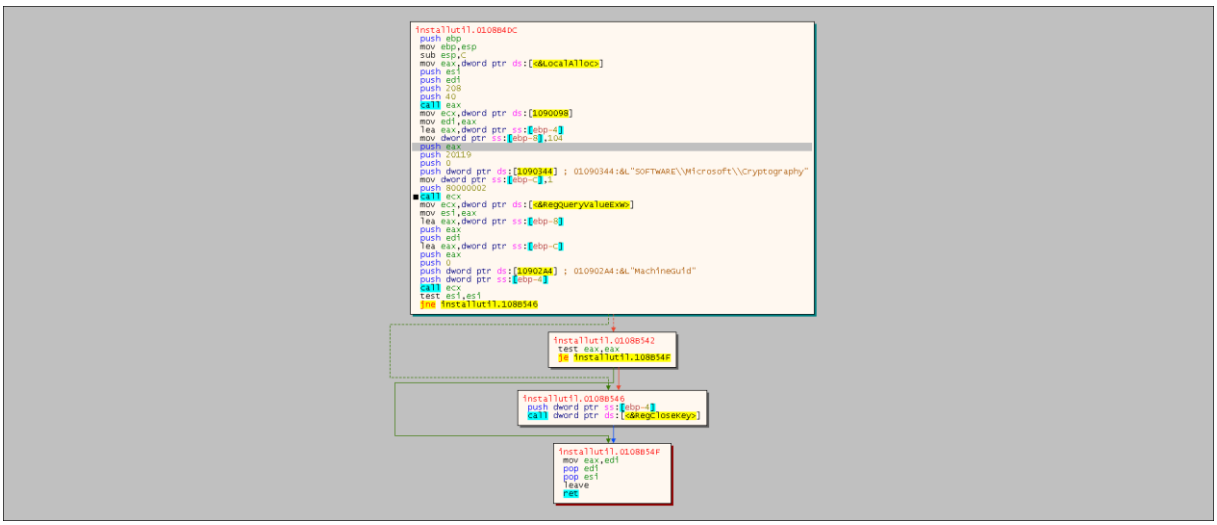

Figure 20 Obtaining Machine GuID information

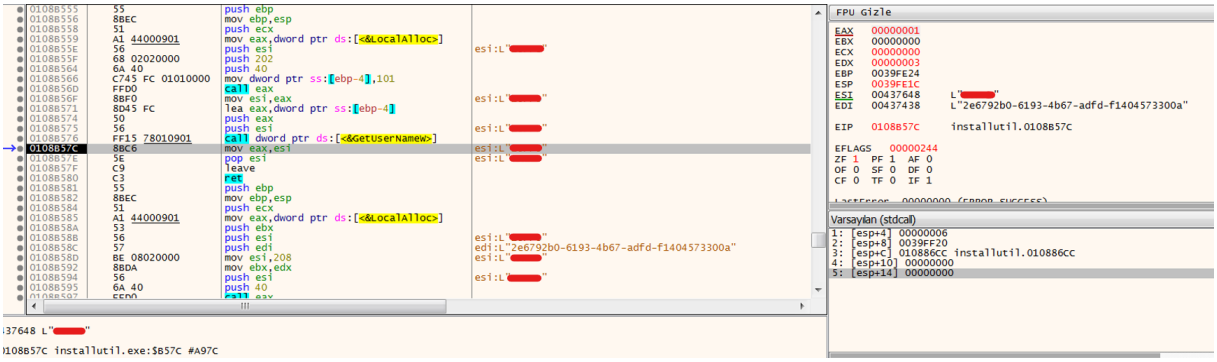

Figure 21 Pulling username information to generate victim ID information

It has been detected that it tries to create a uniq machine ID with MachineGuID and username. The malware will use this machineID and the configID variables it analyses in the http request it will send in the future. The created machineID is as follows:

machineId= <MachineGuID> |<username>&configId=2fed969fd165f32f9d3d5171853ebef5

,

Network Analysis

Request Analysis

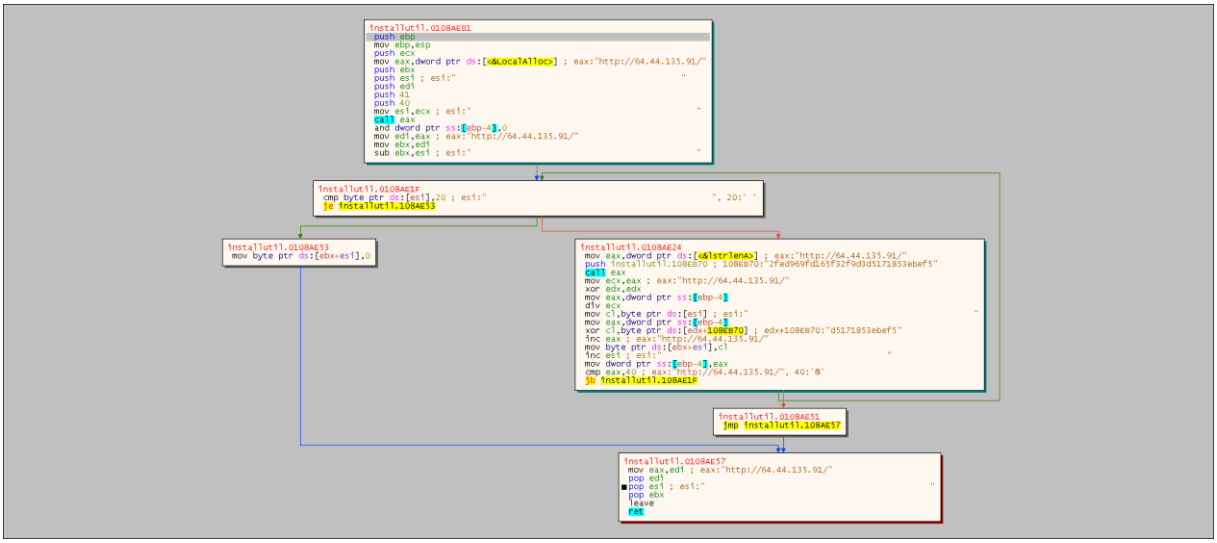

Figure 22 IP resolving

As a result of the analysis, it was determined that the IP **"http://64[.]44[.]135[.]91/"** appeared.

Request İçeriği

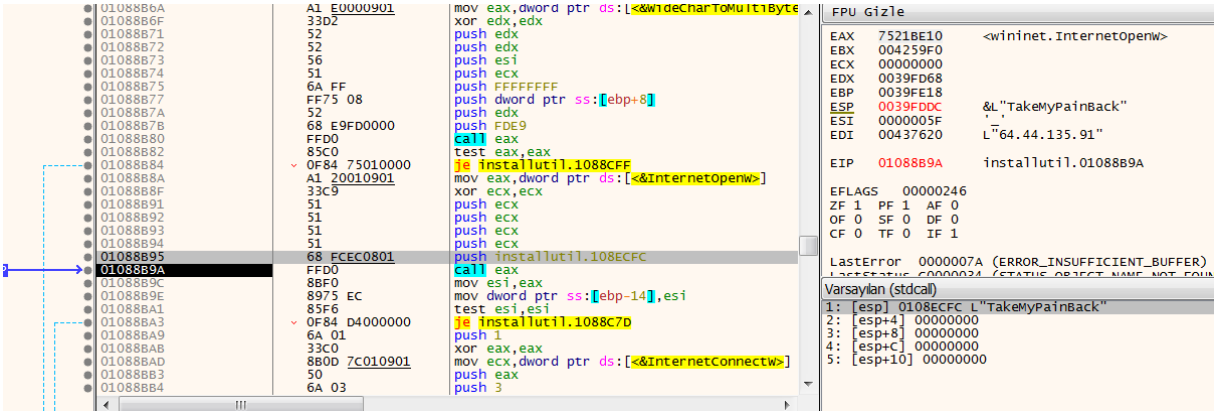

Figure 23 Detection of Agent information used in C2 communication

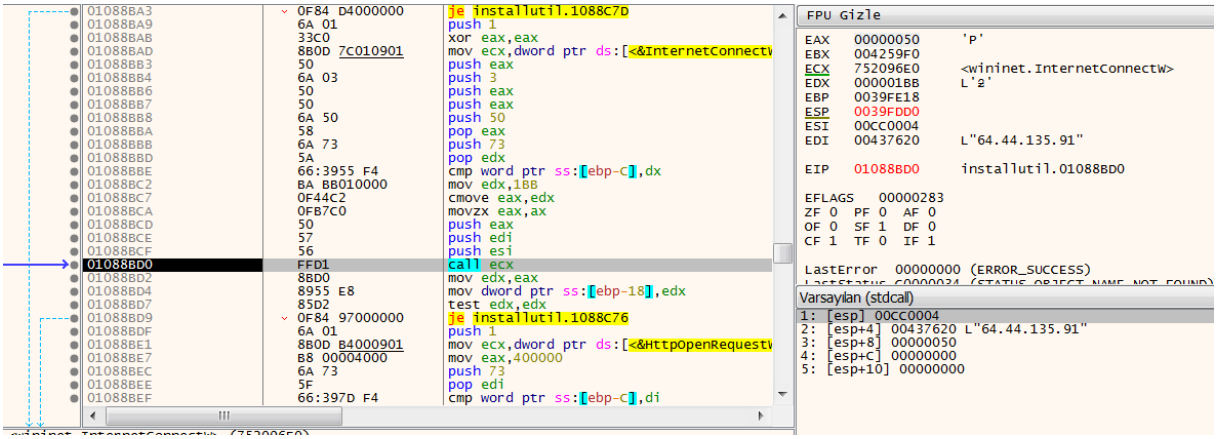

Figure 24

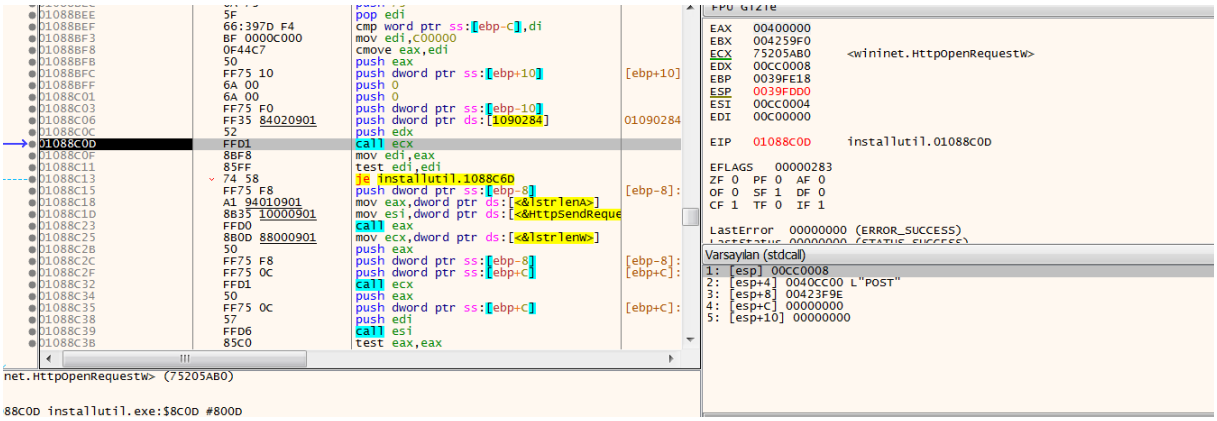

Figure 25 Determining the method of the request to be sent

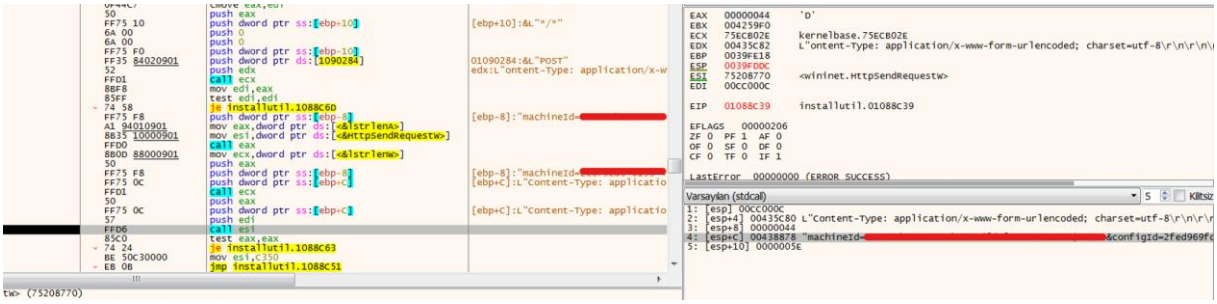

Figure 26

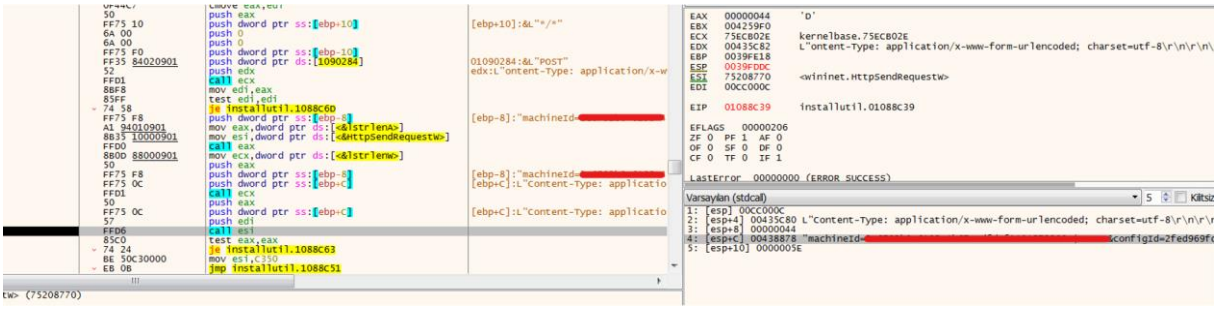

Figure 27 Sending the edited request to the C2 server

It has been determined that the agent information is **"TakeMyPainBack"** and the Request method is POST.

Http request content:

- Content-Type: application/x-www-form-urlencoded; charset=utf-8\r\n\r\n\r\n
- machineId=2e6792b0-6193-4b67-adfd-f1404573300a|zorro&configId=2fed969fd165f32f9d3d5171853ebef5

Expected Variables in Respons

token
wlts_
grbr_
tlgrm_
dr_

Table 7

When the behaviour of malware after the respons was examined, it was determined that it tried to use some data from the respons. The detected data variables are shown in Table 7.

After Respons

Gathering Device Information

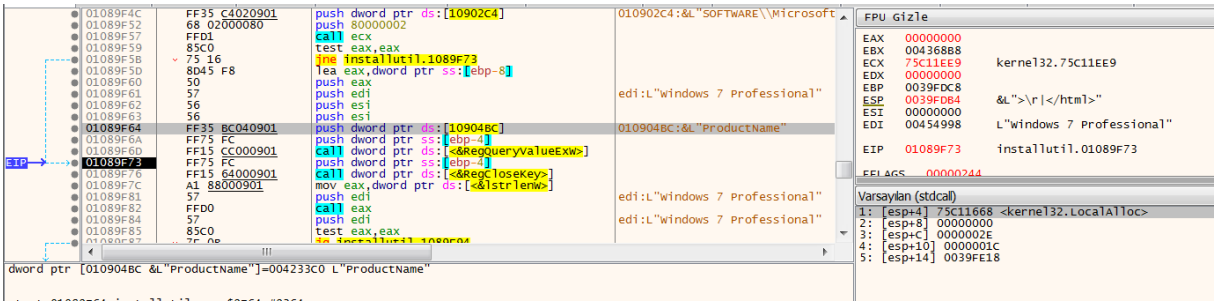

Figure 28 Detection of collecting information through the Registery

It has been found that the malware collects some information from the victim device. This information includes:

Locale	Time Zone	OS	Architecture	System Informations
Memory Information	Display Size	Display Devices	Application Information	

Tablo 1 3 Registery üzerinden aldığı bazı bilgiler

It has been found that malware collects this information in the following format:
- Locale: %s \n\t- Time zone: %c%ld minutes from GMT \n\t- OS: %s\n\t- CPU: %s (%d cores)) - \n\t-Architecture: x%d\n\t- RAM: %d MB \n\t- Display size: %dx%d\n\t- Display Devices: %s

It generates a random file name for the generated collection of information and is transmitted to the server with the POST method along with the file name.
Request içeriği: "Content-Type: multipart/form-data; boundary=bF0xB2TEnUcQ7DfR\r\n\r\n\r\n”

bF0xB2TEnUcQ7DfR = <verilerin toplandığı dosyanın adı>

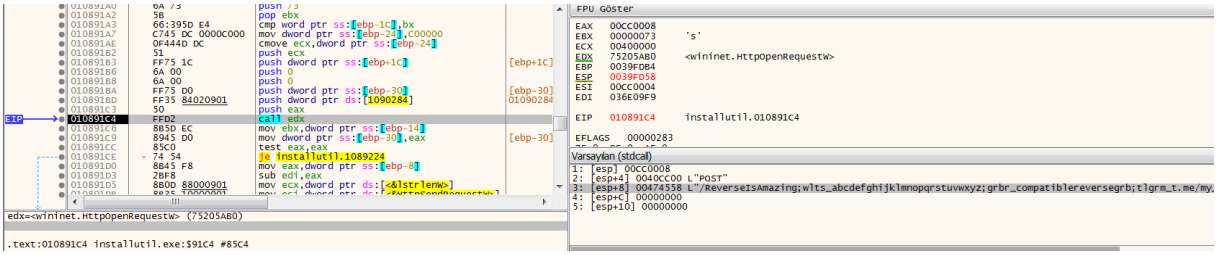

Figure 29 Sending the collected information back to the C2 server with the token information returned by the server

It was found that the token information received from the server was also used in sending requests to the server.

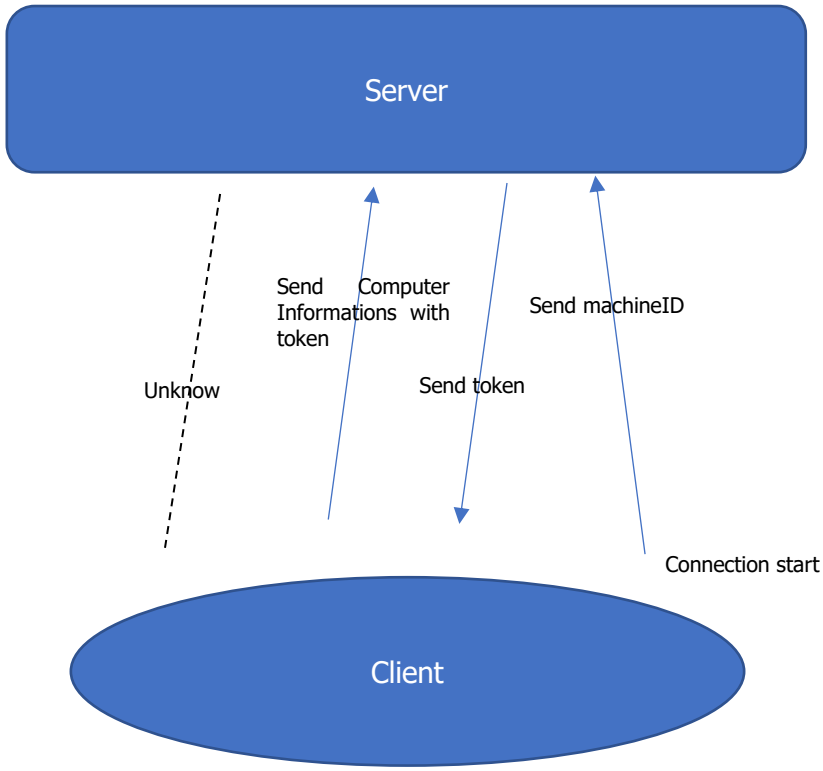

Figure 30 C2 Server and victim device communication

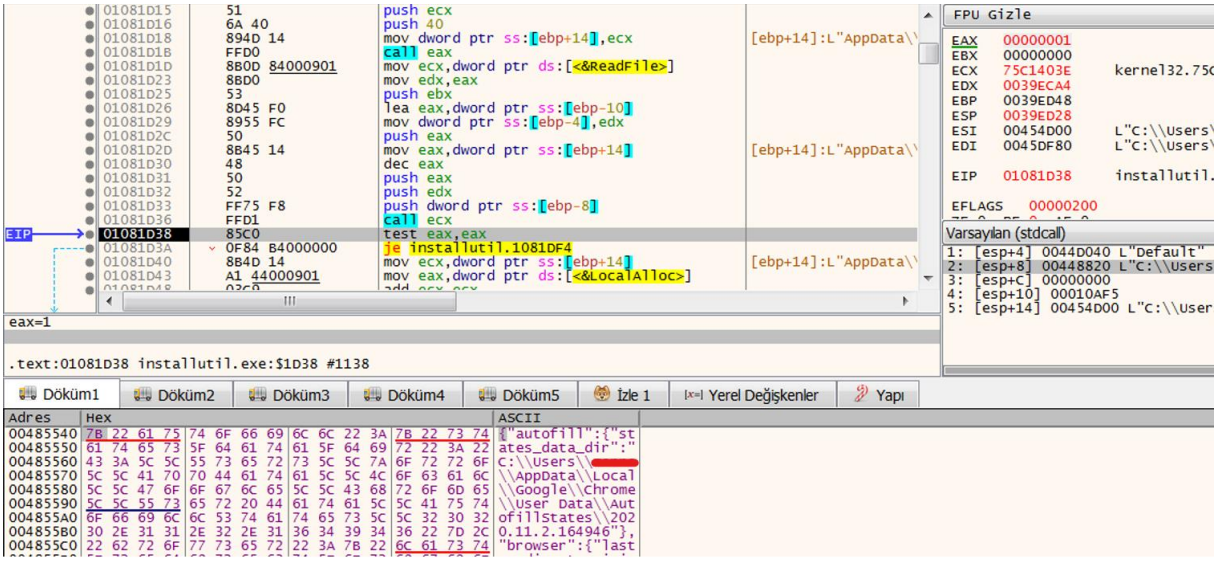

Figure 31

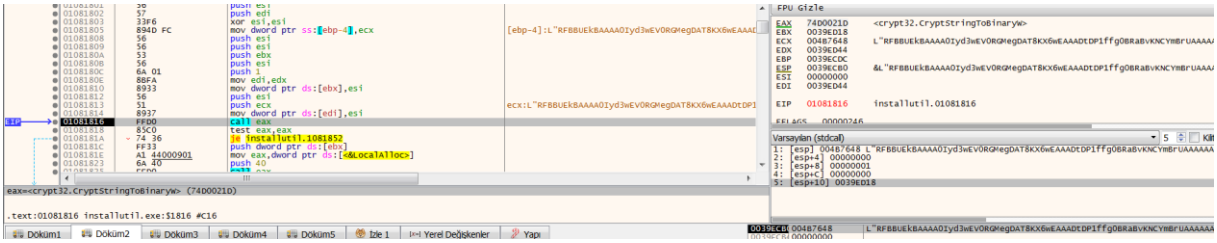

Figure 32 Detection of encrypted_key information encrypted with base64 algorithm and used as byte

It was found that the malware extracted the information in the **"AppData\\\Local\\\Chrome\\\User Data\\\Default\\\Login"** file and the **"AppData\\\Local\\\Google\\\Chrome\\\User Data\\\AutofillStates\\\"** folder.
It was observed that encrypted_key data was encrypted with base64 encryption algorithm.

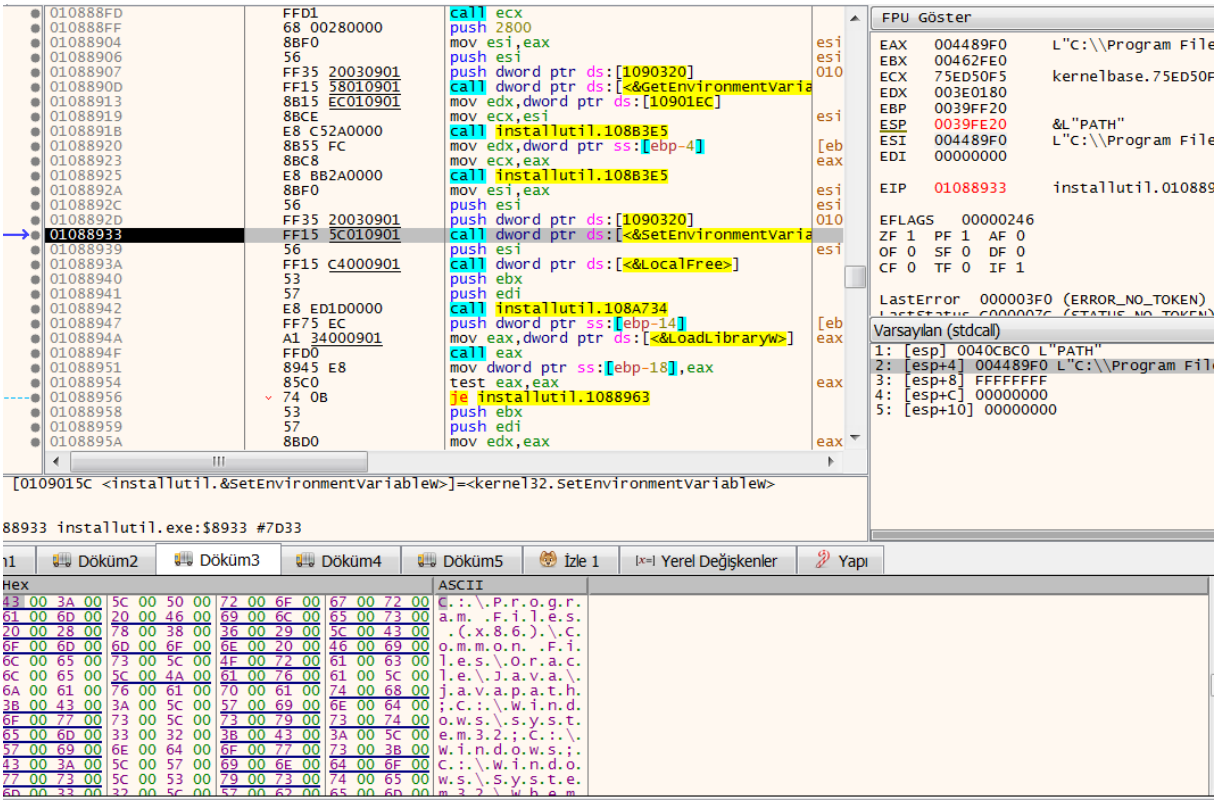

Figure 33 Change detection on environment variables

It was determined that malware made additions on the PATH variable among the environment variables.

C:\\Program Files (x86)\\Common Files\\Oracle\\Java\\javapath;C:\\Windows\\system32;C:\\Windows;C:\\Windows\\System32\\Wbem;C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\;C:\\Users\\<user>\\AppData\\Local\\Programs\\Python\\Python37\\Scripts\\;C:\\Users\\<user>\\AppData\\Local\\Programs\\Python\\Python37\\;C:\\Users\\<user>\\AppData\\LocalLow

DLL Loading

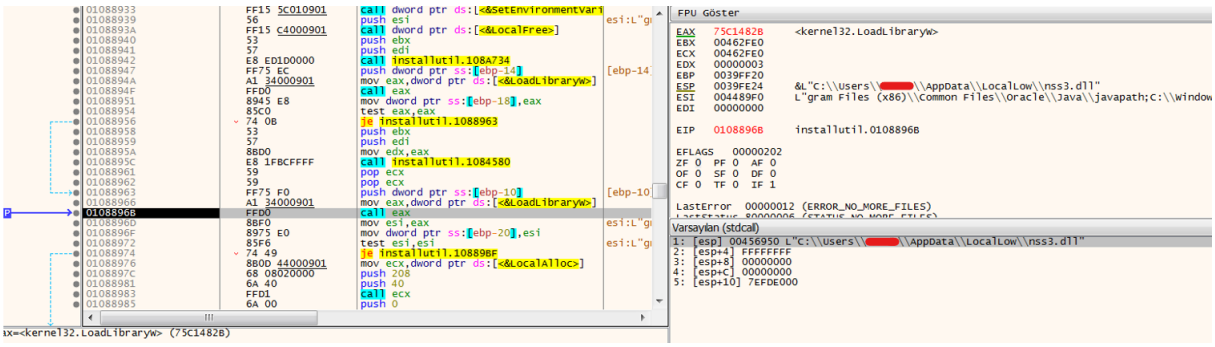

Figure 34 nss3.dll installation process

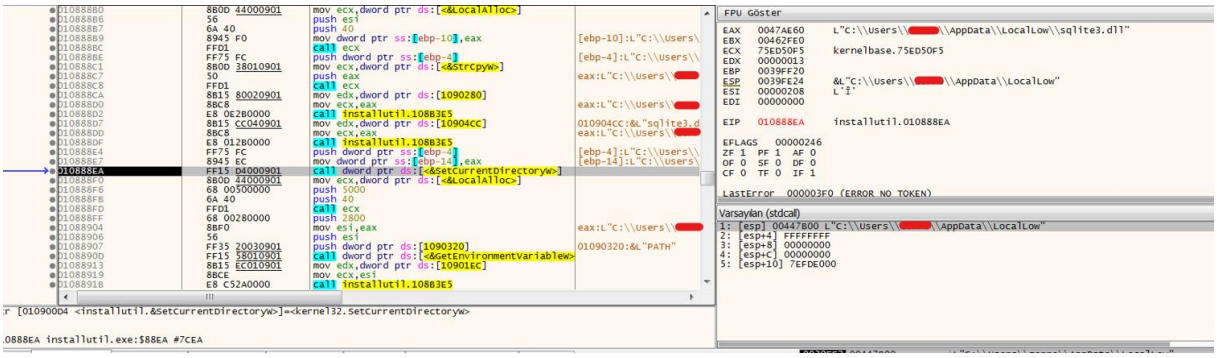

Figure 35 sqlite3.dll installation process

It has been detected that the malware installs sqlite3.dll and nss3.dll created in the LocalLow directory.

Database Operations

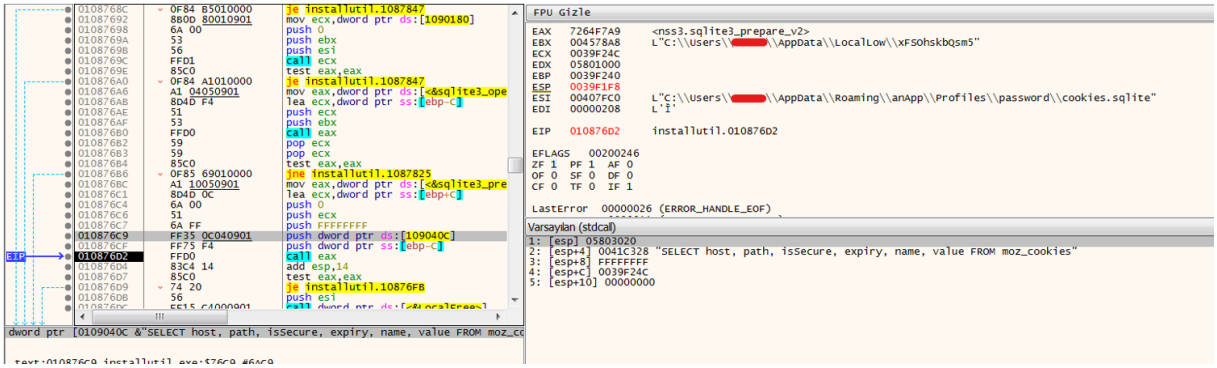

Figure 36

It has been detected that the malware creates a database file with a random name in the AppData\\LocalLow\\ directory. It has been determined that it pulls data from the following files with SQL queries to the created database.

coocies.sqlite
formhistory.sqlite
password.txt
storage\default
wallet.dat
logins.json

Table 8 Some targeted file and directory names

File Traversal Algorithm

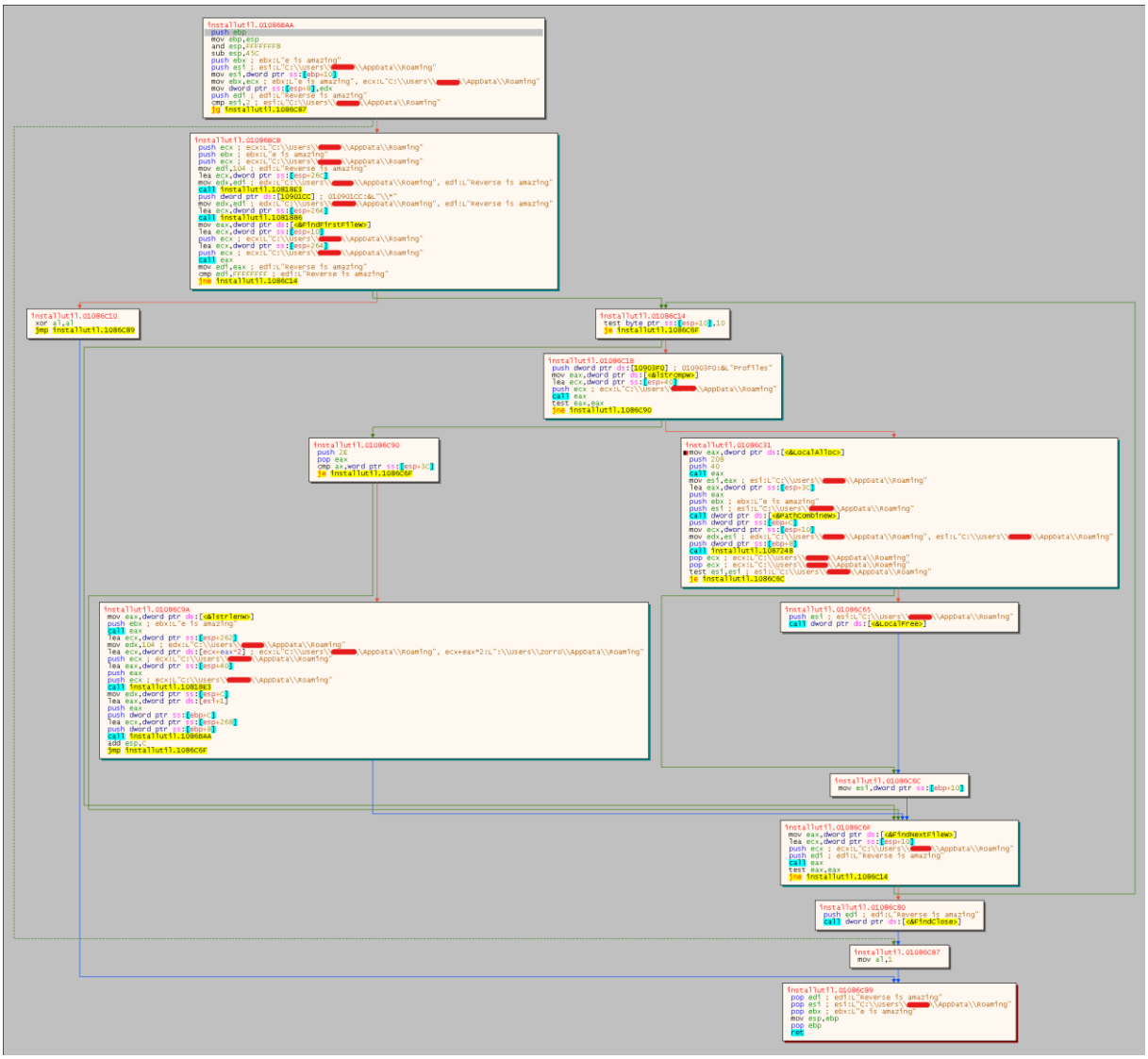

Figure 37 Index scanning algorithm

It was observed that it scanned the directory until it found the Profiles and User Data folder.

Additional Analysis

It was found that malware takes a ScreenShot and saves it as a recording.

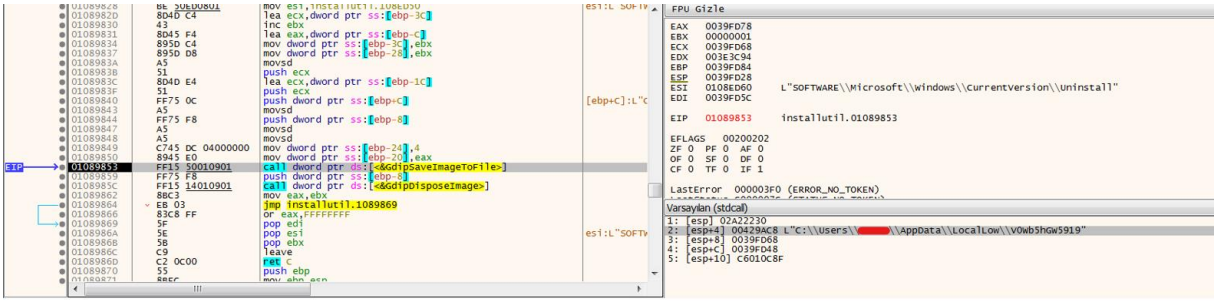

Figure 38 Saving the screen photo taken by Malware

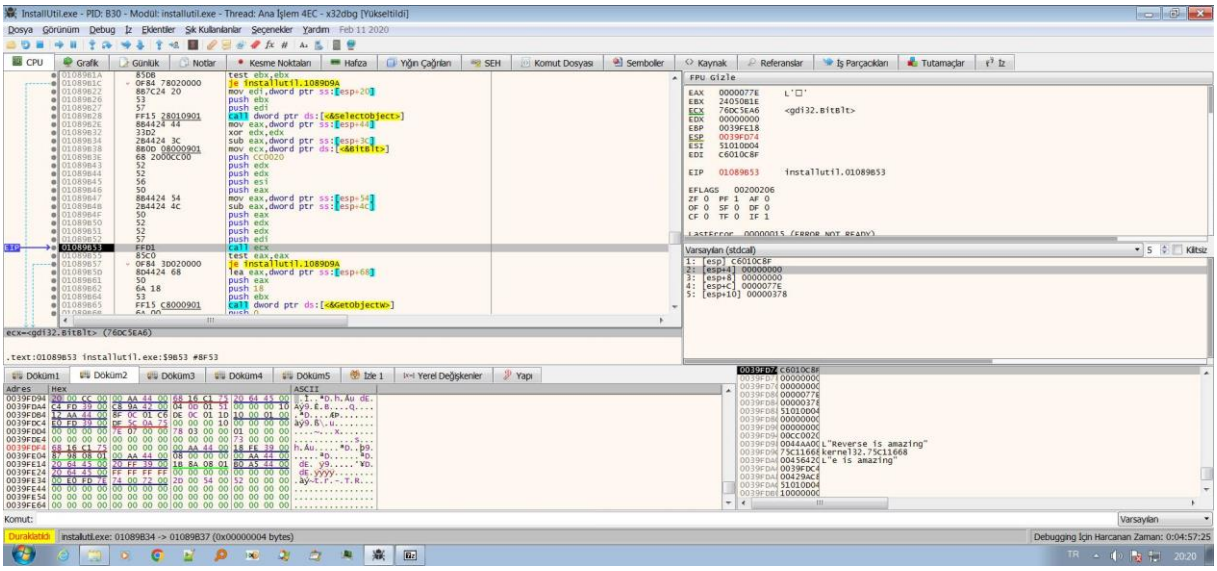

Figure 39 Screen Photo taken by Malware

SQL Queries

SELECT origin_url, username_value, password_value FROM logins
SELECT origin_url, username_value, password_value FROM logins
SELECT host_key, path, is_secure , expires_utc, name, encrypted_value FROM cookies
SELECT name, value FROM autofill
SELECT host, path, isSecure, expiry, name, value FROM moz_cookies
SELECT fieldname, value FROM moz_formhistory
SELECT name_on_card, card_number_encrypted, expiration_month, expiration_year FROM credit_cards

Table 9 Some of the enquiries made

YARA Rule

```
rule Rule_InstallUtil
{
meta:
    author = "Bilal BAKARTEPE (EchoCTI Team)"
    site = "https://github.com/bixploit"
    description = "RaccoonStealler v2.0 second stage PE file"
    hash= "d69ee30203430d1404a2890268bb04e9"
strings:
    $sql1 = "SELECT origin_url, username_value, password_value FROM logins"
    $sql2 = "SELECT host_key, path, is_secure , expires_utc, name, encrypted_value FROM cookies"
    $sql3 = "SELECT name, value FROM autofill"
    $sql4 = "SELECT host, path, isSecure, expiry, name, value FROM moz_cookies"
    $sql5 = "SELECT fieldname, value FROM moz_formhistory"
    $sql6 = "SELECT name_on_card, card_number_encrypted, expiration_month, expiration_year FROM
credit_cards"

    $dir_name1 = "profiles"
    $dir_name2 = "712006f6e7da2882" //User Data
    $dir_name3 = "Default"
    $dir_name4 = "Login Data"
    $file_name1= "password.txt"
    $file_name2= "cookies.sqlite"
    $file_name3= "Cookies"

    $agent="TakeMyPainBack"

    $ip_clear="http://64.44.135.91"
    $ip_enc="d5171853ebef5"

    $enc_str1="aa0bb6f89e4fc28e"
    $enc_str2="ba0c5f9d6a984fdd" //encrypted_values
    $enc_str3="587a51bde849292f"
    $enc_str4="ca82e1c9d5793376"
    $configrationID="2fed969fd165f32f9d3d5171853ebef5"
    $mutex_name="264782971_qJ5tS2bD5fD1nZ5kD2kV"

    $respons_variable1="320dd64ff20444fa" //tlgrm
    $respons_variable2="d286b66a2753e1b1" //wlts
    $respons_variable3="bee04f3449ba713e" //sstmnfo
    $respons_variable4="6ef9561122a8649a" //token
    $respons_variable5="877e12dc4d066d8c" //nss3.dll
    $respons_variable6="274f2fd9bfa77a7c" //sqlite.dll

    $opc1 = {53 56 57 6A 41 6A 40 8B F1 FF D0 83 65 FC 00 8B F8 8B DF 2B DE 80 3E 20 74 2F A1
        94 01 41 00 68 70 EB 40 00 FF D0 8B C8 33 D2 8B 45 FC F7 F1 8A 0E 8B 45 FC 32 8A 70 EB 40
00 40 88 0C 33 46 89 45 FC 83 F8 40 72 CE}// allocation and deobfuscation
    $opc2 = {55 8B EC 51 53 56 57 8B 3D 88 00 41 00 8B DA 53 89 4D FC FF D7 FF 75 FC 8B F0 FF D7
8B 0D 44 00 41 00 8D B8 80 00 00 00 03 FE 8D
        04 3F 50 6A 40 FF D1 FF 75 FC 8B F0 8B D7 8B CE E8 34 64 FF FF 53 8B D7 8B CE E8 57 64 FF
FF FF 75 FC FF 15 D8 00 41 00 5F 8B C6 5E 5B C9 C3}//deobfuscation and ascii to unicode transition

condition:
    (any of ($opc*)) and (2 of ($sql*,$dir_name*, $file_name*,$enc_str*,$respons_variable*) or any
of ($ip_clear,$ip_enc,$agent, $configrationID, $mutex_name))
}
```

MITRE ATTACK TABLE

Reconnaissance	Execution	Discovery	Collection	Defense Evasion	Credential Access	Command and Control	Exfliration
T1592 Gather Victim Host Information: Hardware	T1559 Inter-Process Communication: Component Object Model	T1012 Query Registry	T1005 Data from Local System	T1070 Indicator Removal on Host: File Deletion	T1539 Steal Web Session Cookie	T1071 Application Layer Protocol: Web Protocols	T1041 Exfliration Over C2 Channel
T1589 Gather Victim Identity Information: Credentials		T1082 System Information Discovery	T1113 Screen Capture	T1140 Deobfuscate/Decode Files or Information		T1105 Ingress Tool Transfer	T1020 Automated Exfiltration
T1592 Gather Victim Host Information: Software		T1614 System Location Discovery: System Language Discovery					

Mitigations

1. Emails and senders should be carefully checked to ensure they are genuine before opening any attachments.
2. Do not download any resources from unsafe websites.
3. Use a reliable, high quality and always updated antivirus software.
4. Keep your operating system and applications up to date with the latest security patches.
5. End user training is vital for your organisation, make sure you inform your employees about the dos and don'ts of online security.


ECHO
CYBER THREAT INTELLIGENCE