







[\\_Share](#)

# RedLine Stealer Campaign Using Binance Mystery Box Videos to Spread GitHub-Hosted Payload

## Summary

RedLine Stealer is a malware that [emerged](#) in 2020, [discovered](#) in underground forums being sold in different plans, starting from \$100 per month. The malware offers [many capabilities](#) for device reconnaissance, remote control, and information stealing, including:

- Data from browsers (e.g. login, passwords, credit cards, cookies, etc.);
- Data from Discord and Telegram (e.g. chat logs, tokens, etc.);
- VPN and FTP Credentials;


Since its discovery, attackers have used many different vectors to spread this stealer, including through [fake installers](#) and fake [game hacking](#) tools. Also, RedLine Stealer [was found](#) in compromised devices by the [DEV-0537](#) hacking group (a.k.a. lapsus\$).

In April 2022, Netskope Threat Labs identified a new RedLine Stealer campaign spread on YouTube, using a fake bot to buy Mystery Box NFT from Binance. The video description leads the victim to download the fake bot, which is hosted on GitHub.

In this blog post, we will analyze this campaign, showing how it's being spread and how the fake bot leads to RedLine Stealer.

# YouTube Videos

The malware is spread through YouTube videos that lure victims into downloading a fake bot to automatically buy Binance NFT Mystery Boxes. At this point, we found five videos across multiple channels that are part of the same campaign. All the URLs can be found in our [GitHub repository](#).



**Binance nft  
Mystery box**


1:22

**Binance NFT Free Auto Bot | Bot for instant purchase Mystery Box NFT**

79 views • 1 month ago

Andrés Jiménez

—Tags— binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, ...



**Binance nft  
Mystery box**


1:22

**Binance NFT Free Auto-Bot Bot for instant purchase Mystery Box NFT**

17 views • 1 month ago

손흥민 247

—Tags— binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, ...



**Binance nft  
Mystery box**


1:22

**Binance NFT Free Auto Bot Bot for instant purchase Mystery Box NFT**

64 views • 2 weeks ago

Sam Bankman-Fried

—Tags— binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, ...



**Binance nft  
Mystery box**


1:22

**Binance NFT Free Auto Bot | Bot for instant purchase Mystery Box NFT**

79 views • 1 month ago

Andrés Jiménez

—Tags— binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, ...



**Binance nft  
Mystery box**


1:22

**Binance NFT Free Auto-Bot Bot for instant purchase Mystery Box NFT**

17 views • 1 month ago

손흥민 247

—Tags— binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, ...



**Binance nft  
Mystery box**

1:22

**Binance NFT Free Auto Bot Bot for instant purchase Mystery Box NFT**

64 views • 2 weeks ago

Sam Bankman-Fried

—Tags— binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, ...

Attacker spreading RedLine through YouTube video.

The video description provides details and the download link for the fake bot, which is supposed to be presented as a Chrome extension.

## Binance NFT Free Auto Bot | Bot for instant purchase Mystery Box NFT

79 views • Mar 18, 2022



Andrés Jiménez

396 subscribers

Want to download NFT Mystery Box faster than anyone ?! In the video, a bot for auto-purchase Mystery Box NFT, presented as an extension for the Chrome browser.

Link: <https://github.com/NFTSupp/NFTBOT/raw...>

--Tags--

binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, binance trading bot, binance trade bot, bitcoin trading bot, btc, bot, binance bot trading, nance grid trading, bnb, binance nft, bot for binance, binance trading tutorial

SHOW LESS

Video description with the link to download the fake bot.

The video description also contains different tags, probably to increase its visibility, including:

binance, nft, bot, buy, mysterybox, box, busd, autobuy, cryptobot, binance, bitcoin, crypto trading bot, cryptocurrency, binance trading bot, binance trade bot, bitcoin trading bot, btc, bot, binance bot trading, nance grid trading, bnb, binance nft, bot for binance, binance trading tutorial

## Stage 01 — Loader

All the videos we found are pointing to the same GitHub URL, downloading a file named “BinanceNFT.bot v.1.3.zip”. Once we decompress the ZIP file, we have the packed RedLine sample (“BinanceNFT.bot v.1.3.exe”) and a Microsoft Visual C++ Redistributable installer (“VC\_redist.x86.exe”).

Name	Size
locales	
BinanceNFT.bot v.1.3.exe	948 KB
README.txt	1 KB
VC_redist.x86.exe	13,405 KB

Decompressed ZIP file downloaded from GitHub.

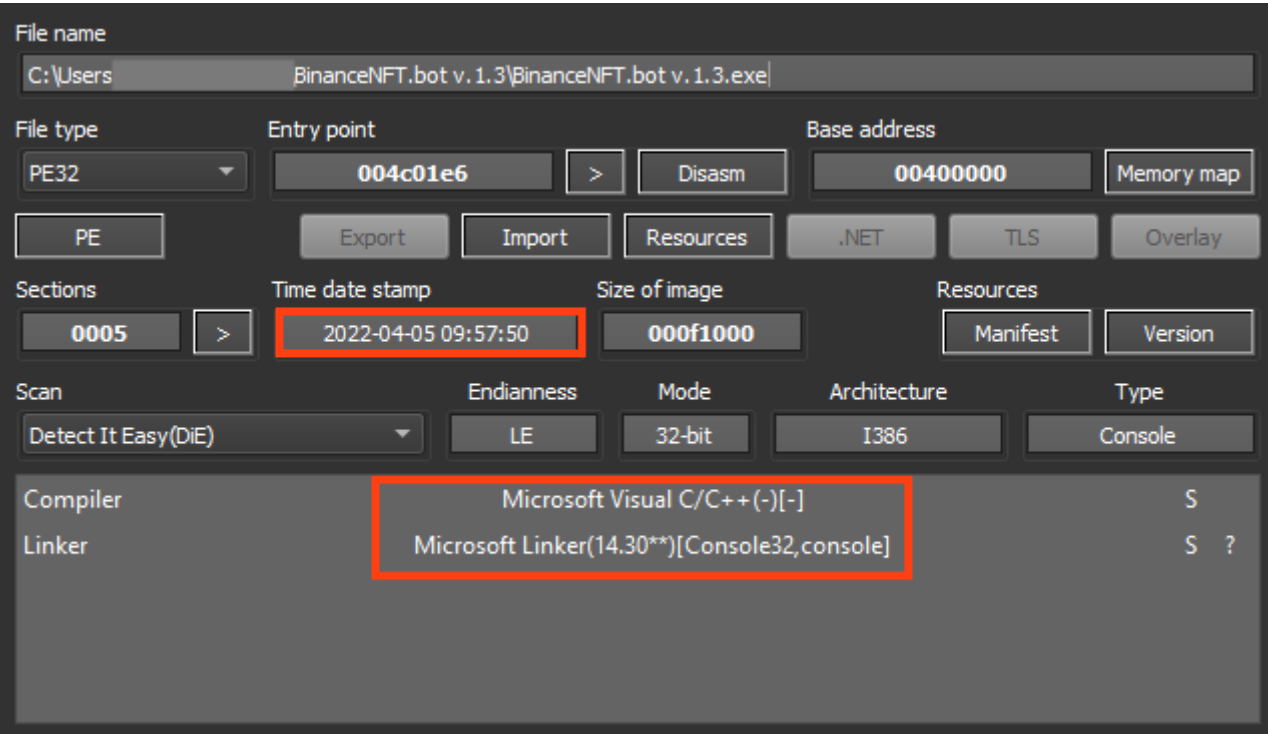
The “README.txt” file contains the instructions that should be followed to run the fake NFT bot, including installing the Microsoft Visual C++. This is probably needed as RedLine is developed in .NET and it is also unpacked and injected into an executable from this framework.

```
1 1.Unzip the archive
2 2.Start the bot
3 3.Install the VC_redist.x86 library
4 4.Start using
5 Good luck!
6
```

Readme file.

The first stage was likely compiled on April 5, 2022, and it’s responsible for decrypting and loading RedLine Stealer into another process.





Details of the packed RedLine Stealer sample.

The binary details also include values that seem to be copied from another executable, using “LauncherPatcher.exe” as the original filename.

```
4 PRODUCTVERSION 1,0,0,14
5 FILEOS 0x40004
6 FILETYPE 0x1
7 {
8 BLOCK "StringFileInfo"
9 {
10     BLOCK "040904b0"
11     {
12         VALUE "CompanyName", "Rockstar Games"
13         VALUE "FileDescription", "Rockstar Games Launcher Patcher"
14         VALUE "FileVersion", "1.0.0.14"
15         VALUE "InternalName", "LauncherPatcher.exe"
16         VALUE "LegalCopyright", "Rockstar Games Inc. (C) 2005-2021"
17         VALUE "OriginalFilename", "LauncherPatcher.exe"
18         VALUE "ProductName", "Rockstar Games Launcher Patcher"
19         VALUE "ProductVersion", "1.0.0.14"
20     }
21 }
22 }
```

Further details about the first stage.

Many malware families use a trick to [delay the execution](#) of its functions, often to delay the execution inside sandboxes, which usually contain limited time of operation. As a result, there are sandboxes that are able to bypass this technique, by patching or hooking [Sleep](#) functions, for example.

This RedLine Stealer loader contains a simple trick to evade sandboxes with such functionality. Upon execution, it tries to delay the execution by 15 seconds and compares the timestamp ([GetTickCount](#)) before and after the Sleep API execution. If the elapsed time is less than 15 seconds, it exits the process.

```
call ds:SendMessageA
call ds:GetForegroundWindow
call ds:GetConsoleWindow
mov edi, ds:GetTickCount
mov esi, eax
call edi ; GetTickCount
push 0 ; nCmdShow
push esi ; hWnd
mov ebx, eax
call ds:ShowWindow
push 15000 ; dwMilliseconds
call ds:Sleep
call edi ; GetTickCount
sub eax, ebx
cmp eax, 15000
jl exit
```

Trick to evade sandbox analysis.

This can be tested by patching the Sleep function in a debugger.

004BDAC0	6A 00	push 0
004BDAC8	68 68660000	push 6668
004BDACD	6A 00	push 0
004BDACF	FF15 50D14D00	call dword ptr ds:[&SendMessageA]
004BDAD5	FF15 44D14D00	call dword ptr ds:[&GetForegroundWindow]
004BDADB	FF15 28D04D00	call dword ptr ds:[&GetConsoleWindow]
004BDAD1	8B3D 20D04D00	mov edi,dword ptr ds:[&GetTickCount]
004BDAE7	8BF0	mov esi,eax
004BDAE9	FFD7	call edi
004BDAEB	6A 00	push 0
004BDAED	56	push esi
004BDAEE	8BD8	mov ebx,eax
004BDAF0	FF15 48D14D00	call dword ptr ds:[&ShowWindow]
004BDAF6	68 983A0000	push 3A98
004BDAFB	90	nop
004BDAFC	90	nop
004BDAFD	90	nop
004BDAFE	90	nop
004BDAFF	90	nop
004BDB00	90	nop
004BDB01	FFD7	call edi
004BDB03	2BC3	sub eax,ebx
004BDB05	3D 983A0000	cmp eax,3A98
004BDB0A	0F8C 57010000	j1 binancenft.bot v.1.3.4BDC67
004BDB10	B9 1D000000	mov ecx,1D
004BDB15	8D7C24 18	lea edi,dword ptr ss:[esp+18]
004BDB19	BE 90D34D00	mov esi,binancenft.bot v.1.3.4DD390
004BDB1E	F3:A5	rep movsd
004BDB20	68 00E1F505	push 5F5E100
004BDB25	E8 266C0000	call binancenft.bot v.1.3.4C4750
004BDB2A	68 00E1F505	push 5F5E100

EIP → 004BDB0A

Jump is taken  
binancenft.bot v.1.3.004BDC67  
.text:004BDB0A binancenft.bot v.1.3.exe:\$BDB0A #BCF0A

RedLine loader exiting the process if the Sleep function is bypassed.

If the sandbox is not detected through this simple trick, it then decrypts the next stage using a simple rolling XOR algorithm with “OdoAAtK” as the key.

```

push edi
mov edi,ecx
test ebp,ebp
je binancenft.bot v.1.3.4BC8F1
push ebx
mov ebx,dword ptr ss:[esp+14]
push binancenft.bot v.1.3.4DD37C
call binancenft.bot v.1.3.printf
mov eax,24924925
add esp,4
mul esi
mov eax,esi
sub eax,edx
shr eax,1
add eax,edx
shr eax,2
lea ecx,dword ptr ds:[eax*8]
sub ecx,eax
mov eax,esi
sub eax,ecx
mov al,byte ptr ds:[eax+4EC000]
xor byte ptr ds:[esi+ebx],al
inc esi
inc dword ptr ds:[edi]
cmp esi,ebp
je binancenft.bot v.1.3.4BC884
pop ebx
fld st(0),dword ptr ds:[4DD498]
pop edi
pop esi
pop ebp
ret 8

```

4DD37C: "yugdfsygdeyufgeyuf4" printf

eax+4EC000: "OdoAAtK"

Address	Hex	ASCII
00650FA0	02 3E FF 41 42 74 4B 4F 60 6F 41 41 8B B4 4F 64	>yABTKO oAA. Od
00650FB0	D7 41 41 74 4B 4F 64 6F 41 41 74 4B 4F 64 6F 41	xAA TKodo. AtKodoA
00650FC0	41 74 4B 4F 64 6F 41 41 74 4B 4F 64 6F 41 41 74	ATKodoAATKodoAAT
00650FD0	4B 4F 64 6F 41 41 74 4B 4F 64 6F 41 41 74 4B 4F	KodoAATKodoAATKo
00650FE0	6A 70 FB 4F 74 FF 46 A9 4E F9 40 38 86 6E 30 07	jp0TyfBn088.n0.
00650FF0	28 32 54 38 3D 08 08 33 20 19 68 2C 05 01 2F 3E	(2T;...3.k.../.
00651000	00 6B 2D 01 4F 33 34 1A 6B 26 0A 4F 05 0E 27 68	"...3.k.../.
00651010	22 0B 08 24 6F 79 46 45 40 6F 41 41 74 4B 4F 64	"...3.k.../.
00651020	3F 04 41 74 07 4E 67 6F 4A A2 BD B2 4F 64 6F 41	?..AT.Ngoj%*odoA
00651030	41 74 4B 4F 64 6F 43 40 7F 4A 7F 64 6F DF 40 74	ATKO..oc0..J.do0t
00651040	4B 4F 64 6F 41 41 74 4B 4F 64 6F 41 41 54 4B 4F	KGdoAATKQoAATKO
00651050	64 AF 40 41 74 4B 4F 64 6F 61 41 74 4B 4F 64 6F	d"0ATK..doAATKMo
00651060	45 41 74 4B 4F 64 6F 41 45 74 4B 4F 64 6F 41 41	EATKodoAETKodoAA
00651070	74 4B 4D 64 6F 43 41 74 4B 4F 64 6F 43 41 34 CE	tkMdoCATKodoCAAI
00651080	4F 64 7F 41 41 64 4B 4F 64 6F 51 41 74 5B 4F 64	Od..AAdKodoQAT[od
00651090	6F 41 74 5B 4F 64 6F 41 41 74 4B 4F 64 6F 41 74	oAAT[odoAATKodoA
006510A0	89 CF 4A 4F 37 6F 41 41 74 8B 4E 64 B9 45 41 74	"...JJO7oAAT.Nd'EAT
006510B0	4B 4F 64 6F 41 41 74 4B 4F 64 6F 41 41 74 4B 4F	KodoAATKodoAATKO
006510C0	64 8F 40 41 78 4B 4F 64 6F 41 41 74 4B 4F 64 6F	d..0AATKodoAATKodo
006510D0	41 41 74 4B 4F 64 6F 41 41 74 4B 4F 64 6F 41 41	AATKodoAATKodoAA
006510E0	74 4B 4F 64 6F 41 41 74 4B 4F 64 6F 41 41 74 4B	TKodoAATKodoAATK

Address	Hex	ASCII
00650FA0	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ.....yy..
00650FB0	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	.....@.....
00650FC0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
00650FD0	00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00	.....
00650FE0	0E 1F BA 0E 00 84 09 CD 21 B8 01 4C CD 21 54 68	..0...!..Li!Th
00650FF0	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00651000	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00651010	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode...\$. ....
00651020	50 45 00 00 4C 01 03 00 0B E3 C9 F9 00 00 00 00	PE..L....0000
00651030	00 00 00 00 E0 00 02 01 0B 01 30 00 00 9E 01 00	.....0.....
00651040	00 08 00 00 00 00 00 00 1E BC 01 00 00 20 00 00	.....%.....
00651050	00 C0 01 00 00 00 40 00 00 20 00 00 00 02 00 00	.....@.....
00651060	04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00	.....
00651070	00 00 02 00 00 02 00 00 00 00 00 00 02 00 85	.....@.....
00651080	00 00 10 00 00 10 00 00 00 10 00 00 00 10 00	.....
00651090	00 00 00 00 10 00 00 00 00 00 00 00 00 00 00	.....
006510A0	C8 BB 01 00 53 00 00 00 00 C0 01 00 D6 04 00 00	E...S...A..0...
006510B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
006510C0	00 E0 01 00 0C 00 00 00 00 00 00 00 00 00 00	.....
006510D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
006510E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....

Loader decrypting RedLine Stealer payload.

Then, it executes a shellcode, which is decrypted using the same algorithm.

Address	Hex	ASCII
0064FB68	1A EF 83 CA 0C 7C C0 8E E4 56 41 35 72 0B CF 5C	.1.E.  A.ava5r.I
0064FB78	6F 34 8B 5F 8A 12 A6 6B 41 14 FF A7 19 33 E4 3C	04>... ka.y5.3a<
0064FB88	49 47 BD 18 8C 8B BE BE 8B C0 87 E1 A6 35 61 7B	IG%. .%A.a!5a[
0064FB98	F5 48 A5 89 45 42 84 C0 89 41 6F 41 41 84 3F 44	0Hy.EB.A.AoAA.7D
0064FBA8	A5 87 59 72 84 CA A9 9B 90 BE 4E 33 02 3A 84 30	%..Yr.E0...N3...0
0064FBB8	CA 87 2A 16 8D 60 6F 14 CA 98 1A 1E 37 39 16 CA	E...o.E...79.E
0064FBC8	09 43 7C 92 E4 06 7D FF 0F 77 1C 6C 86 CA 24 6B	.C .a.jy.w.l.E\$K
0064FBD8	C4 3C 73 42 96 FF 03 6B 67 B0 CA 01 6C 48 80 ED	A<sb.y.kg'E.Th.i
0064FBE8	3A BD C8 39 B3 C6 21 67 C4 81 00 52 C4 60 DD 42	:%E9'Atga..RA'YB
0064FBF8	86 24 A3 CD 9B 90 BE 7A 31 47 3B 70 E4 14 BD 32	.\$fi...%Z1G;pA.%2

Address	Hex	ASCII
0064FB68	55 8B EC 8B 4D 08 8B C1 80 39 00 74 06 40 80 38	U..1.M..A.9.t.@.8
0064FB78	00 75 FA 2B C1 5D C2 04 00 55 8B EC 56 57 8B 7D	.u0+A]A...U.vw.}
0064FB88	08 33 F6 57 E8 D7 FF FF FF 8B C8 85 C9 74 20 0F	.30wxyyy..E.Et.
0064FB98	BE 07 C1 E6 04 03 F0 8B C6 25 00 00 00 74 08 3A	%..Ae..0..E...0t.
0064FBA8	C1 E8 18 33 F0 81 E6 FF FF FF 0F 47 49 75 0E 5F	Ae..30.ayyy.Giua..
0064FBB8	8B C6 5E 5D C2 04 00 55 8B EC 51 51 53 56 57 8B	.A]A..U.iQQSwv.
0064FBC8	7D 08 33 F6 88 47 3C 88 44 38 78 03 C7 8B 50 20	.30.G.C.D8x.C.P
0064FBD8	8B 58 1C 03 D7 8B 48 24 03 DF 8B 40 18 03 CF 89	.X..x.H5.B.@..I.
0064FBE8	55 FC 89 4D F8 89 45 08 85 C0 74 19 8B 04 B2 03	U0..Mo.E..At....
0064FBF8	C7 50 E8 82 FF FF FF 3B 45 0C 74 14 8B 55 FC 46	Cpè.yyy;E.t..U0F

0064FB68	55	push ebp
0064FB69	8BEC	mov ebp,esp
0064FB6B	8B4D 08	mov ecx,dword ptr ss:[ebp+8]
0064FB6E	8BC1	mov eax,ecx
0064FB70	8039 00	cmp byte ptr ds:[ecx],0
0064FB73	74 06	je 64FB7B
0064FB75	40	inc eax
0064FB76	8038 00	cmp byte ptr ds:[eax],0
0064FB79	75 FA	jne 64FB75
0064FB7B	2BC1	sub eax,ecx
0064FB7D	5D	pop ebp
0064FB7E	C2 0400	ret 4
0064FB81	55	push ebp

Loader decrypting and executing a shellcode.

And finally, the payload is injected to “RegSvcs.exe” using a simple process injection technique, similar to [RunPE](#). We also found cases where a similar loader injects RedLine Stealer into “AppLaunch.exe”, as we will describe later.

```
push eax
push edx
push edx
push 4
push edx
push edx
push edx
push dword ptr ss:[ebp+C]
push dword ptr ss:[ebp+8]
call dword ptr ss:[ebp-7C]
test eax, eax
je 7415B2
lea eax, dword ptr ss:76A688E0 <kernel32.CreateProcessW>
push eax
push dword ptr ss:[ebp+8]
call dword ptr ss:[ebp-7C]
test eax, eax
je 7415B2
xor eax, eax
```

[ebp+8]:L"C:\\windows\\Microsoft.NET\\Framework\\v4.0.30319\\RegSvcs.exe"

76A688E0 <kernel32.CreateProcessW>

push ebp

mov ebp, esp

pop ebp

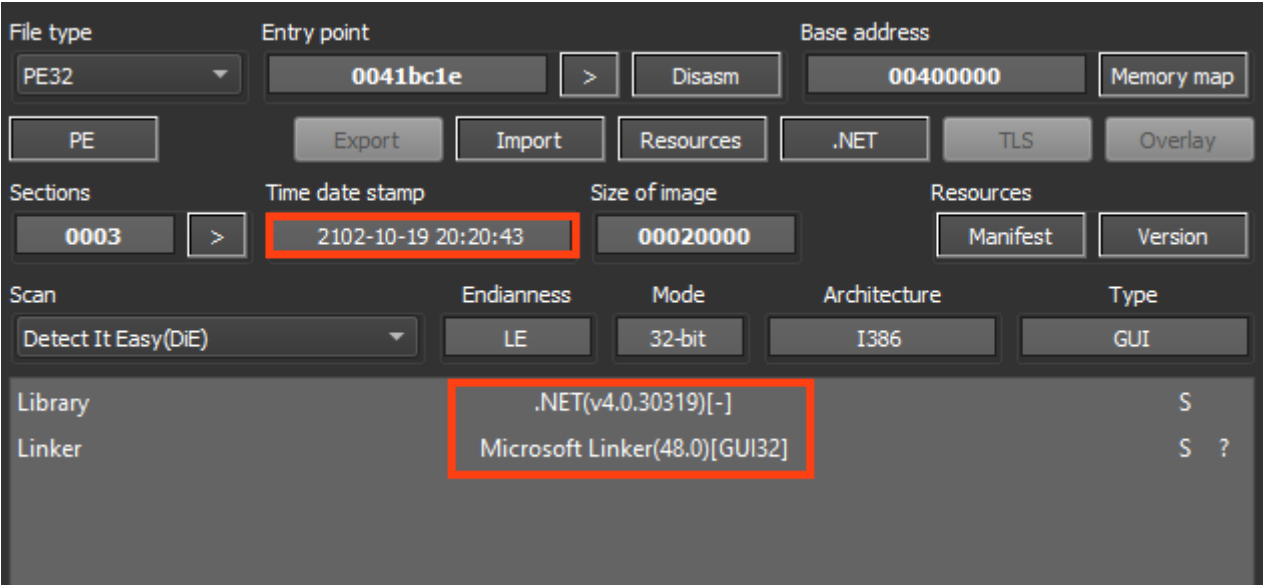
jmp dword ptr ds:[&CreateProcessW]

Loader injecting unpacked RedLine Stealer into another process.

## Stage 02 — Payload

RedLine Stealer is developed in .NET, and the compilation timestamp was altered in the binary, showing a date from the year 2102. [Formbook](#) was also using altered timestamp dates in its payloads, which is a common behavior for malware authors to deceive analysts/researchers.

Fortunately, RedLine Stealer uses a very nonsense date, which can be used for detection in Yara rules, for example.



RedLine Stealer payload details.

Once executed, the infostealer calls a function named “Check”. If this function returns true, the malware exits its process.

```
// Token: 0x06000076 RID: 118 RVA: 0x00002E3D File Offset: 0x0000103D
private static void Main(string[] args)
{
    dnlibDotNetCustomAttributeV.WriteLine();
}

// Token: 0x06000077 RID: 119 RVA: 0x00005DF4 File Offset: 0x00003FF4
public static void WriteLine()
{
    try
    {
        if (dnlibDotNetMDRawExportedTypeRowP.Check())
        {
            Environment.Exit(0);
        }
    }
}
```

RedLine Stealer “Check” function.

In summary, this function verifies if the malware is running in blocklisted countries, by comparing the country name with the OS region information.

```
// Token: 0x04000035 RID: 53
private static readonly string[] RegionsCountry = new string[]
{
    "Armenia",
    "Azerbaijan",
    "Belarus",
    "Kazakhstan",
    "Kyrgyzstan",
    "Moldova",
    "Tajikistan",
    "Uzbekistan",
    "Ukraine",
    "Russia"
};
```

This malware does not execute if any of these countries is detected:

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

We tested this by changing the OS language to Ukrainian. The malware uses the field “EnglishName” from the [.NET RegionInfo Class](#) to compare with the blocklist.

Наименование	Значение
regionInfo	{ru-UA}
CurrencyEnglishName	"Ukrainian Hryvnia"
CurrencyNativeName	"украинская гривна"
CurrencySymbol	"₴"
DisplayName	"Украина"
EnglishName	"Ukraine"
Geoid	0x000000F1
IsMetric	true
ISOCurrencySymbol	"UAH"

RedLine Stealer exits the process if a blocklisted country is found.

RedLine Stealer maintains a simple configuration, where the values are base64 encoded and encrypted with a rolling XOR algorithm.

```
// Token: 0x02000016 RID: 22
public static class SystemNetUnsafeNclNativeMethodsHttpApiHTTPREQUESTTOKENBINDINGINFO
{
    // Token: 0x04000010 RID: 16
    public static string IP = "GTsoFyMhGCIiOTddKRkGXiIIJRw9AwRc";

    // Token: 0x04000011 RID: 17
    public static string ID = "DgIkFyIMR2c=";

    // Token: 0x04000012 RID: 18
    public static string Message = "";

    // Token: 0x04000013 RID: 19
    public static string Key = "Wombles";

    // Token: 0x04000014 RID: 20
    public static int Version = 1;
}
```

RedLine Stealer configuration.

The decryption key used by this sample is “Wombles”, and we can use a simple Python script to retrieve the C2 address value:

```
>>>
>>> def rl_decode(value, key):
...     b64 = b""
...     for i, b in enumerate(b64decode(value)):
...         b64 += chr(b ^ key[i % len(key)]).encode()
...     return b64decode(b64)
...
>>>
>>> rl_decode(b"GTsoFyMhGCIiOTddKRkGXiIIJRw9AwRc", b"Wombles")
b'51.89.155.45:22595'
>>>
>>> _
```

Decrypting RedLine Stealer C2 address.

The “ID” value also uses the same algorithm:



```
>>> rl_decode(b"DgIkFyIMR2c=", b"Wombles")
b'bb.6.4'
>>>
```

Decrypting RedLine Stealer ID.

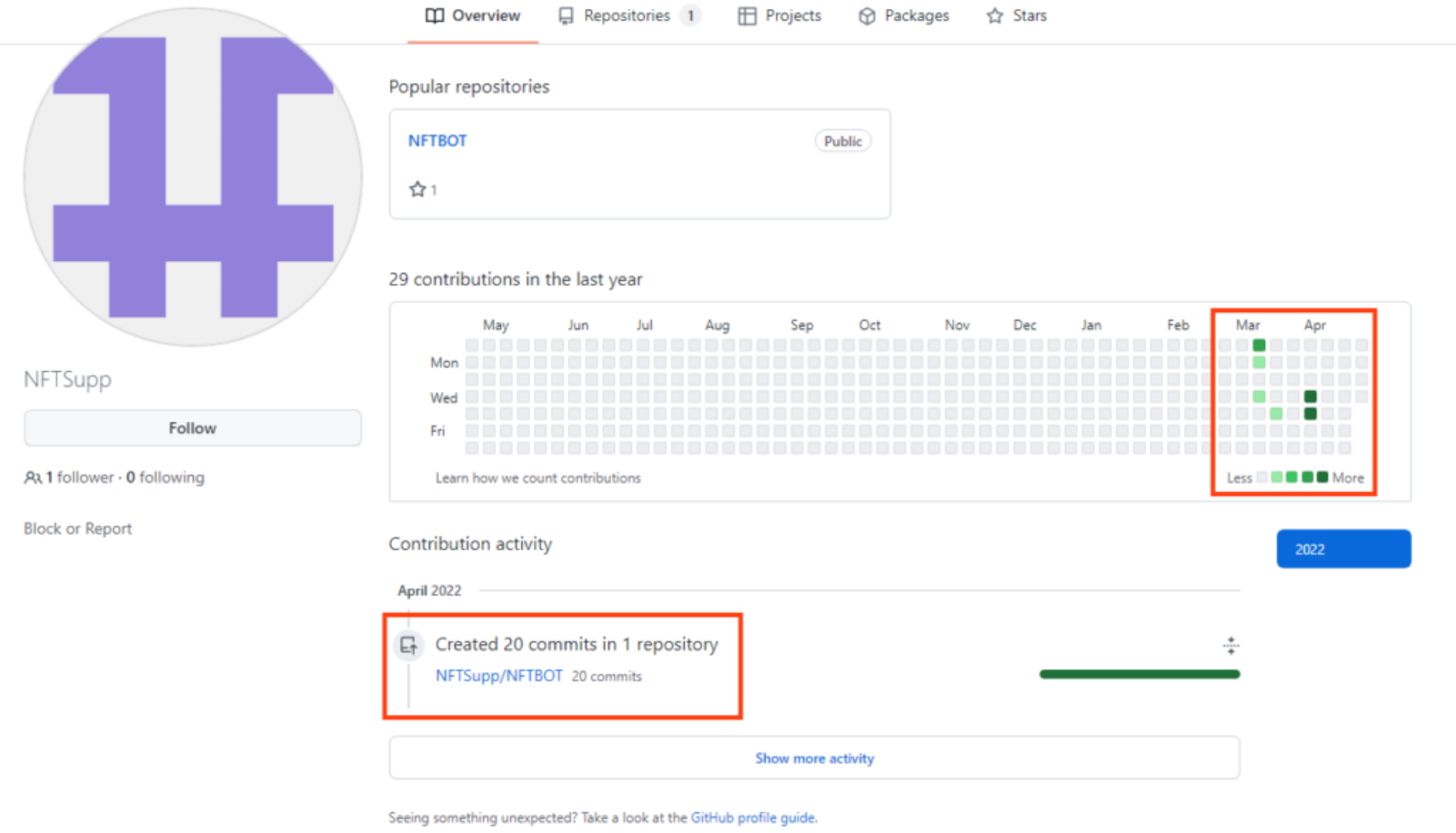
As previously mentioned, RedLine Stealer offers many capabilities to the attacker, including stealing [Discord](#) tokens.

```
// Token: 0x060000D7 RID: 215 RVA: 0x000081E4 File Offset: 0x000063E4
public override IEnumerable<dnlibDotNetWriterDebugDirectoryEntry> Id3()
{
    List<dnlibDotNetWriterDebugDirectoryEntry> list = new List<dnlibDotNetWriterDebugDirectoryEntry>();
    try
    {
        string id = Environment.ExpandEnvironmentVariables(new string(new char[]
        {
            '%',
            'a',
            'p',
            'p',
            'd',
            'a',
            't',
            'a',
            '%',
            '\\',
            'd',
            'i',
            's',
            'c',
            'o',
            'r',
            'd',
            '\\',
            'L',
            'o',
            'c',
            'a',
            'l',
            '\\',
            'S',
            't',
            'o',
            'r',
            'a',
            'g',
            'e',
            '\\',
            'l',
            'e',
            'v',
            'e',
            'l',
            'd'
        }));
        // ... (rest of the function code)
    }
}
```

RedLine Stealer function that reads Discord tokens.




















### More Files From the Same Campaign

Looking at the GitHub account (“NFTSupp”) that owns the repository where the file linked on the YouTube videos is hosted, we can see that the activities started in March, 2022.














GitHub account and repository hosting RedLine Stealer.

Aside from the files we analyzed in this blog post contained within “BinanceNFT.bot v.1.3.zip”, there are 15 additional compressed files hosted in the same repository (“NFTBOT”), where two of them are password protected (“45.rar” and “Upload.Openbot.rar”).

 <b>NFTSupp</b> Add files via upload		d1b5de8 20 days ago	 27 commits
	45.rar	Add files via upload	20 days ago
	AxieFarmBot.v5.7.zip	Add files via upload	last month
	BinanceNFT.bot v.1.3.zip	Add files via upload	21 days ago
	MIR4.FarmBOT.v8.3.zip	Add files via upload	21 days ago
	MIR4_BOT.rar	Add files via upload	last month
	MetaMask_Bot.V2.9.rar	Add files via upload	21 days ago
	NFT.bot v.1.3.zip	Add files via upload	2 months ago
	NFT.bot.OpenSea v.1.3.zip	Add files via upload	2 months ago
	NFT.bot.v5.7.zip	Add files via upload	21 days ago
	New.OpenSea.bot.v.3.5.rar	Add files via upload	20 days ago
	OpenSea.bot.v1.6.rar	Add files via upload	20 days ago
	OpenSea.bot.v1.6.zip	Add files via upload	20 days ago
	OpenSea.bot.v2.6.zip	Add files via upload	20 days ago
	README.md	Initial commit	2 months ago
	Upload.Openbot.rar	Add files via upload	20 days ago
	Upload.Openbot.zip	Add files via upload	20 days ago
	upload.opensea.zip	Add files via upload	20 days ago

Compressed files within the same repository.

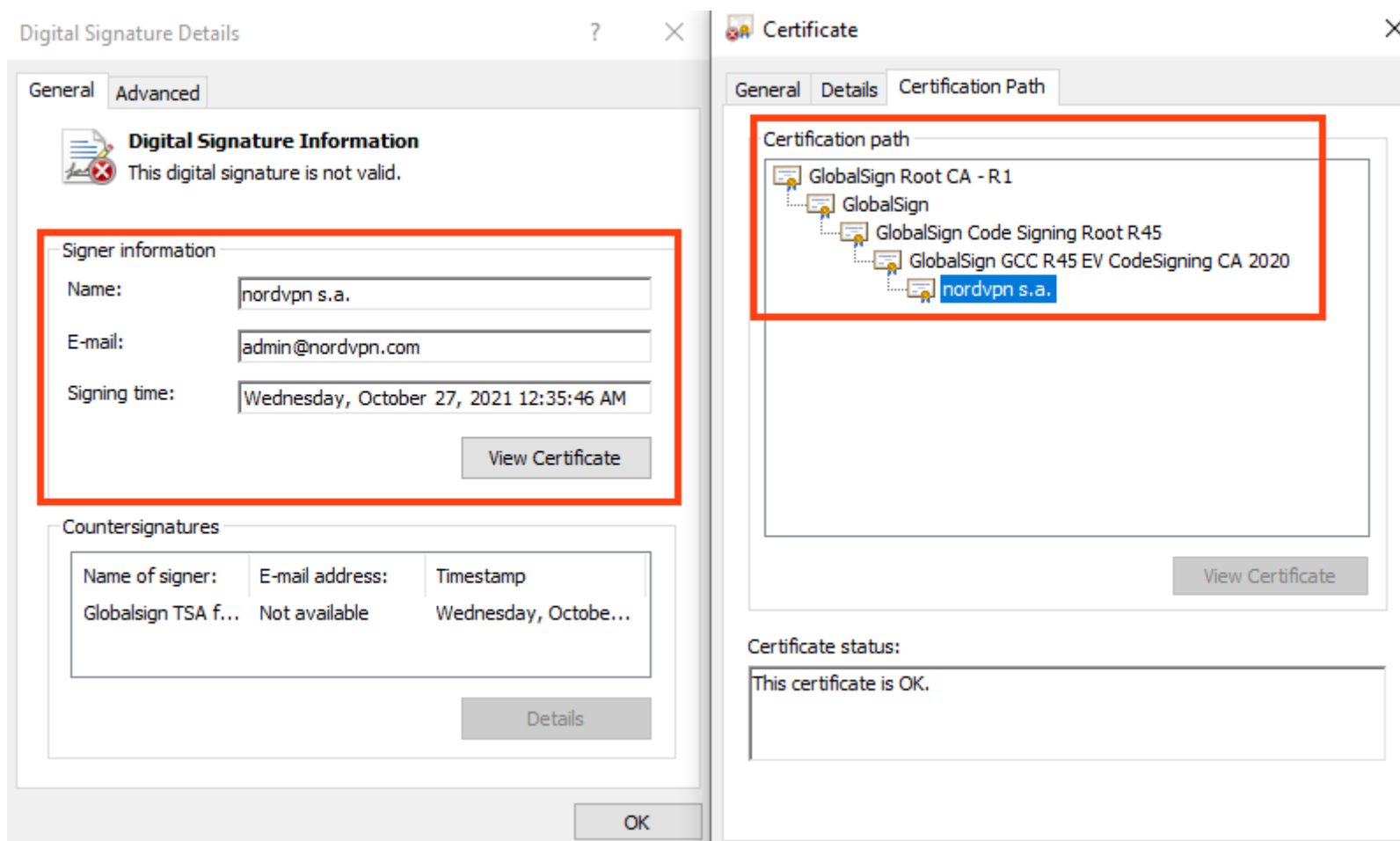
Within these compressed files, we found five distinct RedLine Stealer loaders.

Filename	MD5
 AxieFarmBot.v5.7.exe	4d77e265722624b5d4d1841d45c7c677
 OpenSea.bot.v1.6.exe	500a62b980fe1089beb63e14d61b244a
 OPENSEA.Nft.bot.exe	500a62b980fe1089beb63e14d61b244a
 BinanceNFT.bot v.1.3....	8c24b7746d006c63db615dd43187651b
 MetaMask_Bot.V2.9.exe	8c24b7746d006c63db615dd43187651b
 MIR4.farmbot.v8.3.exe	8c24b7746d006c63db615dd43187651b
 NFT.bot.v5.7.exe	8c24b7746d006c63db615dd43187651b
 OpenSea.bot.v.3.5.exe	8c24b7746d006c63db615dd43187651b
 OpenSea.bot.v2.6.exe	8c24b7746d006c63db615dd43187651b
 Mir4_Bot.v1.9.exe	d3f749cc20369e215d59f9d8bfde1a41
 OpenSea Bot.exe	d3f749cc20369e215d59f9d8bfde1a41
 OpenSea Bot2.exe	d3f749cc20369e215d59f9d8bfde1a41
 OpenSea.bot.v1.62.exe	f0d65470988478921ff40b6fb3def616

Different RedLine Stealer loaders in the same repository.

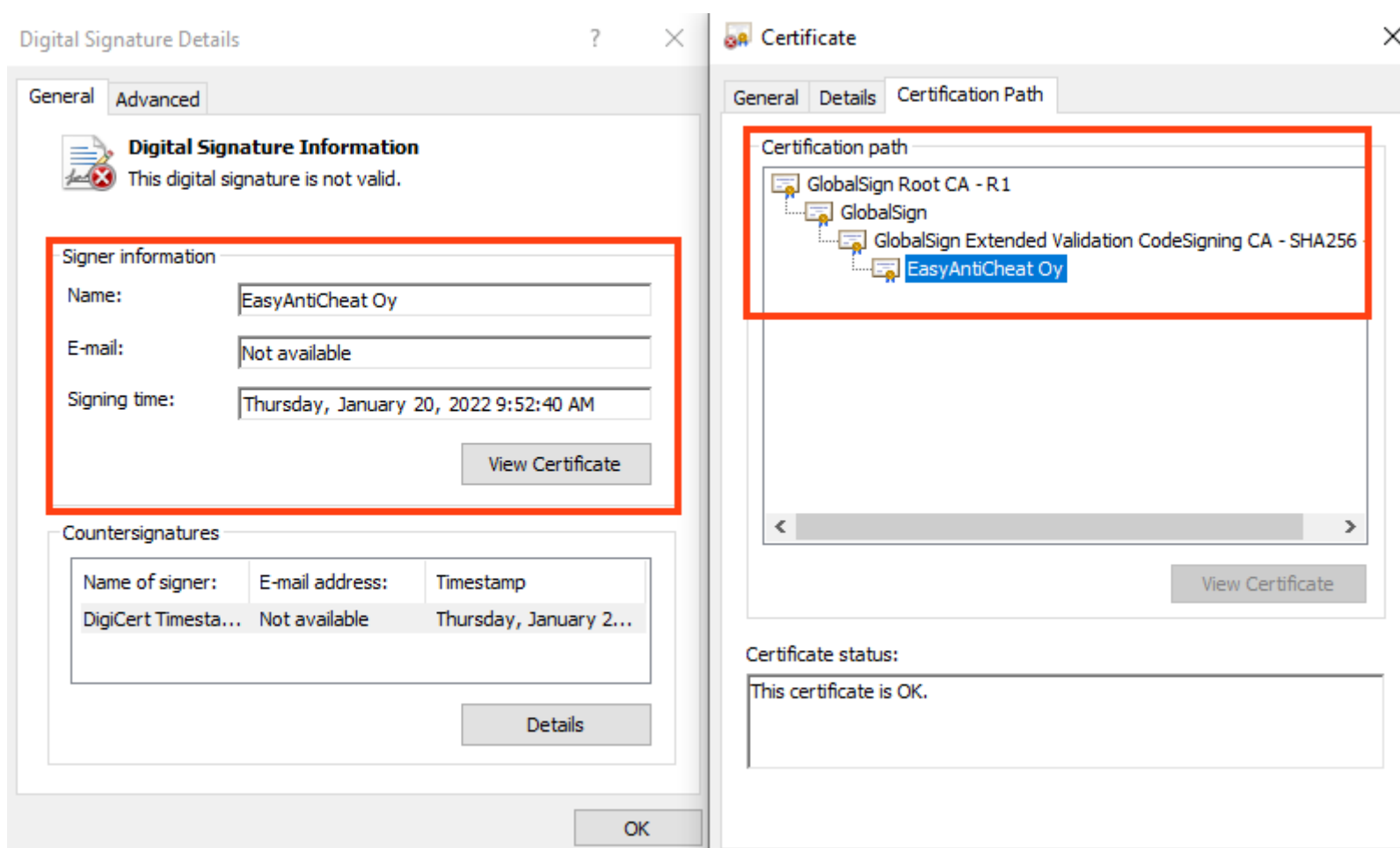
All five loaders we analyzed are slightly different, but they all unpack and inject RedLine Stealer in a similar way, as we described earlier in this analysis. The oldest sample we found was likely compiled on March 11, 2022 and the newest one on April 7, 2022.

Furthermore, two out of five files are digitally signed, which may bypass some antivirus engines. The first one seems to be using a signature from “[NordVPN S.A.](#)”



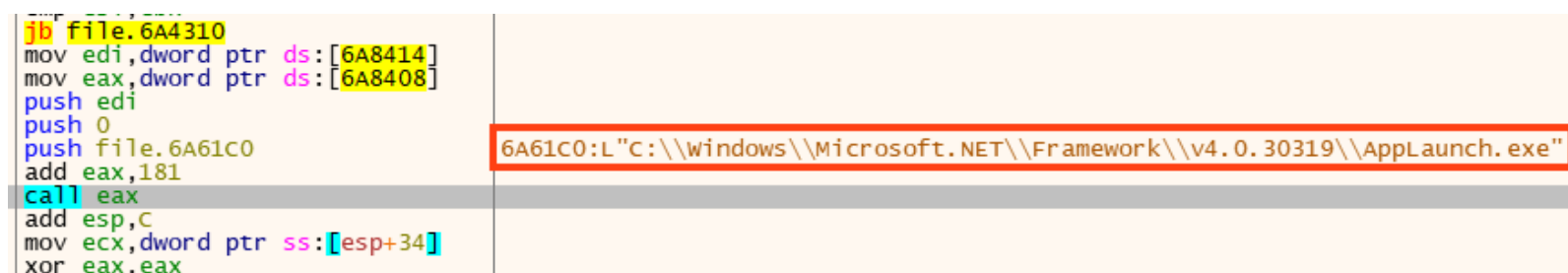
RedLine Stealer digitally signed.

And the second is signed for “[EasyAntiCheat Oy](#)”.



RedLine Stealer digitally signed.

Also, one of the loaders is injecting the payload into “AppLaunch.exe” instead of “RegSvcs.exe”.



RedLine Stealer being injected into AppLaunch process.

We found four distinct RedLine Stealer payloads from these five loaders, which are all sharing the same C2 address.

## Conclusions

Although RedLine Stealer is a low-cost malware, it offers many capabilities that could cause serious damage to its victims, such as the loss of sensitive data. RedLine Stealer was already known for abusing YouTube videos to spread through fake themes, however, we saw in this campaign that the attacker is also abusing GitHub in the attack flow, to host the payloads.

## Protection

Netskope Threat Labs is actively monitoring this campaign and has ensured coverage for all known threat indicators and payloads.

- Netskope Threat Protection
  - Win32.Trojan.RedLineStealer
- Netskope Advanced Threat Protection provides proactive coverage against this threat.
  - Gen.Malware.Detect.By.StHeur indicates a sample that was detected using static analysis
  - Gen.Malware.Detect.By.Sandbox indicates a sample that was detected by our cloud sandbox

## IOCs

All the IOCs related to this campaign and the Yara rules can be found in our [GitHub repository](#).



< [Threat Labs Next Story](#) > < [Back Next](#) >

About the author Gustavo Palazolo is an expert in malware analysis, reverse engineering and security research, working many years in projects related to electronic fraud protection. He is currently working on the Netskope Research Team, discovering and analyzing new malware threats. Gustavo Palazolo is an expert in malware analysis, reverse engineering and security research, working many years in projects related to electronic fraud protection. He is currently working on the Netskope Research Team, discovering and analyzing new malware threats. [Read Gustavo Palazolo's full Bio > More Articles by Gustavo Palazolo > Read full Bio > More articles > Related Articles](#)Threat Labs By Gustavo Palazolo [Emotet: New Delivery Mechanism to Bypass VBA Protection](#)





[Read article](#)Threat Labs By

Paolo Passeri [Cloud Threats Memo: What We Can Learn From the Top 15 Routinely Exploited Threats of 2021](#)



[Read article](#)Threat Labs By

Allen Funkhouser [Multi-Factor Authentication \(MFA\) Bypass Through Man-in-the-Middle Phishing Attacks](#)



[Read article](#) [Load More](#)

[Articles](#) [Contact Us](#)[Contact Us](#)

We'd love to hear from you!

Loading...