Mars, a red-hot information stealer

By Pierre Le Bourhis, Quentin Bourgue and Sekoia TDR

Published: 2022-04-07 · Archived: 2026-04-05 22:55:47 UTC

lang: en_US

Table of contents

Mars Stealer is an information stealer sold on underground forums by *MarsTeam* since June 22, 2021, with the malware-as-a-service model. The malware capabilities are those of a classic stealer with a focus on cryptocurrency theft. As a quick summary, Mars Stealer is able to:

- collect data from several browsers (passwords, cookies, credit cards, *etc*.);
- steal credentials from crypto plugins, crypto wallets and 2FA plugins;
- grab files;
- fingerprint the infected host.

It shares code with other information stealers including Arkei, Oski and Vidar.

Given its interesting functionalities, its ease of use and reasonable price, the Mars Stealer malware has become popular on several underground forums. Moreover, the presumed developers regularly release new versions of the

malware to fix some bugs and especially to improve the Mars Stealer capabilities in terms of data collection and defense evasion.

Mars Stealer has been recently brought to light by 3xp0rt's in-depth analysis[1] and the release of a cracked version. In the blog post, 3xp0rt wrote an analysis of a Mars Stealer sample of an early version by exposing the different obfuscation methods and the data targeted by the malware on the infected hosts. A few days later, some members of the infosec community shared their findings on the builder and the administration panel of the information stealer. In recent days, some campaigns distributing Mars Stealer have been publicly described[2] [3].

SEKOIA.IO analysts have been monitoring the threat on underground forums to be up-to-date on the latest developments. We have also recently carried out an in-depth analysis of samples of different versions of Mars Stealer, and noticed many changes in the obfuscation techniques.

Why investigate the Mars Stealer malware?

Information stealers are a threat to be considered, as many threat actors are using them to harvest credentials and other personal information. The stolen data can then be sold on underground forums and then possibly leveraged in "Big Game Hunting" operations, as the Lapsus$ threat group does[4].

Among these information stealers, Mars Stealer has become an emerging threat in recent months.

First of all, the malware is widely advertised on numerous underground forums and the publications reach a large audience. Furthermore, users of Mars Stealer usually give good feedback and do not hesitate to recommend it when forum members are looking for an information stealer. The Mars Stealer malware therefore appeared in our Dark Web monitoring during the second half of 2021.

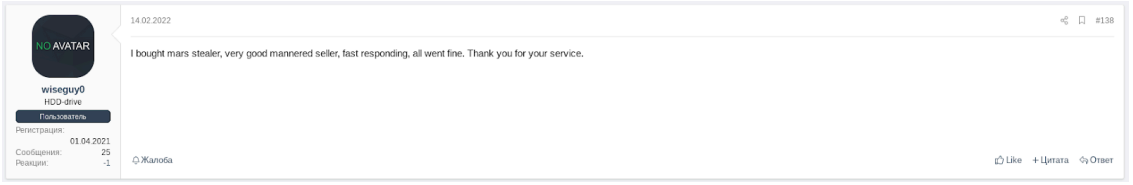


Figure 1. Positive feedback on the Mars Stealer software and service on the XSS forum

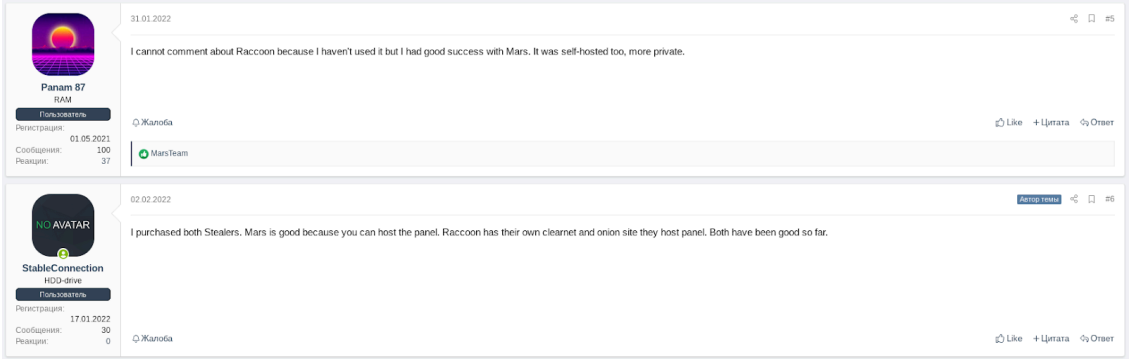


Figure 2. Users' comments advising Mars Stealer on the XSS publication named "Raccoon or Mars Stealer"

Dynamic of ransomware activity in Q1 2022

It is worth noting that Mars Stealer is under continuous development and the project is professionally maintained, which makes it attractive and trustworthy to potential clients. Indeed, the presumed developers (*MarsTeam*) regularly collect user feedback on posts on the underground forums, on the Telegram support channel or on Jabber. *MarsTeam* then takes this feedback into account to make improvements, new features or bug fixes. New versions are regularly released and accompanied by a changelog to list the notable changes made to the Mars Stealer agent and also the panel.

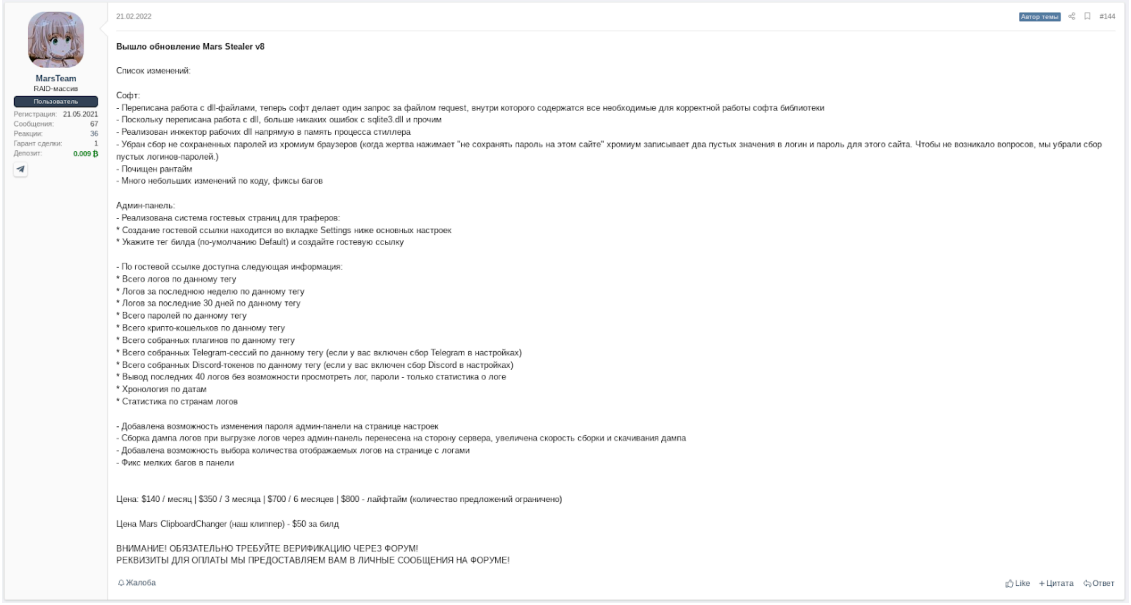


Figure 3. Changelog of the Mars Stealer version 8 published on the XSS forum

Furthermore, the malware appeared in OSINT reports in early 2022, especially with the great 3xp0rt's in-depth analysis. Since then, its occurrence has increased in the infosec community because of the publication of its

builder and above all, some campaigns distributing Mars Stealer have been brought to light in the Cyber Threat Intelligence sphere.

Last, the abrupt shutdown of Raccoon Stealer operations, which is one of the most widespread stealers, leaves a significant part of the market for the information stealers. Indeed, on March 25, 2022, the profile *raccoonstealer* announced on the Russian-speaking underground forum XSS that the group operating Raccoon Stealer closed the project for an undetermined period of time. This unexpected shutdown is due to the loss of a developer of the project Raccoon Stealer during the “special operation”, in reference to the Russian war in Ukraine.

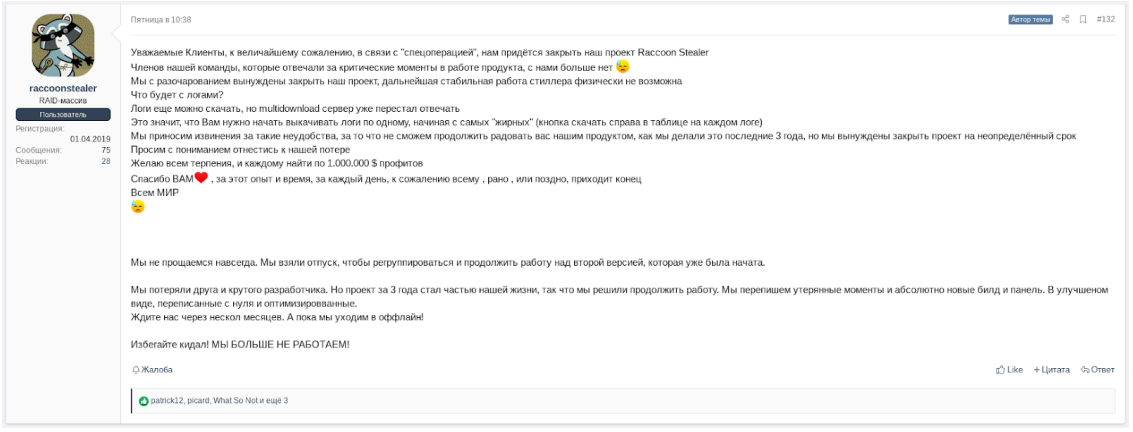


Figure 4. Raccoonstealer’s statement on the shutdown of the Raccoon Stealer project on the XSS forum

A publication of *MarsTeam* on the XSS forum fully confirmed this hypothesis. On March 24, 2022, *MarsTeam* responded to two potential clients:
“*Guys, deal with the message backlog, will reply to all within 24 hours. A lot of people came from Raccoon. We do not have time to process all messages physically.*“” (translated from Russian)

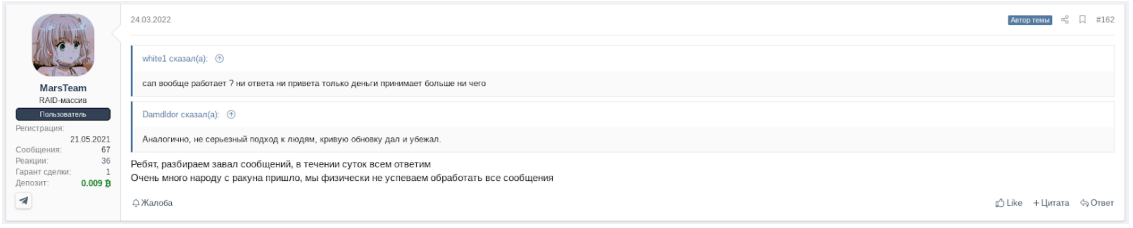


Figure 5. MarsTeam mentioning a wave of clients coming from Raccoon on the XSS forum

For all of the above reasons it seems relevant to us to monitor the Mars Stealer threat, to stay-up-to-date on the development of the malware and to track Indicators of Compromise (IoCs) to detect Mars Stealer. These two points are the subject of the next parts.

A journey with the malware developers

In this part, we analyze the publications and the activities of the presumed Mars Stealer developers (*MarsTeam*) on underground forums. According to our Dark Web monitoring, *MarsTeam* is active on numerous underground forums including XSS.is, lolz.guru and bhf.io. However, we focus on the XSS forum as *MarsTeam* publishes first, and more frequently on this one.

On May 21, 2021, the Mars Stealer team joined the XSS underground forum under the name *MarsTeam* with a deposit of 0.009 Bitcoin ($336 at May 2021 exchange rate). One month later, *MarsTeam* opened a new discussion whose title translated from Russian is “Mars Stealer – a native, non-resident stealer with loader and stealer functionality”.

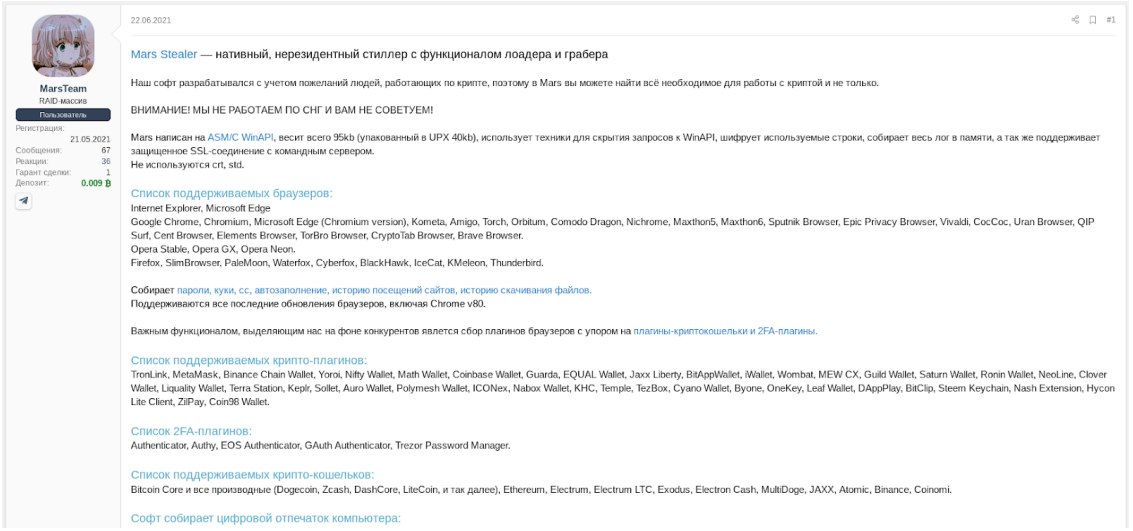


Figure 6. First publication of MarsTeam advertising Mars Stealer

The publication introduces Mars Stealer as “*a new software developed for people working with crypto*” – to be interpreted: for people who want to steal cryptocurrencies. The malware is sold with the malware-as-a-service model for $140 per month.

Mars Stealer capabilities

As mentioned in the introduction, the Mars Stealer capabilities advertised by *MarsTeam* are those of a classic information stealer with a specific focus on cryptocurrency theft. Our technical analysis of Mars Stealer confirmed that the applications targeted by the malware samples are those described in the *MarsTeam* publications.

The stealer collects personal information from numerous browsers: passwords, cookies, credit cards, autofill data, history of websites visited and files downloaded. The list of supported browsers is quite wide, from the most popular (Google Chrome, Internet Explorer, Microsoft Edge, Firefox, *etc.*) to the less common. Mars Stealer collects this data in the default path of the different browser user data, or browser profiles.

The theft of cryptocurrencies is one of its distinguishing features, *MarsTeam* specifies that: “*Important feature that makes us stand out from competitors is the collection of browser plugins with an emphasis on cryptocurrency and 2FA plugins*” (translated from Russian). Indeed, the list of targeted crypto plugins is very long with more than 40 references, including the most used (Coinbase, MetaMask, Binance). The same goes for the list of crypto wallets and 2FA plugins targeted by the information stealer.

The malware also fingerprints the infected host to collect information about hardware, installed software and other personal information. Mars Stealer collects these data using WinAPI calls such as *GetSystemInfo*, *GetCurrentProcess* or by requesting the Windows Registry keys such as

HARDWARE\DESCRIPTION\System\CentralProcessor\0,
SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall.

Last but not least, Mars Stealer acts as a file grabber which is easily customizable by the operator. *MarsTeam* describes it as "*a powerful feature*" as the start path, file extension, file size and the recursive search can be set up. It also captures a screenshot of the victim's desktop, loads files on the infected machine and executes them with arguments.

High-quality service for malware operators

In addition to the stealing agent, Mars Stealer is sold with the administration panel and customer support. It is worth noting that the malware operator must host the Command & Control (C2) server and associated panel on its own server to access the stolen data. *MarsTeam* insists on the fact that all the traffic is available only for the customer and does not pass through a server of the Mars Stealer developers. This functionality is often requested by information stealer users who want to control and own all the data.

In the first *MarsTeam*'s publication, the Mars Stealer administration panel is described as "*a powerful data search functionality*" and the user experience seems to be a priority in the development of Mars Stealer. *MarsTeam* describes multiple capabilities of the panel on which the operator can easily manage, sort, filter, and remove the logs.

Moreover, the purchase of Mars Stealer includes a high-quality service including the support of all issues, the access to the customer chat and the new releases of the malware.

All the malware capabilities accompanied by a user-friendly interface and quality support make Mars Stealer very attractive and popular to attackers on underground markets.

Underground forum presence

The responsiveness of the Mars Stealer team on the underground forums sends a positive message to potential customers. *MarsTeam* is not only responsive to user requests, but also very active on several underground forums. Since the first communication on Mars Stealer in June 2021, *MarsTeam has regularly* published changelogs to announce new malware releases, as shown by the following figure. Special offers or teasing new features are also the subject of many of *MarsTeam*'s posts.

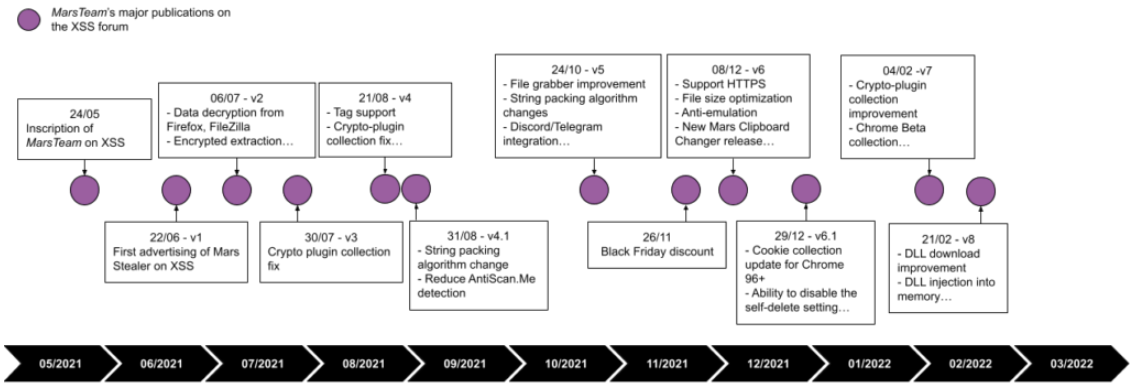


Figure 7. Timeline of MarsTeam's major publications on the XSS forum

Let us consider the releases of new Mars Stealer versions on the XSS forum. Most of these changelogs include new features, improvements, or bug fixes both on the agent and the administration panel. We noticed that the early versions did not have a version number. The official versioning started with version 4 released on August 21, 2021, so we approximately associated the first versions to *MarsTeam*'s early changelogs. Most of them are divided into two sections: on the software and the administration panel.

Software improvements are focused on the coverage of targeted applications by adding new browsers or new crypto plugins, as well as the upgrade of defense evasion techniques. For example, version 7 introduced the support of the Google Beta browser and numerous crypto plugins, which increased the number of collectable crypto plugins to 101. Version 6 brings new evasion methods for VM and antiviruses.

Improvements achieved on the web administration panel aim to facilitate the user experience by giving new possibilities of sorting, filtering and searching for stolen data. The Mars Stealer developer also optimized the performance related to the SQL database.

The analysis of the *MarsTeam*'s publications on different underground forums shows that the presumed Mars Stealer professionally works on software development, but also on communication and support. This conscientiousness seems to seduce the attackers who want to buy and use an information stealer well maintained and in continuous enhancement.

How to collect Mars Stealer IoCs?

In the previous part, we showed how Mars Stealer has become an emerging threat and why we must be interested in it. In this part we see how to collect Mars Stealer IoCs based on YARA signatures, C2 server tracker and the automated extraction of the malware configuration. The last method requires a technical analysis of Mars Stealer and its versions which implement different obfuscation techniques. This in-depth analysis is the subject of the part named Mars Stealer objective C2.

Collecting Mars Stealer samples

Early versions

Before analyzing the different versions of the Mars Stealer malware, a first step consists of collecting the malware samples. We have therefore written a YARA rule based on the sample shared in the 3xp0rt analysis to identify other Mars Stealer samples.

A reliable method to detect samples of a malware family consists in searching for operation code (opcode) patterns used in the deobfuscation routine. In the early versions, the malware implemented a deobfuscation function which first decodes base64-encoded strings and then decrypts RC4-encrypted strings. This algorithm is applied on numerous obfuscated strings as shown in the following figure, which corresponds to the function loading obfuscated strings.

```
55                                      load_obfuscated_str proc near
8B EC                                   push    ebp
68 C4 31 41 00                          mov     ebp, esp
E8 53 24 00 00                          push    offset aT1aor4xzdwsffk ; "t1AOR4xZDwsfFK++SuZNpKGhVA=="
83 C4 04                                call    deobfuscate_func
A3 24 76 41 00                          add     esp, 4
68 E4 31 41 00                          mov     dec_date, eax
E8 41 24 00 00                          push    offset a7rxvb45zeg ; "7RxVB45ZEg=="
83 C4 04                                call    deobfuscate_func
A3 88 73 41 00                          add     esp, 4
68 F4 31 41 00                          mov     dec_http, eax
E8 2F 24 00 00                          push    offset a5gdohmytxl5evf ; "5gdOHMYTXl5EVfv9VLoI+g=="
83 C4 04                                call    deobfuscate_func
A3 CC 77 41 00                          add     esp, 4
C7 05 50 71 41 00 AC 30 41 00           mov     dec_domain, eax
68 10 32 41 00                          mov     dword_417150, offset a86223203794583 ; "86223203794583053453"
E8 13 24 00 00                          push    offset aQhtogtfee0tfvq ; "qhtOGtFEE0tFVQ=="
83 C4 04                                call    deobfuscate_func
A3 AC 72 41 00                          add     esp, 4
68 24 32 41 00                          mov     dec_uri, eax
E8 01 24 00 00                          push    offset aOghuwjeesbqitf ; "oGhUWJEeSBQITfquX7QIrr75EbJErz4="
83 C4 04                                call    deobfuscate_func
A3 30 71 41 00                          add     esp, 4
68 48 32 41 00                          mov     dword_417130, eax
E8 EF 23 00 00                          push    offset a6hhegq  ; "6hhEGQ=="
83 C4 04                                call    deobfuscate_func
A3 AC 76 41 00                          add     esp, 4
68 54 32 41 00                          mov     dec_open, eax
E8 DD 23 00 00                          push    offset aQhhufdgfxhrevo ; "qhhUFdgfXhReVOPnDrlOuv/9CA=="
83 C4 04                                call    deobfuscate_func
A3 7C 77 41 00                          add     esp, 4
68 74 32 41 00                          mov     dec_public_sqlite3, eax
E8 CB 23 00 00                          push    offset aXlj9j8yzwklmsm ; "xlJ9J8YZWklMSMvvDr0h5+r9DfwE9GV29Hk="
83 C4 04                                call    deobfuscate_func
A3 BC 76 41 00                          add     esp, 4
68 9C 32 41 00                          mov     dec_programa_data_sqlite3, eax
E8 B9 23 00 00                          push    offset aQhhufdgfxhrlvR ; "qhhUFdgfXhRLV+rrGLBOuv/9CA=="
                                        call    deobfuscate_func
```

Figure 8. Function loading obfuscated strings in a Mars Stealer early version sample

Four instructions are repeated for each obfuscated data: *push*, *call*, *add* and *mov*. A YARA rule identifying this version of Mars Stealer can be written based on the repetition of these opcodes. Adding to the detection pattern some specific strings can fine-tune the rule in order to identify only Mars Stealer binaries. Here is a possible YARA rule to find Mars Stealer samples:

```
rule infostealer_win_mars_stealer_early_version {
   meta:
       description = "Identifies samples of Mars Stealer early version based on opcodes of the function loading
       source = "SEKOIA.IO"
       reference = "https://blog.sekoia.io/mars-a-red-hot-information-stealer/"
       classification = "TLP:WHITE"
       hash = "7da3029263bfbb0699119a715ce22a3941cf8100428fd43c9e1e46bf436ca687"

   strings:
       $dec = {a3 ?? ?? ?? ?? 68 ?? ?? ?? ?? e8 ?? ?? 00 00 83 c4 ??}

       $api00 = "LoadLibrary" ascii
       $api01 = "GetProcAddress" ascii
       $api02 = "ExitProcess" ascii
       $api03 = "advapi32.dll" ascii
       $api04 = "crypt32.dll" ascii
       $api05 = "GetTickCount" ascii
       $api06 = "Sleep" ascii
       $api07 = "GetUserDefaultLangID" ascii
       $api08 = "CreateMutex" ascii
       $api09 = "GetLastError" ascii
       $api10 = "HeapAlloc" ascii
       $api11 = "GetProcessHeap" ascii
```

```
        $api12 = "GetComputerName" ascii
        $api13 = "VirtualProtect" ascii
        $api14 = "GetUserName" ascii
        $api15 = "CryptStringToBinary" ascii

        $str0 = "JohnDoe" ascii

    condition:
        uint16(0)==0x5A4D and
        #dec > 400 and 12 of ($api*) and $str0
}
```

Figure 9. YARA rule identifying Mars Stealer early version samples

We shared the rule on sample sharing platforms and collected several results from this YARA rule. As expected from our Dark Web monitoring, we observed different Mars Stealer versions based on their deobfuscation routine among our collected samples. We also noticed several samples for which a new PE section name appeared: *LLCPPC*. More details on the different versions can be found in the following technical analysis.

LLCPPC versions

The PE section name *LLCPPC* is a highly discriminating factor of the Mars Stealer malware. We can therefore easily identify Mars Stealer samples using this characteristic in a YARA rule using the PE module:

```
 import "pe"

rule infostealer_win_mars_stealer_llcppc {
    meta:
        description = "Identifies samples of Mars Stealer based on the PE section name LLCPPC."
        source = "SEKOIA.IO"
        reference = "https://blog.sekoia.io/mars-a-red-hot-information-stealer/"
        classification = "TLP:WHITE"
        hash = "fd92fe8a4534bc6e14e177fee38a13f771a091fa6c7171fcee2791c58fbecf40"

    condition:
        uint16(0)==0x5A4D and
        for any i in ( 0..pe.number_of_sections-1 ): (
                pe.sections[i].name == "LLCPPC" and pe.sections[i].raw_data_size < 5000 )
}
```

Figure 10. YARA rule identifying Mars Stealer samples based on the PE section name

For information, *LLCPPC* is a profile on the underground forum lolz.guru that reverses engineer some popular malware (Redline, Mars Stealer, DCRat, X-FILES and SHurkSteal) in order to debunk the misleading information used to advertise the product.

The developer of Mars Stealer probably named the PE section of the malware to rag *LLCPPC* after the analysis of the malware which revealed untruths in the malware's advertisement. On August 24, 2021, *LLCPPC* published a technical analysis of a Mars Stealer sample with the title "*Mars Stealer is the worst stealer | How the shit coder cheats on you*".

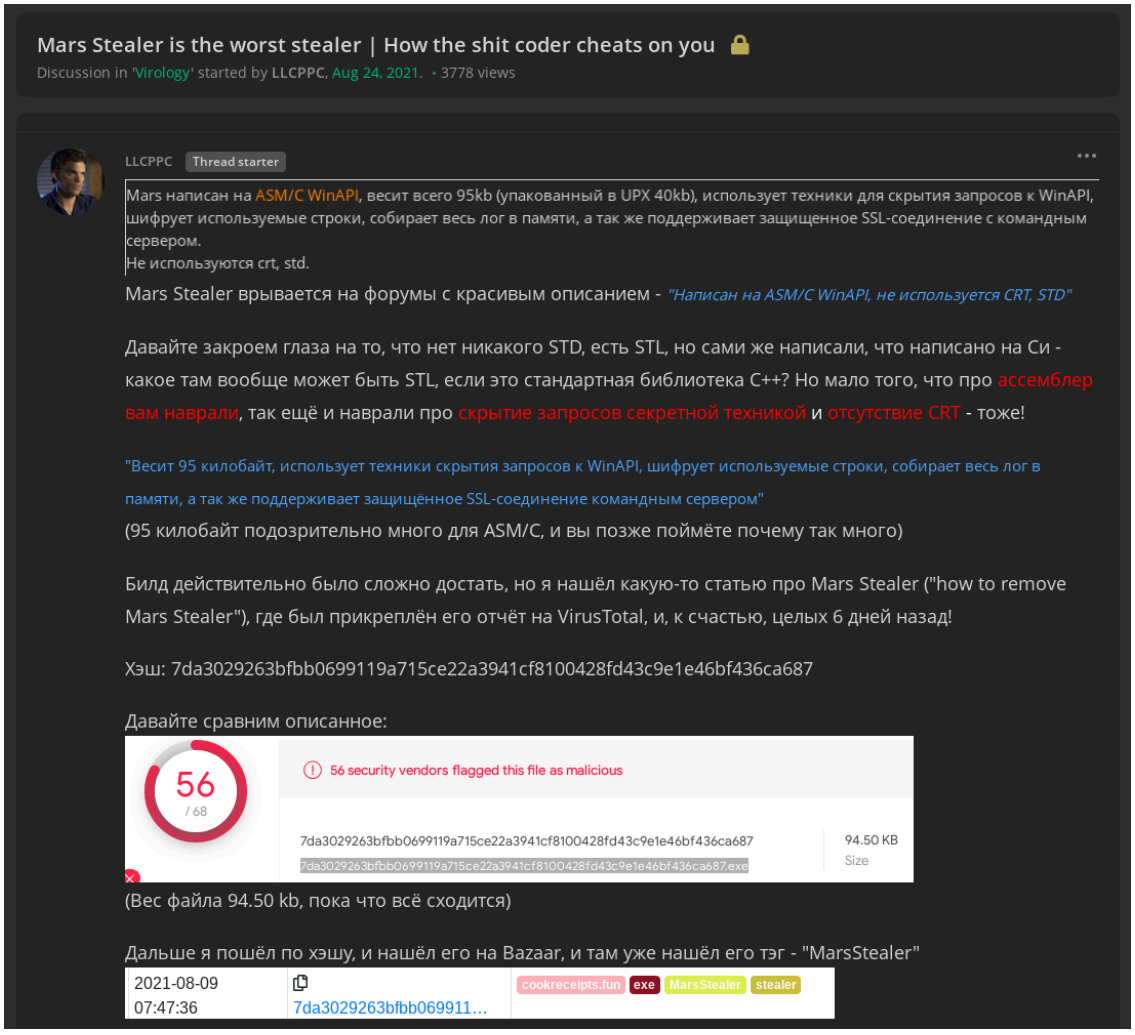


Figure 11. LLCPPC's publication debunking a Mars Stealer sample on the lolz.guru forum

In this analysis, *LLCPPC* concludes that only encryption and the secure import address table are the pros of Mars Stealer while the list of cons is much longer. Among them, *LLCPPC* raised the non-optimized code, the lack of debugging and virtualization evasion, the absence of multithreading, etc. As shown by the previous figure, *LLCPPC* also mentioned that the Mars Stealer sample is well detected by VirusTotal and Any.Run analysis, unlike *MarsTeam* stated.

Latest version (version 8)

The LLCPPC section disappeared in the latest version of Mars Stealer, which corresponds to version 8 according to the *MarsTeam* releases on XSS. Indeed, samples of the latest Mars Stealer version only send one request to retrieve the DLLs (*freebl3.dll*, *mozglue.dll*, *msvcp140.dll*, *nss3.dll*, *softokn3.dll*, *sqlite3.dll* and *vcruntime140.dll*). These files are linked to legitimate third-party DLLs allowing Mars Stealer to collect data from the infected host. As written by *MarsTeam* in the version 8 release notes "*now the software makes one request after the request file*,

which contains all necessary libraries for correct work of the software" (translated from Russian), it therefore corresponds to version 8 samples.

Communication with the C2 server is performed over HTTP, and since version 6, the stealer can use HTTPS.

No.	Time	Source	Destination	Protocol	Length	Info
2515	4664.9062074…	192.168.122.151	194.87.218.39	HTTP	156	GET /RyC66VfSGP.php HTTP/1.1
2517	4665.0109517…	194.87.218.39	192.168.122.151	HTTP	662	HTTP/1.1 200 OK (text/html)
2519	4665.0280519…	192.168.122.151	194.87.218.39	HTTP	171	GET /request HTTP/1.1
2975	4665.8855121…	194.87.218.39	192.168.122.151	HTTP	8294	HTTP/1.1 200 OK
3074	4666.8162891…	192.168.122.151	194.87.218.39	HTTP	6616	POST /RyC66VfSGP.php HTTP/1.1
3134	4667.2382048…	194.87.218.39	192.168.122.151	HTTP	366	HTTP/1.1 200 OK

Figure 12: HTTP communication with the C2 server

1. Implant sends a GET request to the C2 URL to grab its configuration.
2. Implant fetches all DLLs on the "*/request*" endpoint, the libraries are zipped (*c.f.* figure 13).
3. Stolen data are posted to the C2 on the same URL used in step (1).

```
└─$ unzip -v request
Archive:  request
 Length   Method    Size  Cmpr    Date    Time   CRC-32   Name
--------  ------  ------- ---- ---------- ----- --------  ----
  144848  Defl:N    78085  46% 2022-01-30 15:16 760685c5  softokn3.dll
  645592  Defl:N   328449  49% 2022-01-30 15:16 9f30a75e  sqlite3.dll
   83784  Defl:N    46569  44% 2022-01-30 15:16 9bb5124b  vcruntime140.dll
  334288  Defl:N   156303  53% 2022-01-30 15:16 b698d0ca  freebl3.dll
  137168  Defl:N    75691  45% 2022-01-30 15:16 e28a5e21  mozglue.dll
  440120  Defl:N   156208  65% 2022-01-30 15:16 97bcf588  msvcp140.dll
 1246160  Defl:N   723576  42% 2022-01-30 15:16 9f24f4e3  nss3.dll
--------          ------- ---                            -------
 3031960          1564881  48%                            7 files
```

Figure 13: HTTP response on /request that contains all DLLs zipped

To identify and collect these samples, we can again write a YARA rule based on the string deobfuscation routine since the discriminating section name is no longer used. From our technical analysis, we identify the deobfuscation routine based on XOR keys, which is further detailed in the "Mars Stealer objective C2" section. The algorithm consists in xoring each obfuscated string and its corresponding key.

Our resulting YARA rule is:

```
rule infostealer_win_mars_stealer_xor_routine {
   meta:
       description = "Identifies samples of Mars Stealer based on the XOR deobfuscation routine."
       source = "SEKOIA.IO"
       reference = "https://blog.sekoia.io/mars-a-red-hot-information-stealer/"
       classification = "TLP:WHITE"
       hash = "4bcff4386ce8fadce358ef0dbe90f8d5aa7b4c7aec93fca2e605ca2cbc52218b"

   strings:
       $xor = {8b 4d ?? 03 4d ?? 0f be 19 8b 55 ?? 52 e8 ?? ?? ?? ?? 83 c4 ?? 8b c8 8b 45 ?? 33 d2 f7 f1 8b 45
   condition:
```

```
        uint16(0)==0x5A4D and $xor
}
```

Figure 14. YARA rule identifying the XOR routine implemented by Mars Stealer

To conclude this section, we would like to mention another rather classical but very efficient method to collect and classify the malware samples based on the PE creation time. Indeed, many unpacked Mars Stealer samples share identical PE creation dates, making them easy to identify on sample sharing platforms: "*2021-08-12T17:45:33*" and "*2022-01-05T14:09:08*".

The Command & Control infrastructures of cyber attackers observed in 2021 by SEKOIA.IO

Tracking Mars Stealer C2 servers

Tracking servers used to host malware C2 servers or more widely adversary infrastructures is a proactive hunting approach we have intensively developed at SEKOIA.IO[5]. Concerning the Mars Stealer malware, C2 servers are hosted by the attackers and not by the malware developer. This makes it more difficult or even impossible to identify a heuristic based on the HTTP response to find the malware C2 servers, as each attacker should configure its own HTTP server hosting the Mars Stealer administration panel.

However, a good and simple method to track the widely sold malware consists in finding servers hosting one of the characteristic web pages of the administration panel. By searching the hash of these specific web pages (JavaScript, PHP or HTML pages) on URL scanning platforms, we can identify the servers used by the malware to download payloads and exfiltrate the stolen information.

Concerning the Mars Stealer administration panel, we can track several pages which are specific to the malware panel. For example:

- *d8f09307b60c5bef5ceacfd8501bd3d91f1de9e5e746bb2d7def94d86789da50* and *304288329069ad8eaafce0f10a369101607c9248fbc9aaaa733c9e2dab5c467f* are specific to the Mars Stealer PHP login pages (*login.php*). The first hash corresponds to the login page of the version 8 of Mars Stealer, while the second corresponds to the login page of the version 7 and below. We were able to confirm these results from the leaked source code of the administration panel.
- *20e6bb3cf9d13f10bca7b7b5d1f4cb82146c274747e8c2ae7fe3307881f00829* is specific to a CSS page used by the Mars Stealer login page (*bootstrap.min.css*).

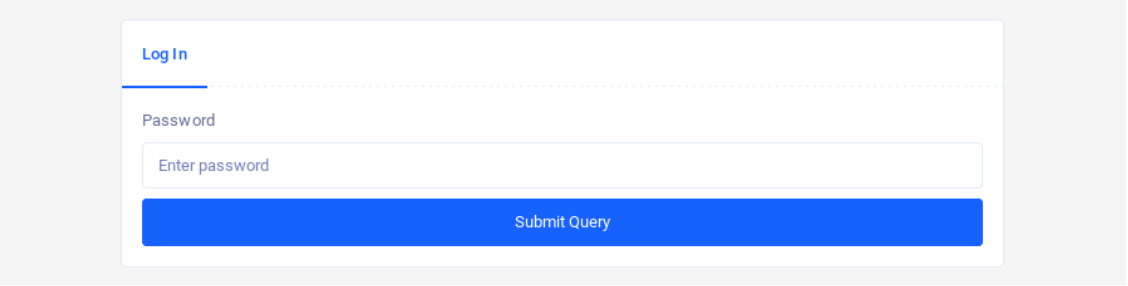


Figure 15. Login page of the Mars Stealer administration panel

From this information, we can write a heuristic on urlscan.io[6] (or other similar services) like:

```
hash:(20e6bb3cf9d13f10bca7b7b5d1f4cb82146c274747e8c2ae7fe3307881f00829 OR 304288329069ad8eaafce0f10a
```

On April 7, 2022, this query resulted in 43 URLs which hosted a Mars Stealer administration panel.

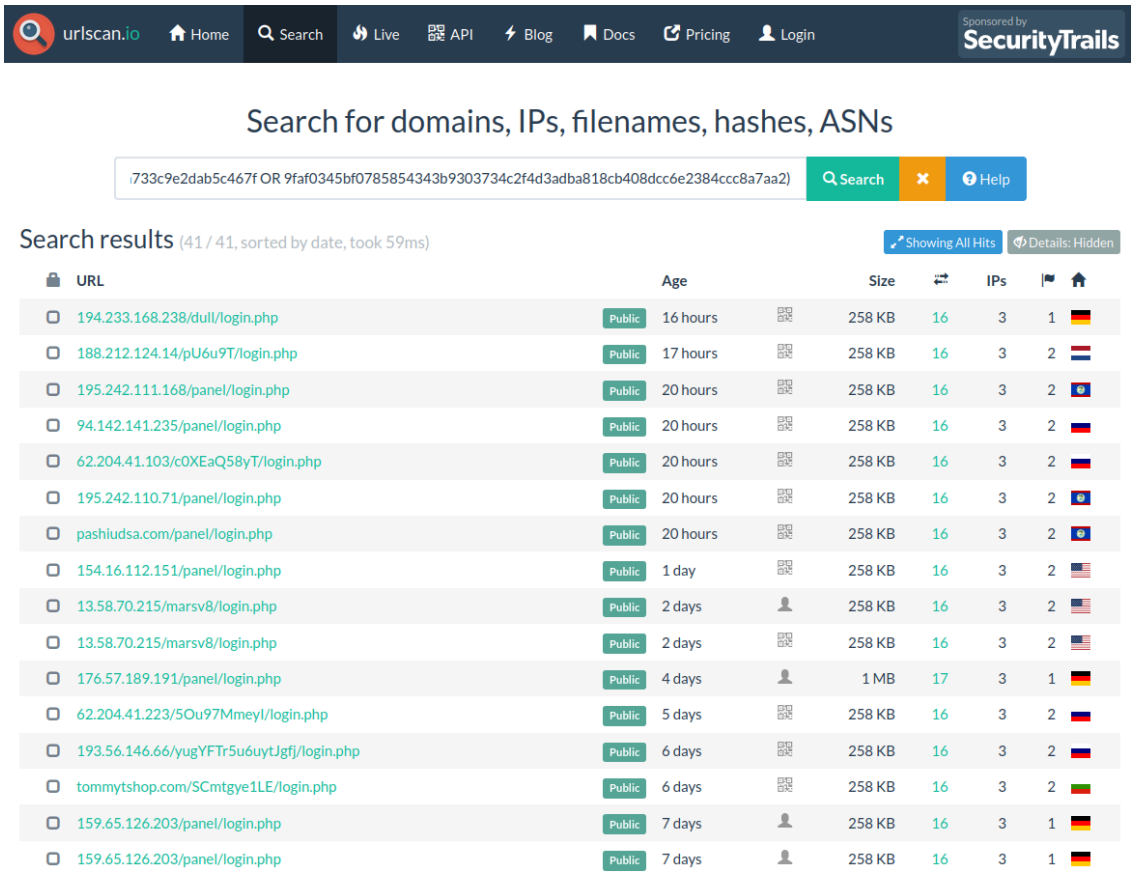


Figure 16. Urlscan.io results on the heuristic identifying Mars Stealer login pages

This method is based on URL submissions and, unfortunately, is not as proactive as heuristics based on the HTTP response. But it has the merit of collecting network IoCs (domain names or IP addresses) over time. Moreover, the webpages of the malware administration panel rarely changes, unlike the source code of the agent. It is therefore a good way to track a malware family over time and then pivot to new samples that might not be detected by our YARA rules.

Another way of tracking Mars Stealer C2 servers is to pivot on the file *request* (*3de1fb0d1108907fd61d6d6b9a4c6b856af509e0af35578f158cfce5d634fe07*) which is the archive containing the legitimate DLLs. All Mars Stealer samples of version 8 request this resource. For example, on VirusTotal, some C2 servers can be identified hosting the zip file and samples requesting it.

In conclusion, we are tracking the Mars Stealer threat by different means which allows us to collect network and system IoCs. Another technique we used to collect Mars Stealer C2 URLs consists of extracting them from the samples. The next part is focused on our in-depth analysis of several Mars Stealer versions, which allowed us to implement a configuration extractor.

Mars Stealer objective C2

An in-depth analysis of this malware has been done by 3xp0rt, the article describes how the strings are loaded, the checks performed against the infected hosts (languages, anti-emulation, *e.g.*: *John Doe* & *HALT9*), the cookies and crypto wallets extraction, the C2 exfiltration and configuration grabbing. In our context, the analysis here is focused on the Command and Control configuration moreover, on how to automatically extract it. To answer this need, an analysis of the different versions of the malware and on how the C2 URL is deobfuscated and loaded in PE is required. From this analysis, we tried to identify the Mars Stealer releases based on the *MarsTeam* publications on XSS forum.

Mars Stealer main functionalities

The figure below briefly introduces what Mars Stealer core function looks like (for more details checks 3xp0rt in-depth analysis[1]), its main function could be split in 8 units described below:

1. Load static string: this function prepares the malware to load further libraries and functions and setup the decryption key;
2. Link basic function to setup malware functionalities for step 3 to 4 includes;
3. Anti-debugging / Anti-emulation:
    1. Check time
    2. Check Windows Defender sandbox
4. Decrypt all strings: this includes other libraries' function name, strings used by the stealer (e.g. SQL requests for cookies extraction);
5. Load all required functions and libraries (*sqlite3.dll*, *freebl3.dll*, *mozglue.dll*, etc…);
6. Steal cookies, crypto wallet, password, *etc.*…;
7. Exfiltration stolen data over HTTP to its C2;
8. Removed itself via a *ShellExecuteExA* (`C:\Windows\System32\cmd.exe /c timeout /t 5 & del /f /q "%s" & exit`).

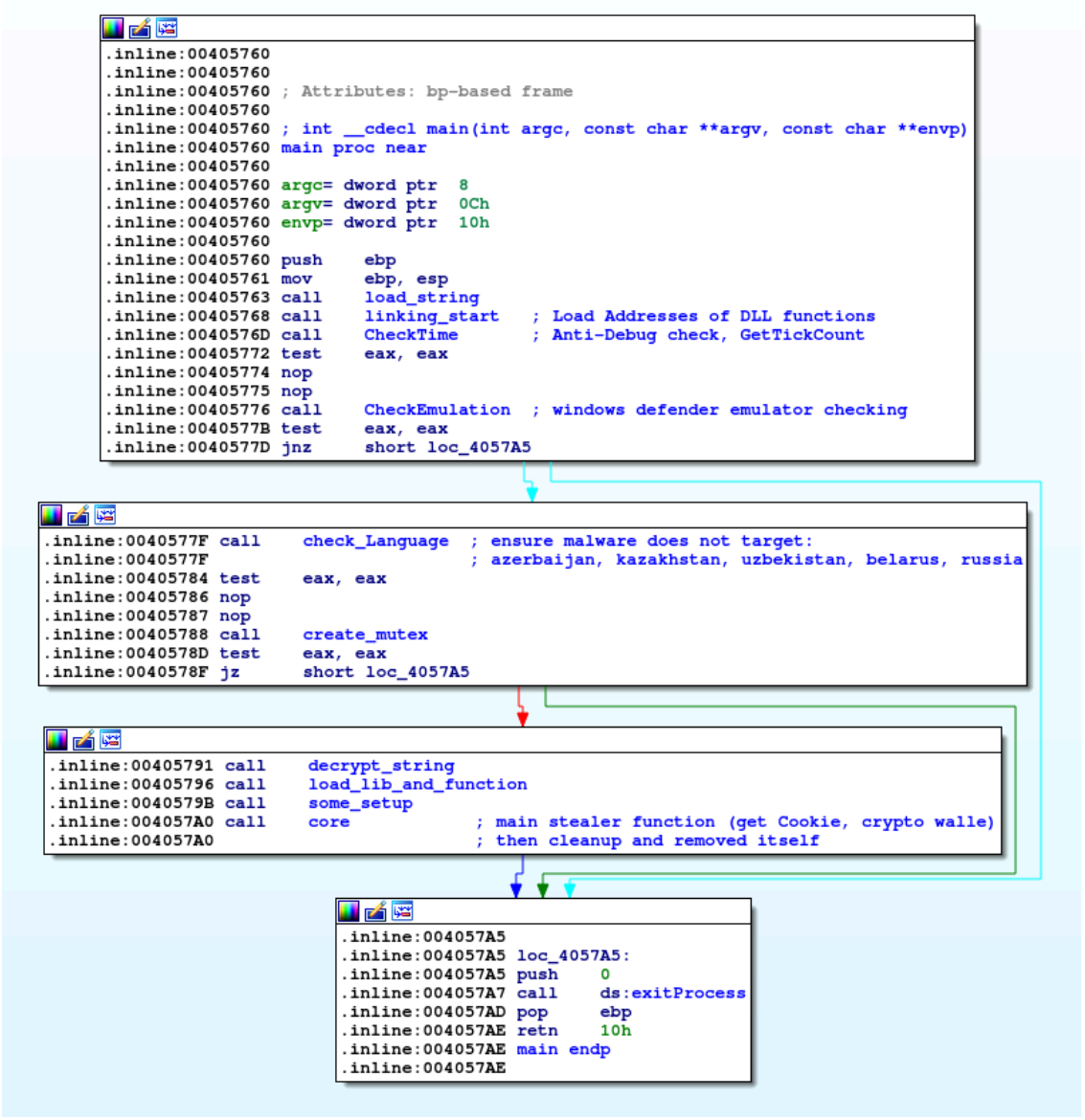


Figure 17. Mars Stealer core function

SIGMA, design and MITRE ATT&CK… new features of the XDR and CTI platform

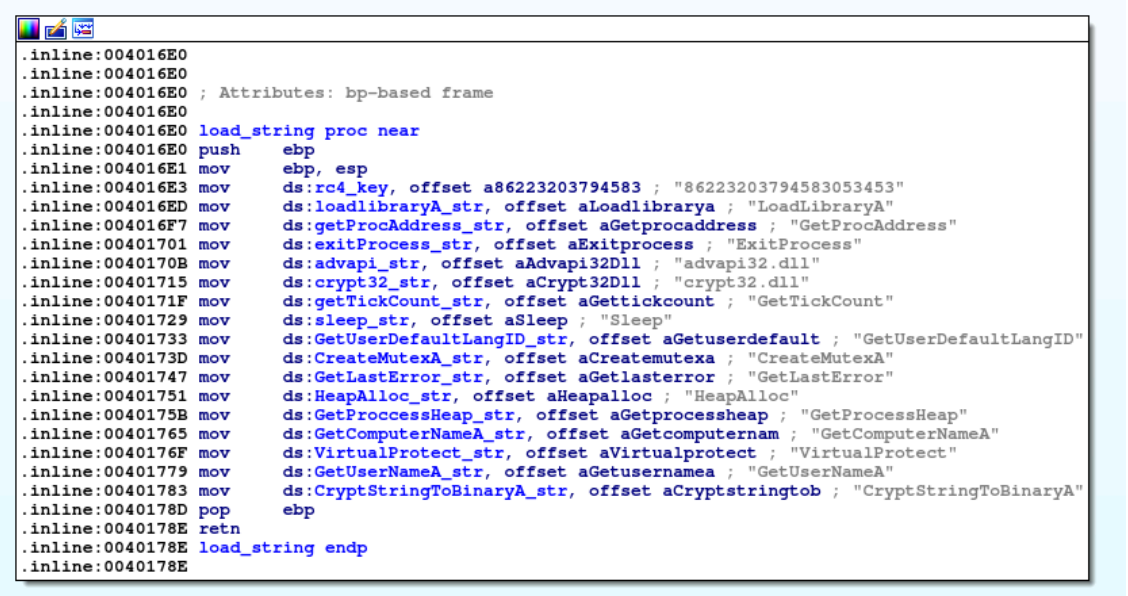
This article focuses on differences observed through the different Mars Stealer samples we have collected. As mentioned previously in the analysis of publications on underground forums, *MarsTeam* tries to improve its product with each release (bug fix, some obfuscation & new functionalities). Release advertisements on the XSS forum might not match what has been identified as a version in this article. The version identification process is based on how the malware stores and deobfuscates its C2, which could lead to version overlap of the announced versions by *MarsTeam* and our work.

The next section describes the first and older technique used by Mars Stealer to load its C2, which involved retrieving the RC4 key and spotting the position of the Command and Control in the PE.

String loading

One of the first functions of the malware aims to load a small set of strings, which are used a bit later with the two known functions of *Kernel32.dll*: *LoadLibrary* and *GetProcAddress*. These two functions are used for further functions and library loading. In this same function (*load_string*), the first string loaded is twenty bytes long, which appears to be the key used to decrypt (RC4) strings of the malware.

Those obfuscated strings are stored in the *.data* section in this given format: encrypted string with RC4 (Rivest Cipher 4)[7] algorithm stored in base64. Figure below shows how the RC4 key is loaded.



```
.inline:004016E0
.inline:004016E0
.inline:004016E0 ; Attributes: bp-based frame
.inline:004016E0
.inline:004016E0 load_string proc near
.inline:004016E0 push    ebp
.inline:004016E1 mov     ebp, esp
.inline:004016E3 mov     ds:rc4_key, offset a86223203794583 ; "86223203794583053453"
.inline:004016ED mov     ds:loadlibraryA_str, offset aLoadlibrarya ; "LoadLibraryA"
.inline:004016F7 mov     ds:getProcAddress_str, offset aGetprocaddress ; "GetProcAddress"
.inline:00401701 mov     ds:exitProcess_str, offset aExitprocess ; "ExitProcess"
.inline:0040170B mov     ds:advapi_str, offset aAdvapi32Dll ; "advapi32.dll"
.inline:00401715 mov     ds:crypt32_str, offset aCrypt32Dll ; "crypt32.dll"
.inline:0040171F mov     ds:getTickCount_str, offset aGettickcount ; "GetTickCount"
.inline:00401729 mov     ds:sleep_str, offset aSleep ; "Sleep"
.inline:00401733 mov     ds:GetUserDefaultLangID_str, offset aGetuserdefault ; "GetUserDefaultLangID"
.inline:0040173D mov     ds:CreateMutexA_str, offset aCreatemutexa ; "CreateMutexA"
.inline:00401747 mov     ds:GetLastError_str, offset aGetlasterror ; "GetLastError"
.inline:00401751 mov     ds:HeapAlloc_str, offset aHeapalloc ; "HeapAlloc"
.inline:0040175B mov     ds:GetProccessHeap_str, offset aGetprocessheap ; "GetProcessHeap"
.inline:00401765 mov     ds:GetComputerNameA_str, offset aGetcomputernam ; "GetComputerNameA"
.inline:0040176F mov     ds:VirtualProtect_str, offset aVirtualprotect ; "VirtualProtect"
.inline:00401779 mov     ds:GetUserNameA_str, offset aGetusernamea ; "GetUserNameA"
.inline:00401783 mov     ds:CryptStringToBinaryA_str, offset aCryptstringtob ; "CryptStringToBinaryA"
.inline:0040178D pop     ebp
.inline:0040178E retn
.inline:0040178E load_string endp
.inline:0040178E
```

Figure 18. RC4 key loading

For example, for the following string: "*K3vgIP3rMlysQNU=*":

1. The base64-decoded string is *2b7be020fdeb325cac40d5*.
2. The RC4-decrypted string with the key *85297062256884302049* is "*MachineGuid*".

RC4 key retrieving is the first step to get the C2 in clear text. Then, finding the offset of the string that contains the C2 is the objective. This strings are the third and the fourth string loaded and decrypted in the function *decrypt_string*.

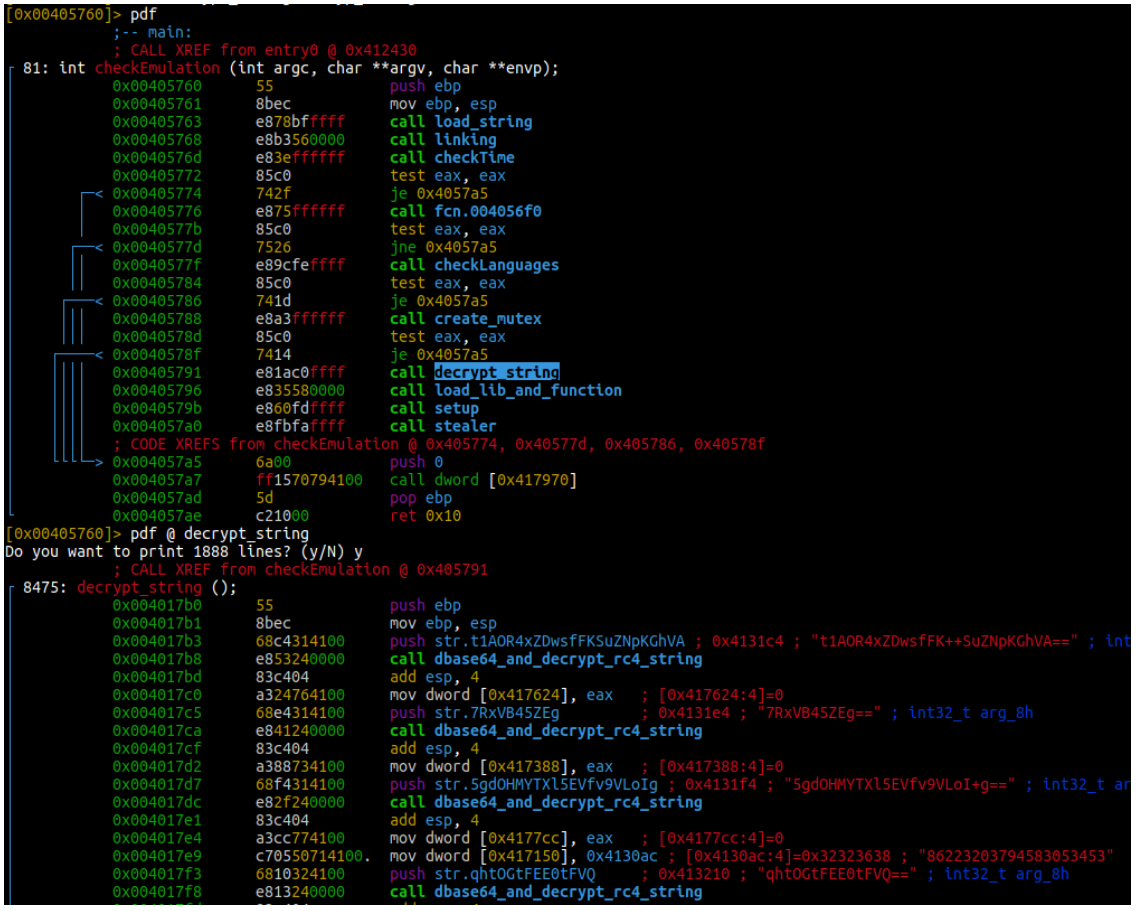


Figure 19. Offset of C2 encoded in base64 and encrypted with RC4 (addresses: 0x4131f4 and 0x413210)

In one of the most recent releases, a new section was created by the Mars Stealer authors. The C2 location is now at an offset defined in the LLCPPC code, *c.f.* figure 20. This version and its upgrade are analyzed in depth in the next three sections.

A new section entered the ring

As introduced in the section “Collecting Mars Stealer samples”, a new section with the singular name LLCPPC has been introduced in the Mars Stealer PE structure.

The structure of this section has evolved during the different releases. First of all, this section was composed of one segment that contains code used by the malware, and then authors add data, sometimes in clear text or obfuscated.

The new section of malware has the given structure:

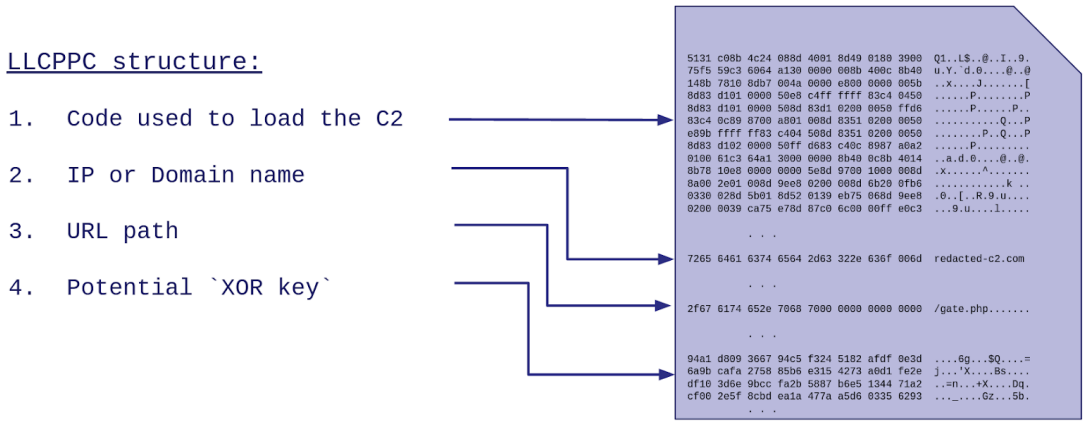


Figure 20. LLCPPC section structure

In the previous LLCPPC section structure, the C2 URL is in clear text, that means it is not obfuscated and the XOR key is not used. However, in most samples, the C2 URL is obfuscated with the XOR key.

Three different versions of this section have been observed during our investigation. The first version of this section is the one using RC4 encryption on the C2 located in the data section, then in next version malware developers append at the bottom of the section the C2 and change the obfuscation method. Data can be obfuscated with XOR operation or can be stored in clear text.

LLCPPC with RC4 encryption

The simplest version of the LLCPPC section only contains code. The code is in charge of loading the obfuscated string from a specific offset and calling the deobfuscation function (*e.g.* base64 decode plus RC4 decryption).

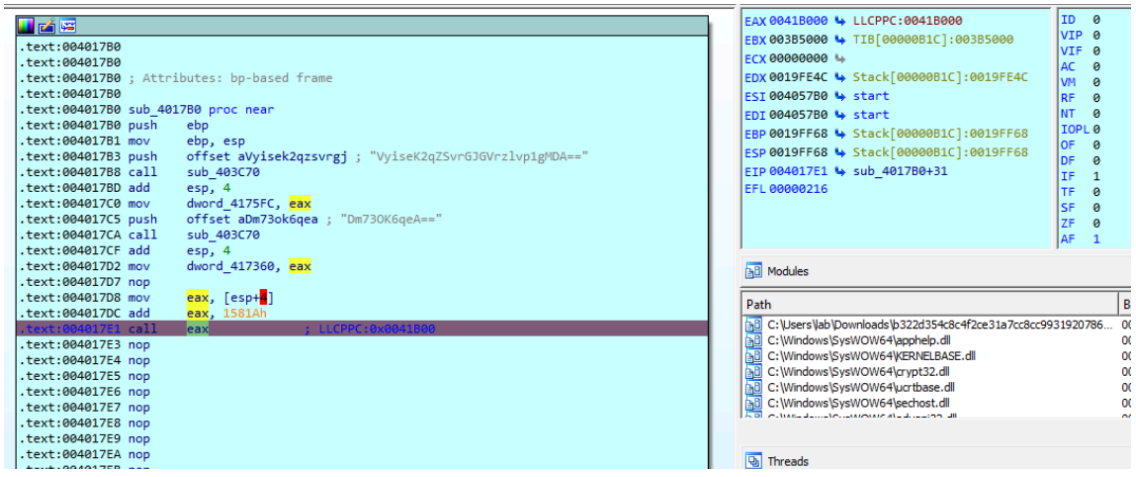


Figure 21. IDA decompilation of the function "decrypt_string" that calls a function in LLCPPC section to load the C2

Malware developers had replaced the C2 decryption (*see figure 21)* by a call to the function located at the beginning of the LLCPPC section, oddly followed by multiple NOP operations.

The calling function of the LLCPPC section is the following one:

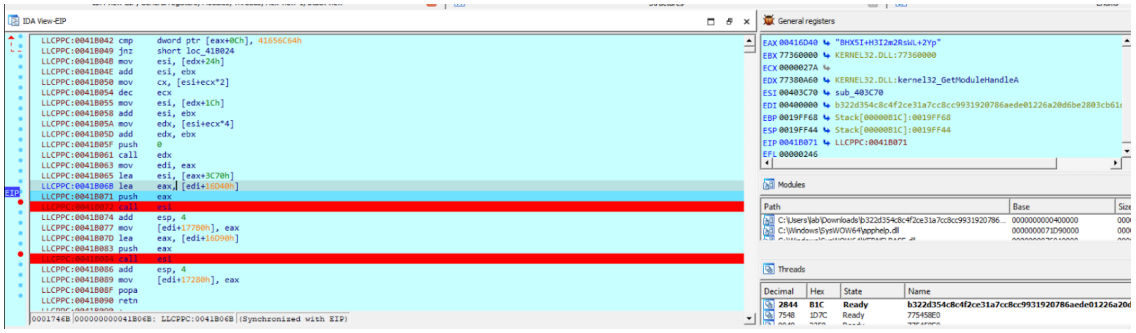


Figure 22. IDA decompilation of the C2 loading before its decryption (RC4)

1. The call *edx* is a call to the *kernel32.dll* function *GetModuleHandleA* used to return the handle to the file that is loaded, where the request module here is 0, this is a trick to return the base address of the PE.
2. From this address an offset is computed (different in each Mars Stealer sample): *lea eax, [edi + 0x16d40]*, *edi* which holds the base address of the PE.
3. Then, a call *esi* is performed, register *esi* contains the address of the function that decodes base64 and decrypts RC4. This function is located in the .text section and is used to decrypt other strings of the malware. This is the same function that is called multiple times by *decrypt_string*.
4. Finally, the decoded and decrypted string is then stored at a specific location with *mov dword [edi + 0x177b0]*, *eax*; the *edi* register still contains the base address of the PE.

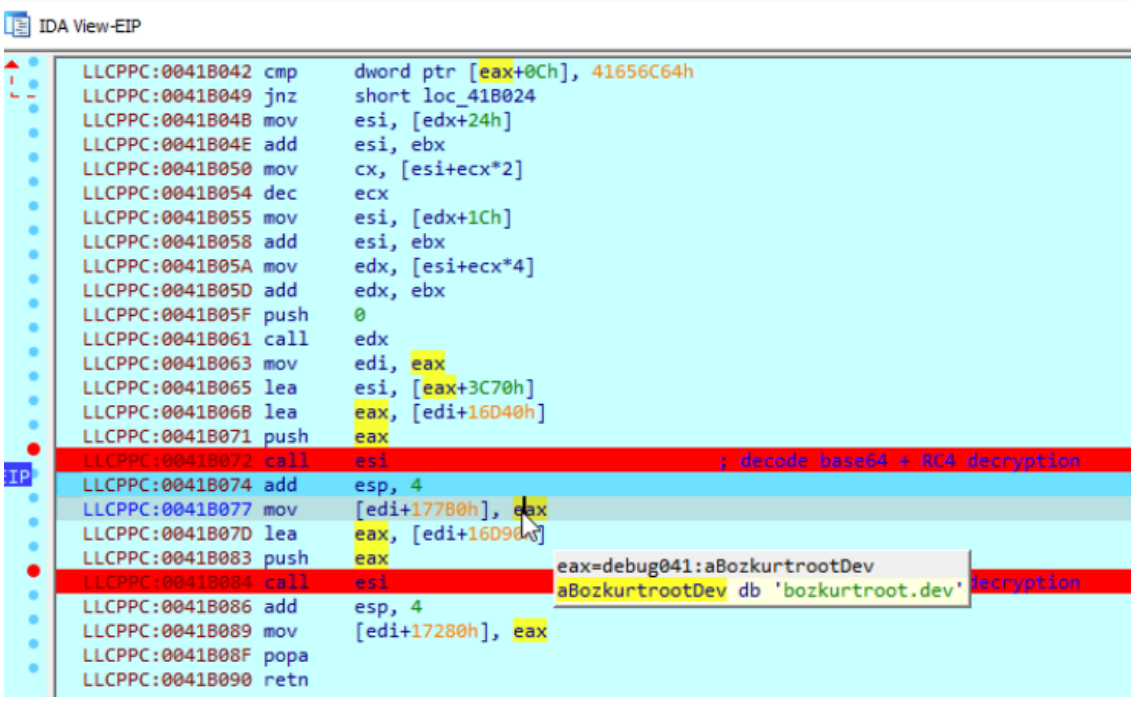


Figure 23. IDA decompilation of the C2 bozkurtoot[.]dev loading after its decryption (RC4)

The deobfuscation process is called twice, once for the C2 IP address or domain name and for the URL path as highlighted in the figure above.

LLCPPC with embedded data

As mentioned previously, some versions of this section embed data, those data appear to be the C2 (IP address or domain name) plus its URL path.

From now two variants exist, one with the C2 in clear text (*c.f.* figure 24) and a variant where the C2 is xored (*c.f.* figure 25):

```
[0x0042e200]> iS
[Sections]

nth paddr          size vaddr         vsize perm name
―――――――――――――――――――――――――――――――――――――――――――――――――
0   0x00000400  0x12e00 0x00401000  0x13000 -rwx .text
1   0x00013200   0x5600 0x00414000   0x6000 -r-- .rdata
2   0x00018800    0x200 0x0041a000  0x12000 -rw- .data
3   0x00018a00   0x2000 0x0042c000   0x2000 -r-- .reloc
4   0x0001aa00    0x400 0x0042e000   0x1000 -rwx LLCPPC

[0x0042e200]> px 300
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x0042e200  7061 6e65 6c69 6d65 726f 632e 636f 6d00  panelimeroc.com.
0x0042e210  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e220  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e230  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e240  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e250  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e260  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e270  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e280  2f67 6174 652e 7068 7000 0000 0000 0000  /gate.php.......
0x0042e290  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e2a0  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e2b0  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e2c0  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e2d0  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e2e0  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e2f0  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0042e300  94a1 d809 3667 94c5 f324 5182 afdf 0e3d  ....6g...$Q....=
0x0042e310  6a9b cafa 2758 85b6 e315 4273 a0d1 fe2e  j...'X....Bs....
0x0042e320  df10 3d6e 9bcc fa2b 5887 b6e5            ..=n...+X...
[0x0042e200]>
```

Figure 24. Radare2 dump of LLCPPC section with C2 in clear text

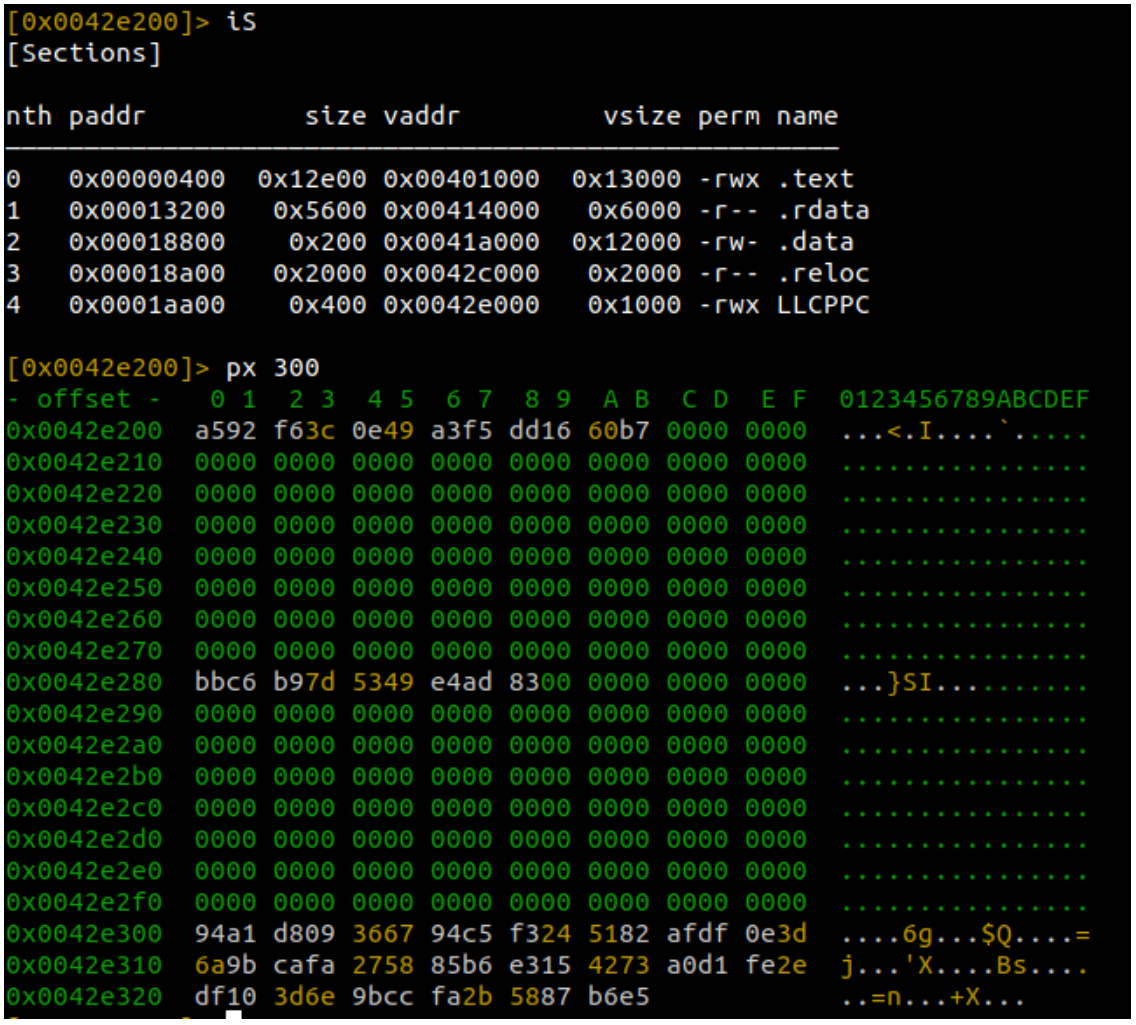


Figure 25. Radare2 dump of LLCPPC section with C2 obfuscated

A dynamic analysis helps to identify how the C2 was obfuscated and where its obfuscation key was stored.

The deobfuscation function (*c.f.* figure 27) determines the string length, then loads the C2 and xor key from a known offset located in the LLCPPC section to finally call the function that unxors data. Besides, this function is located in the *.text* section and is also used for other strings deobfuscation.

In fact, the obfuscated C2 could be read in clear text with the short following Python snippet of code:

```
def unxor(string: iterable, key: iterable) -> str:
   """Method to unxor obfuscated data from llcppc section"""

   unxored = ""

   for c1, c2 in zip(key, string):
           unxored += chr(c1 ^ c2)

   return unxored
```

Figure 26. Function deobfuscating data from LLCPPC section

The function responsible for the string unxor data is the following one:

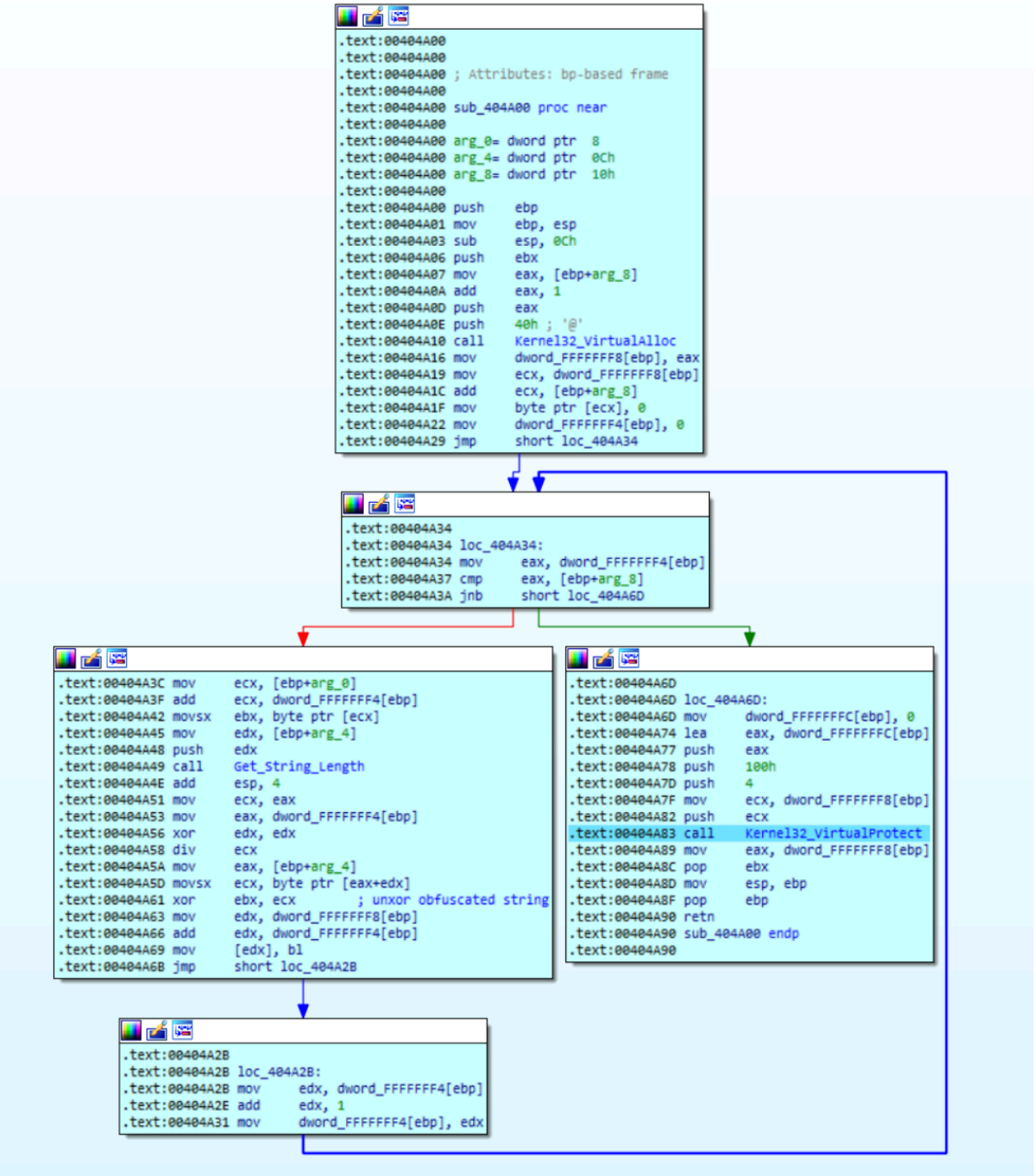


Figure 27. IDA decompilation of function responsible to unxor string

As observed while computing analysis on multiple samples using XOR obfuscation and data stored in the LLCPPC section (*c.f.* figure 20). The obfuscated C2 target, C2 URL path and XOR key are located at the same position for each sample of this version; this indicates a static assignment of the location by the authors.

- C2 target offset is: `LLCPPC base address + 0x200`.
- C2 target clear text is in the .data section: `0x41A800`.
- C2 URL path offset is: `LLCPPC base address + 0x280`.
- C2 URL path clear text is in .data section: `0x41A2A0`.
- XOR key for C2 target and C2 URL path is: `LLCPPC base address + 0x300`.

The mechanism described in the above section to get the C2 might be assigned to version 4 and above. A new variant then came with virtual machines and environment analysis detection.

LLCPPC Anti VM mechanism

The Mars Stealer team has updated its C2 loading; some anti-VM checks have been introduced in a new version. As shown by the figure below, the malware loops over the running process looking for particular process names. This version checks if a process contains *VBOX* or if a process is named *q.exe* which is a the Sysinternal tool: Rootkit detection utility[8]. In case none of the suspicious processes is identified, the malware allocated a virtual memory space for the Mars Stealer core function (*c.f.* figure 17) and then execute it (*c.f.* figure 28, "jmp eax").

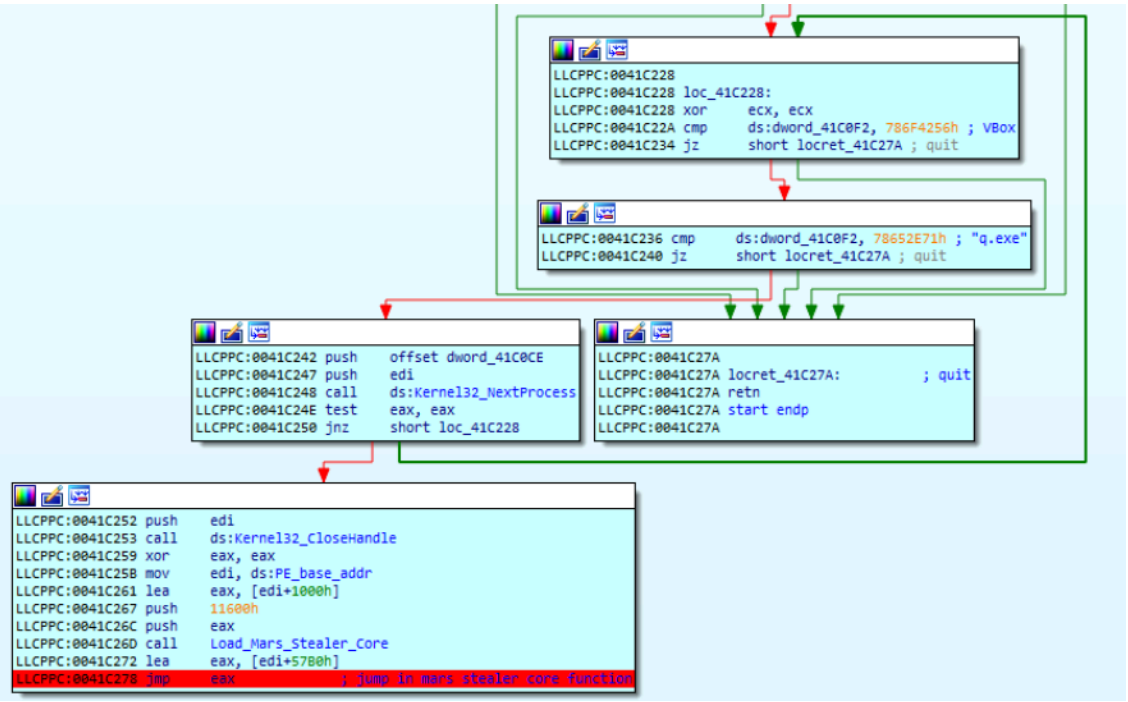


Figure 28. Anti-VM mechanism

The C2 loading method for this version does not differ that much from other LLCPPC versions. Indeed, after the virtual memory allocation of Mars Stealer core function followed by the jump into this new allocated memory area, the code of Mars Stealer remains the same as the one described in the "Mars Stealer capabilities" section. However, from a scripting point of view it will be required to look a step further to find out the "OEP" (Original Entry Point).

LLCPPC left the game: Mars Stealer V8

As explained previously in the section “Collecting Mars Stealer samples”, the last version has evolved, both the C2 server and the implant benefit from some upgrades. From the implant point of view, two major functionalities were introduced:

1. Downloaded DLLs are now grouped in a single ZIP file named and fetched from the “/request” URL.
2. A new anti-analysis check is introduced.

The C2 is obfuscated using the same techniques as described in the section “LLCPPC with embedded data”, but for this version, obfuscated data are located in the *.rdata* section. Moreover, each part of the C2, the IP address or the domain name and the URL path, has their own XOR key. The structure of the code does not change regarding the previous version with XOR obfuscation; the xored string is followed in the PE structure by its XOR key.

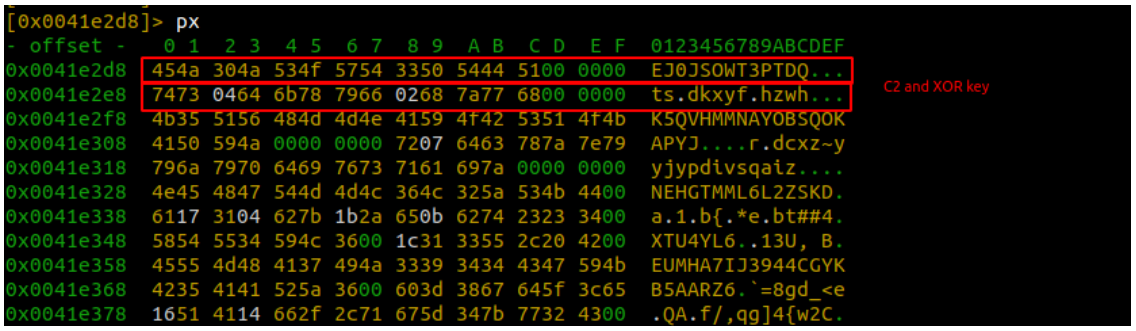


Figure 29. Obfuscated C2 IP address followed by its XOR key

This simple version 8 of Mars Stealer does not represent the trend we observed during our tracking session, various samples identified as version 8 are packed with Themida[9].

As mentioned in the past section, the samples retrieved from VirusTotal were identified with only two different RC4 keys: *8529706225688430204* and *86223203794583053453* with massive usage of the first one. We cannot affirm that one key is linked to a specific release, nor a selling batch, nor a threat actor (this hypothesis does not make sense here, in a malware-as-a-service context).

The XOR key located at the end of the LLCPPC section is repeatedly present even if no obfuscation is applied on the C2 (*c.f.* figure 24).

Some older (below version 7) Mars Stealer samples came “packed” with a VMProtect layer, related hashes are:

- *a6cd1f6158ce5a16bd500218333e81fcb6ecd960da3cfa0c1b701a5cf9f98dec*
- *8ded24590c991f33438fe38f3ae10e91672369b1f029bf339a94d74c8645932a*
- *af503eb7e314b4a8acb2ef849fc7cea7f273fa9544b40904314b651859b66a17*

SEKOIA.IO interaction

Malware C2 loading and deobfuscation have been analyzed in their different versions and the behavior of the multiple versions are now identified. A FAME module[10] to automatically extract Command and Control has been developed. This module uses the Python library r2pipe[11] to interact with the PE file.

Twice a week, one of our workers of the SEKOIA Malware Watcher project pulls samples from VirusTotal that match our YARA rule mentioned in the “Collecting Mars Stealer samples” section and submits these samples to our FAME instance. C2 are extracted via FAME and finally pushed contextualized to SEKOIA.IO with the relationship indicating the Mars Stealer malware.

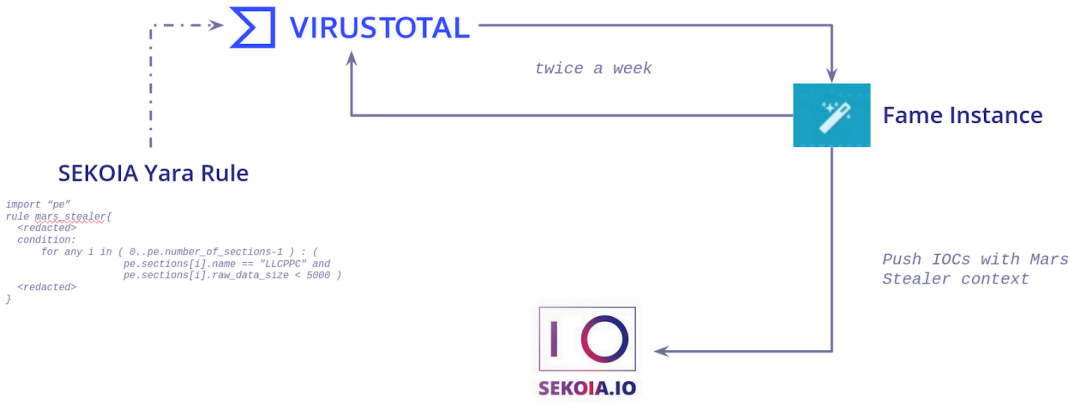


Figure 29. From VirusTotal to SEKOIA.IO with FAME

The FAME module will be release as Open Source on the SEKOIA.IO public Github repository named fame_module[12]. The extractor is available as standalone script[13].

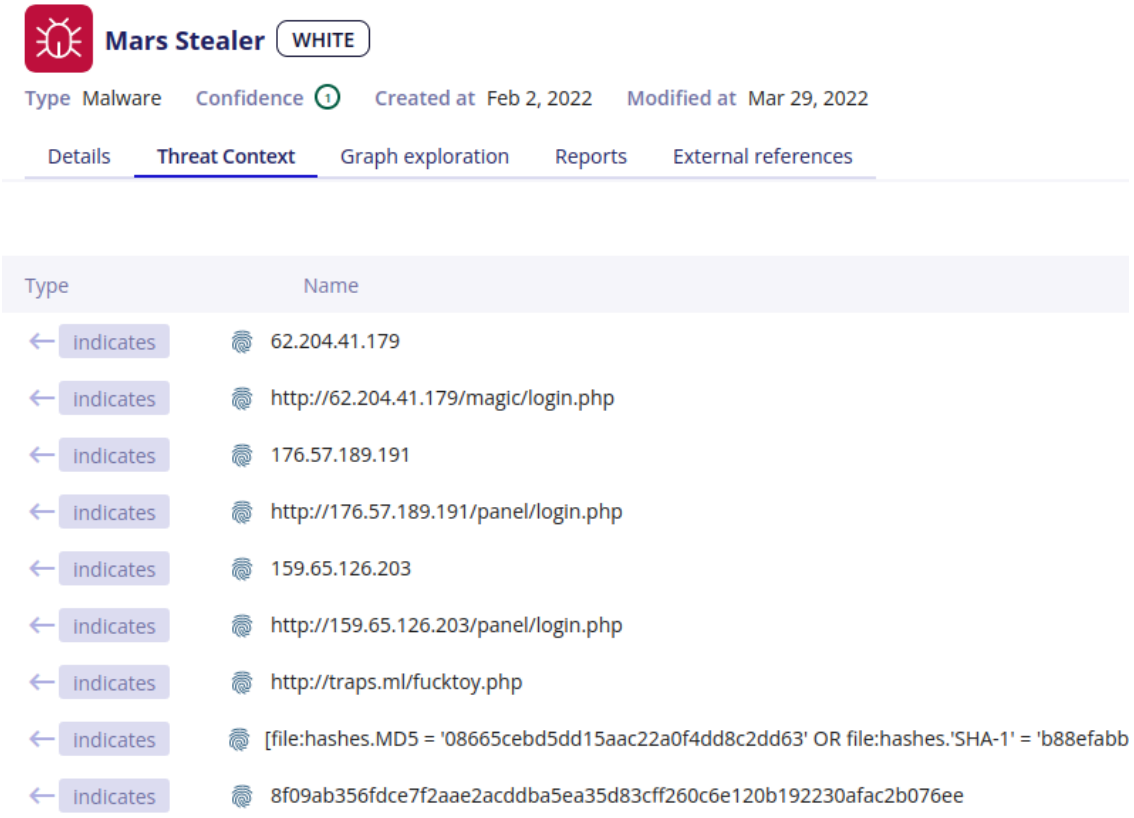


Figure 30. Pushed IoCs in SEKOIA.IO from SEKOIA Malware Watcher

Extracted IoCs can be found in the SEKOIA.IO Intelligence Center under the Mars Stealer malware object on the Threat Context tab and these IoCs are tagged with the source SEKOIA Malware Watcher.

Thanks for reading. You can also read our article on how to track and detect cobalt strike?

Resources

- IoCs: https://github.com/SEKOIA-IO/Community/blob/main/IOCs/marsstealer/mars_stealer_iocs_20220407.csv
- YARA rules:
  - infostealer_win_mars_stealer_early_version: https://github.com/SEKOIA-IO/Community/blob/main/IOCs/marsstealer/yara_rules/infostealer_marsstealer_early_version.yar
  - infostealer_win_mars_stealer_llcppc: https://github.com/SEKOIA-IO/Community/blob/main/IOCs/marsstealer/yara_rules/infostealer_marsstealer_llcppc.yar
  - infostealer_win_mars_stealer_xor_routine: https://github.com/SEKOIA-IO/Community/blob/main/IOCs/marsstealer/infostealer_marsstealer_xor_routine.yar
- Standalone extraction script: https://github.com/SEKOIA-IO/Community/blob/main/scripts/mars_stealer_c2_extractor.py

External references

[1] 3xp0rt analysis of Mars Stealer, February 1, 2022

[2] Exclusive Threat Research: Mars (Stealer) Attacks!, MorphisecLab, March 29, 2022

[3] Масове розповсюдження шкідливої програми MarsStealer серед громадян України та вітчизняних організацій, CERT UA, March 30, 2022

[4] Lapsus$: when kiddies play in the big league, SEKOIA TDR, March 23, 2022

[5] The Command & Control infrastructures of cyber attackers observed in 2021 by SEKOIA.IO, SEKOIA TDR, January, 2022

[6] URL Scan Mars Stealer Administration Panel Heuristic

[7] RC4: Rivest Cipher 4, Wikipedia

[8] Sysinternal tool: Rootkit detection utility, Microsoft

[9] Themida Overview, Oreans Technologies

[10] FAME Automates Malware Evaluation, CERT Société Générale & CERT SEKOIA

[11] Python library r2pipe, Radare2

[12] FAME module: Mars Stealer configuration extractor, SEKOIA TDR

[13] Standalone script: Mars Stealer configuration extractor, SEKOIA TDR

Thank you for reading this article. You can also read our article on:

Chat with our team!

Would you like to know more about our solutions?
Do you want to discover our XDR and CTI products?
Do you have a cybersecurity project in your organization?
Make an appointment and meet us!

Discover our:

- CTI platform
- XDR platform
- SOC platform
- Tools for SOC analyst
- SIEM solution

Share

CTI Malware Reverse Tracker

Share this post:

Source: https://blog.sekoia.io/mars-a-red-hot-information-stealer/