Raccoon Stealer: “Trash panda” abuses Telegram

decoded.avast.io/vladimirmartyanov/raccoon-stealer-trash-panda-abuses-telegram

March 9, 2022

by Vladimir MartyanovMarch 9, 20227 min read

We recently came across a stealer, called `Raccoon Stealer,` a name given to it by its author. Raccoon Stealer uses the `Telegram` infrastructure to store and update actual `C&C` addresses.

Raccoon Stealer is a password stealer capable of stealing not just passwords, but various types of data, including:

- Cookies, saved logins and forms data from browsers
- Login credentials from email clients and messengers
- Files from crypto wallets
- Data from browser plugins and extension
- Arbitrary files based on commands from C&C

In addition, it’s able to download and execute arbitrary files by command from its C&C. In combination with active development and promotion on underground forums, Raccoon Stealer is prevalent and dangerous.

The oldest samples of Raccoon Stealer we’ve seen have timestamps from the `end of April 2019` . Its authors have stated the same month as the start of selling the malware on underground forums. Since then, it has been updated many times. According to its authors, they fixed bugs, added features, and more.

Distribution

We’ve seen Raccoon distributed via downloaders: `Buer Loader` and `GCleaner` . According to some samples, we believe it is also being distributed in the form of `fake game cheats` , `patches for cracked software` (including hacks and mods for `Fortnite` , `Valorant` , and `NBA2K22`), or other software. Taking into account that Raccoon Stealer is for sale, it’s distribution techniques are limited only by the imagination of the end buyers. Some samples are spread unpacked, while some are protected using `Themida` or malware packers. Worth noting is that some samples were packed more than five times in a row with the same packer!

Technical details

Raccoon Stealer is written in `C/C++` and built using `Visual Studio` . Samples have a size of about `580-600 kB` . The code quality is below average, some strings are encrypted, some are not.

Once executed, Racoon Stealer starts checking for the default user locale set on the infected device and won’t work if it’s one of the following:

- Russian
- Ukrainian
- Belarusian
- Kazakh
- Kyrgyz
- Armenian
- Tajik
- Uzbek

C&C communications

The most interesting thing about this stealer is its communication with C&Cs. There are four values crucial for its C&C communication, which are hardcoded in every Raccoon Stealer sample:

- `MAIN_KEY` . This value has been changed four times during the year.
- `URLs` of Telegram gates with channel name. Gates are used not to implement a complicated Telegram protocol and not to store any credentials inside samples
- `BotID` – hexadecimal string, sent to the C&C every time
- `TELEGRAM_KEY` – a key to decrypt the C&C address obtained from Telegram Gate

Let's look at an example to see how it works:
`447c03cc63a420c07875132d35ef027adec98e7bd446cf4f7c9d45b6af40ea2b` unpacked to:
`f1cfcce14739887cc7c082d44316e955841e4559ba62415e1d2c9ed57d0c6232` :

1. First of all, `MAIN_KEY` is decrypted. See the decryption code in the image below:

```
.text:0042B84B                  mov     byte ptr [ebp-4], 3
.text:0042B84F                  mov     cl, 53h ; 'S'
.text:0042B851                  mov     dword ptr [ebp-2C6h], 9DF5C653h
.text:0042B85B                  mov     edx, ebx
.text:0042B85D                  mov     dword ptr [ebp-2C2h], 0D69FE2CDh
.text:0042B867                  mov     dword ptr [ebp-2BEh], 0C69EF6h
.text:0042B871
.text:0042B871 loc_42B871:                              ; CODE XREF: _main+4E7↓j
.text:0042B871                  not     cl
.text:0042B873                  xor     [ebp+edx-2C5h], cl
.text:0042B87A                  inc     edx
.text:0042B87B                  cmp     edx, 0Ah
.text:0042B87E                  jnb     short loc_42B888
.text:0042B880                  mov     cl, [ebp-2C6h]
.text:0042B886                  jmp     short loc_42B871
.text:0042B888 ; ---------------------------------------------------------------------
.text:0042B888
.text:0042B888 loc_42B888:                              ; CODE XREF: _main+4DF↑j
.text:0042B888                  mov     [ebp-2BBh], bl
```

In this example, the `MAIN_KEY` is `jY1aN3zZ2j` . This key is used to decrypt Telegram Gates `URLs` and `BotID` .

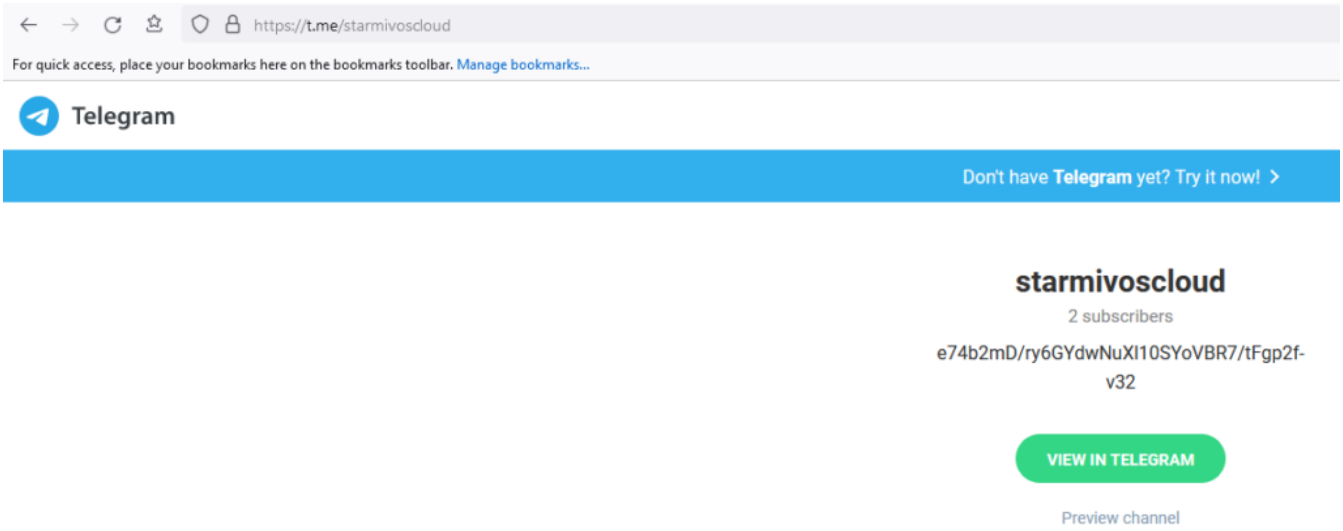
1. This example decodes and decrypts Telegram Gate URLs. It is stored in the sample as:
   `Rf66cjXWSDBo1vlrnxFnlmWs5Hi29V1kU8o8g8VtcKby7dXlgh1EIweq4Q9e3PZJl3bZKVJok2GgpA90j35LVd34QAiXtpeV2UZQS5VrcO7UWo0E1`
   After decoding Base64 it has this form:

```
00000000:  45 FE BA 72-35 D6 48 30-68 D6 F9 6B-9F 11 67 96  E■|r5╓H0h╓·kЯ◄g
00000010:  65 AC E4 78-B6 F5 5D 64-53 CA 3C 83-C5 6D 70 A6  eмфx╢ї]dS╩<Г┼mp¦
00000020:  F2 ED D5 E5-82 1D 44 23-07 AA E1 0F-5E DC F6 49  Єэ╒хВ↔D#•кс◘^▄ўI
00000030:  97 76 D9 29-52 68 93 61-A0 A4 0F 74-8F 7E 4B 55  Чv┘)RhУaадотП~KU
00000040:  DD F8 40 08-97 B6 97 95-D9 46 50 4B-95 6B 70 EE  ▌°@◘чЙ╢ЧХ┘FPKXkpю
00000050:  D4 5A 8D 04-D4 93 B3 C0-8D 19 AA B7-64 F6 3C C4  ╘ZH♦╘у│└Н↓к╖dў<─
00000060:  19 02 04 CD-DB D2 97 91-D6 4B 39 45-46 E2 23 06  ↓☻♦═▄╥ЧС╓K9EFт#♠
00000070:  63 8B 9A 01-FF 57 CE 0F-09 11 5E BD-CE 34 66 E4  cЛЪ☺ W╬☼○◄^╜╬4fф
00000080:  C0 65 4A BE-AB 52 72 C0-BF 86 FF 5C-73 5D 07 58  └eJ╛лRr└┐Ж \s]•X
00000090:  66 22 96 79-AF 0E 0E 06-8D FB 80 7C-71 3D C0 A9  f"Цуп♫♫♠НVA|q=└й
000000A0:  B1 E6 63 9F-BA 30 71 01-59 3A 93 B0-84 CF 03 12  ▒цсЯ║0q☺Y:У░Д╧♥↕
000000B0:  9D 3C CB B7-74 2F 67 C1-D0 45 DE 79-6F 99 7E 3F  Э<╦╖t/g┴╨E▌yoЩ~?
000000C0:  AB F8 12 3D-F9 58 17 B6-88 53 5C 05-85 9C 55 89  л°↕=·X↨╢ИS\♣ЕьUЙ
000000D0:  6A B6        -           -           -           j╢
```

Decrypting this binary data with `RC4` using `MAIN_KEY` gives us a string with Telegram Gates:

```
00000000:  68 74 74 70-3A 2F 2F 31-38 38 2E 31-36 36 2E 31  http://188.166.1
00000010:  2E 31 31 35-2F 73 74 61-72 6D 69 76-6F 73 63 6C  .115/starmivoscl
00000020:  6F 75 64 2C-68 74 74 70-3A 2F 2F 39-31 2E 32 31  oud,http://91.21
00000030:  39 2E 32 33-36 2E 31 33-39 2F 73 74-61 72 6D 69  9.236.139/starmi
00000040:  76 6F 73 63-6C 6F 75 64-2C 68 74 74-70 3A 2F 2F  voscloud,http://
00000050:  31 39 34 2E-31 38 30 2E-31 37 34 2E-31 34 37 2F  194.180.174.147/
00000060:  73 74 61 72-6D 69 76 6F-73 63 6C 6F-75 64 2C 68  starmivoscloud,h
00000070:  74 74 70 3A-2F 2F 31 38-35 2E 33 2E-39 35 2E 31  ttp://185.3.95.1
00000080:  35 33 2F 73-74 61 72 6D-69 76 6F 73-63 6C 6F 75  53/starmivosclou
00000090:  64 2C 68 74-74 70 3A 2F-2F 31 38 35-2E 31 36 33  d,http://185.163
000000A0:  2E 32 30 34-2E 32 32 2F-73 74 61 72-6D 69 76 6F  .204.22/starmivo
000000B0:  73 63 6C 6F-75 64 2C 68-74 74 70 73-3A 2F 2F 74  scloud,https://t
000000C0:  2E 6D 65 2F-73 74 61 72-6D 69 76 6F-73 63 6C 6F  .me/starmivosclo
000000D0:  75 64        -           -           -           ud
```

1. The stealer has to get it's real C&C. To do so, it requests a Telegram Gate, which returns an HTML-page:



Here you can see a Telegram channel name and its status in Base64: `e74b2mD/ry6GYdwNuXl10SYoVBR7/tFgp2f-v32`
The prefix (always five characters) and postfix (always six characters) are removed and it becomes `mD/ry6GYdwNuXl10SYoVBR7/tFgp` The Base64 is then decoded to obtain an encrypted C&C URL:

```
00000000:  98 3F EB CB-A1 98 77 03-6E 5E 5D 74-49 8A 15 05  ?ëËí?w♥n^]tIK§♣
00000010:  1E FF B4 58-29                                   ▲ ´X)
```

The `TELEGRAM_KEY` in this sample is a string `739b4887457d3ffa7b811ce0d03315ce` and the Raccoon uses it as a key to `RC4` algorithm to finally decrypt the C&C URL: `http://91.219.236[.]18/`

1. Raccoon makes a query string with PC information (machine GUID and user name), and `BotID`
2. Query string is encrypted with `RC4` using a `MAIN_KEY` and then encoded with Base64.
3. This data is sent using POST to the C&C, and the response is encoded with Base64 and encrypted with the `MAIN_KEY`. Actually, it's a JSON with a lot of parameters and it looks like this:

Thus, the Telegram infrastructure is used to store and update actual C&C addresses. It looks quite convenient and reliable until Telegram decides to take action.

Analysis

The people behind Raccoon Stealer

Based on our analysis of seller messages on underground forums, we can deduce some information about the people behind the malware. Raccoon Stealer was developed by a team, some (or maybe all) members of the team are Russian native speakers. Messages on the forum are written in Russian, and we assume they are from former USSR countries because they try to prevent the Stealer from targeting users in these countries.

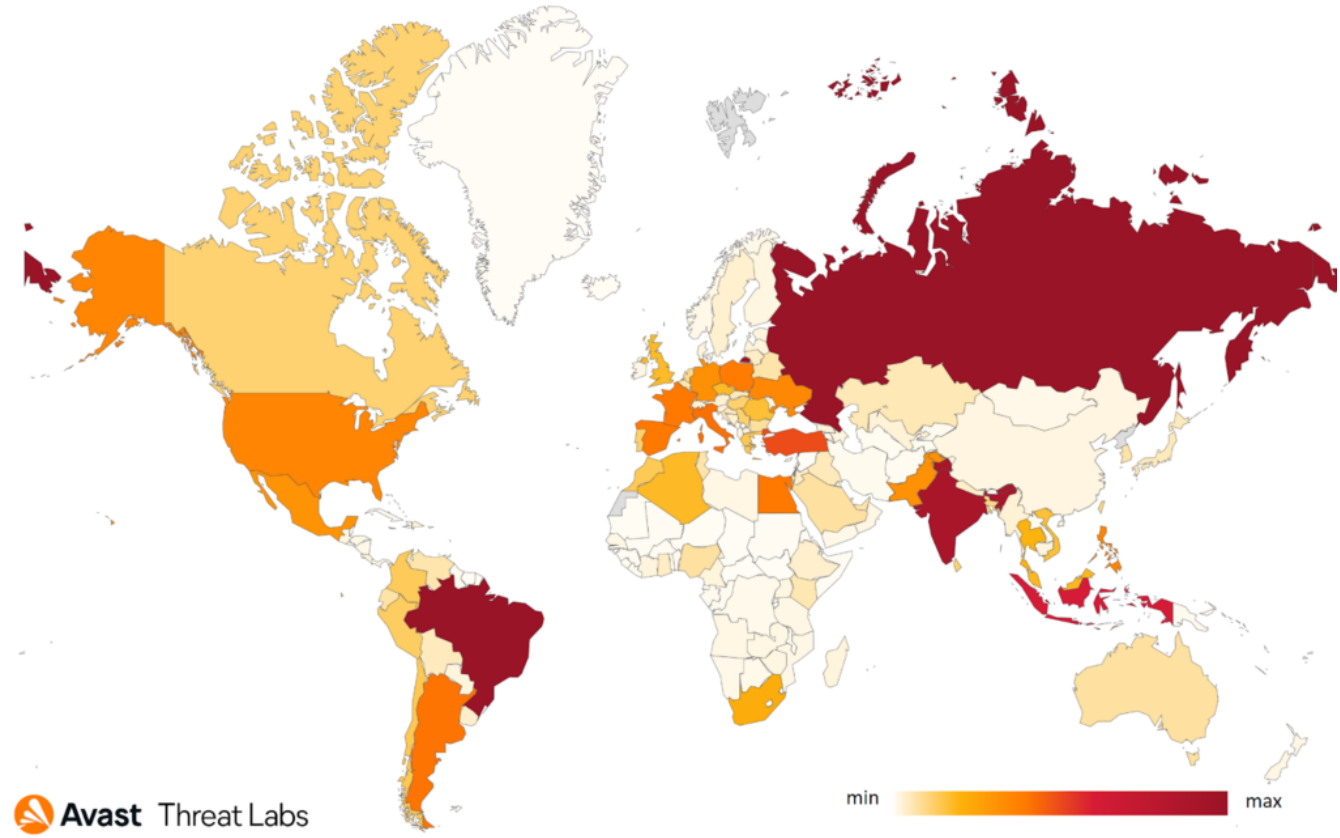
Possible names/nicknames of group members may be supposed based on the analysis of artifacts, found in samples:

- `C:\Users\a13xuiop1337\`
- `C:\Users\David\`

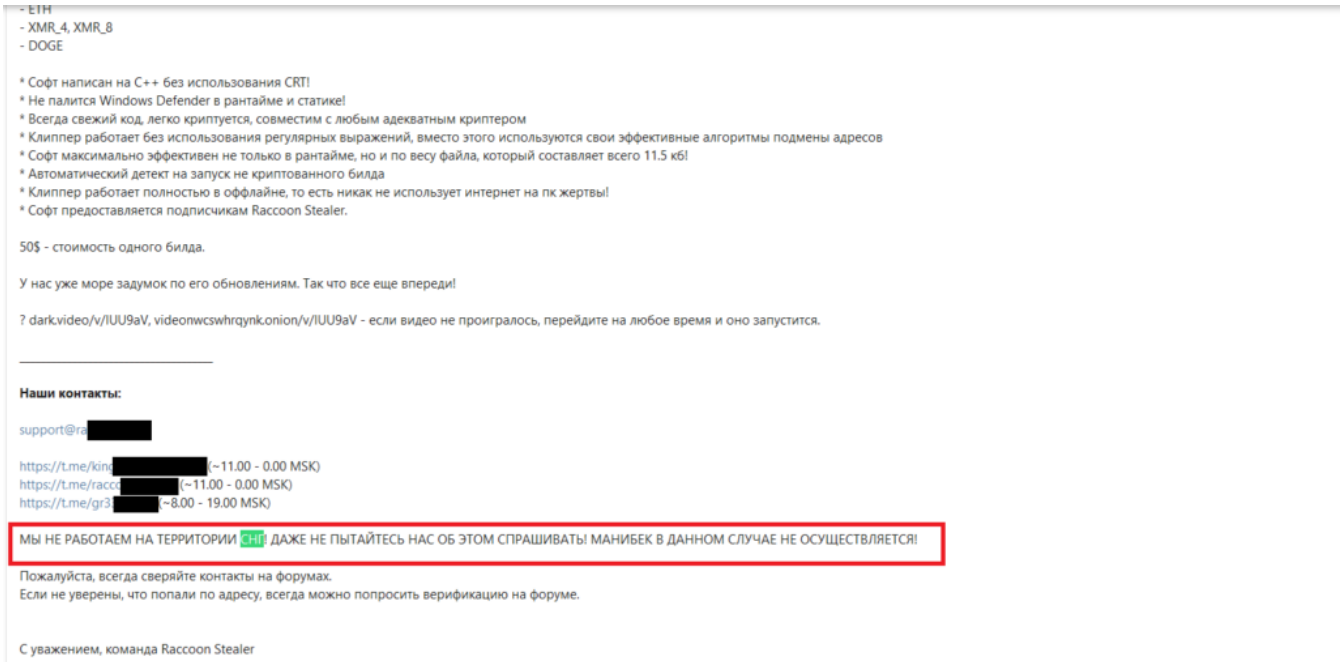
Prevalence

Raccoon Stealer is quite prevalent: from `March 3, 2021 - February 17, 2022` our systems detected `more than 25,000` Raccoon-related samples. We identified `more than 1,300` distinct configs during that period.

Here is a map, showing the number of systems Avast protected from Raccoon Stealer from `March 3, 2021 - February 17, 2022` . In this time frame, Avast protected nearly `600,000` Raccoon Stealer attacks.



The country where we have blocked the most attempts is Russia, which is interesting because the actors behind the malware don't want to infect computers in Russia or Central Asia. We believe the attacks spray and pray, distributing the malware around the world. It's not until it makes it onto a system that it begins checking for the default locale. If it is one of the language listed above, it won't run. This explains why we detected so many attack attempts in Russia, we block the malware before it can run, ie. before it can even get to the stage where it checks for the device's locale. If an unprotected device that comes across the malware with its locale set to English or any other language that is not on the exception list but is in Russia, it would still become infected.


- ETH
- XMR_4, XMR_8
- DOGE

* Софт написан на C++ без использования CRT!
* Не палится Windows Defender в рантайме и статике!
* Всегда свежий код, легко криптуется, совместим с любым адекватным криптером
* Клиппер работает без использования регулярных выражений, вместо этого используются свои эффективные алгоритмы подмены адресов
* Софт максимально эффективен не только в рантайме, но и по весу файла, который составляет всего 11.5 кб!
* Автоматический детект на запуск не криптованного билда
* Клиппер работает полностью в оффлайне, то есть никак не использует интернет на пк жертвы!
* Софт предоставляется подписчикам Raccoon Stealer.

50$ - стоимость одного билда.

У нас уже море задумок по его обновлениям. Так что все еще впереди!

? dark.video/v/IUU9aV, videonwcswhrqynk.onion/v/IUU9aV - если видео не проигралось, перейдите на любое время и оно запустится.

Наши контакты:

support@ra[redacted]

https://t.me/kin[redacted](~11.00 - 0.00 MSK)
https://t.me/racc[redacted](~11.00 - 0.00 MSK)
https://t.me/gr3[redacted](~8.00 - 19.00 MSK)

МЫ НЕ РАБОТАЕМ НА ТЕРРИТОРИИ СНГ! ДАЖЕ НЕ ПЫТАЙТЕСЬ НАС ОБ ЭТОМ СПРАШИВАТЬ! МАНИБЕК В ДАННОМ СЛУЧАЕ НЕ ОСУЩЕСТВЛЯЕТСЯ!

Пожалуйста, всегда сверяйте контакты на форумах.
Если не уверены, что попали по адресу, всегда можно попросить верификацию на форуме.

С уважением, команда Raccoon Stealer

Screenshot with claims about not working with CIS

Telegram Channels

From the more than 1,300 distinct configs we extracted, 429 of them are unique Telegram channels. Some of them were used only in a single config, others were used dozens of times. The most used channels were:

- `jdiamond13` – 122 times
- `jjbadb0y` – 44 times
- `nixsmasterbaks2` – 31 times
- `hellobyegain` – 25 times
- `h_smurf1kman_1` – 24 times

Thus, five of the most used channels were found in about `19%` of configs.

Malware distributed by Raccoon

As was previously mentioned, Raccoon Stealer is able to download and execute arbitrary files from a command from C&C. We managed to collect some of these files. We collected `185 files`, with a total size `265 Mb`, and some of the groups are:

- `Downloaders` – used to download and execute other files
- `Clipboard crypto stealers` – change crypto wallet addresses in the clipboard – very popular (more than 10%)
- `WhiteBlackCrypt Ransomware`

Servers used to download this software

We extracted unique links to other malware from Raccoon configs received from C&Cs, it was `196 unique URLs`. Some analysis results:

- `43%` of URLs have `HTTP` scheme, `57%` – `HTTPS`.
- `83 domain names` were used.
- About `20% of malware` were placed on `Discord CDN`
- About `10%` were served from `aun3xk17k[.]space`

Conclusion

We will continue to monitor Raccoon Stealer's activity, keeping an eye on new C&Cs, Telegram channels, and downloaded samples. We predict it may be used wider by other cybercrime groups. We assume the group behind Raccoon Stealer will further develop new features, including new software to steal data from, for example, as well as bypass protection this software has in place.

IoC

```
447c03cc63a420c07875132d35ef027adec98e7bd446cf4f7c9d45b6af40ea2b
f1cfcce14739887cc7c082d44316e955841e4559ba62415e1d2c9ed57d0c6232
```

Tagged asmalware, Malware Analysis, Racoon, reversing, stealer, telegram, trash panda