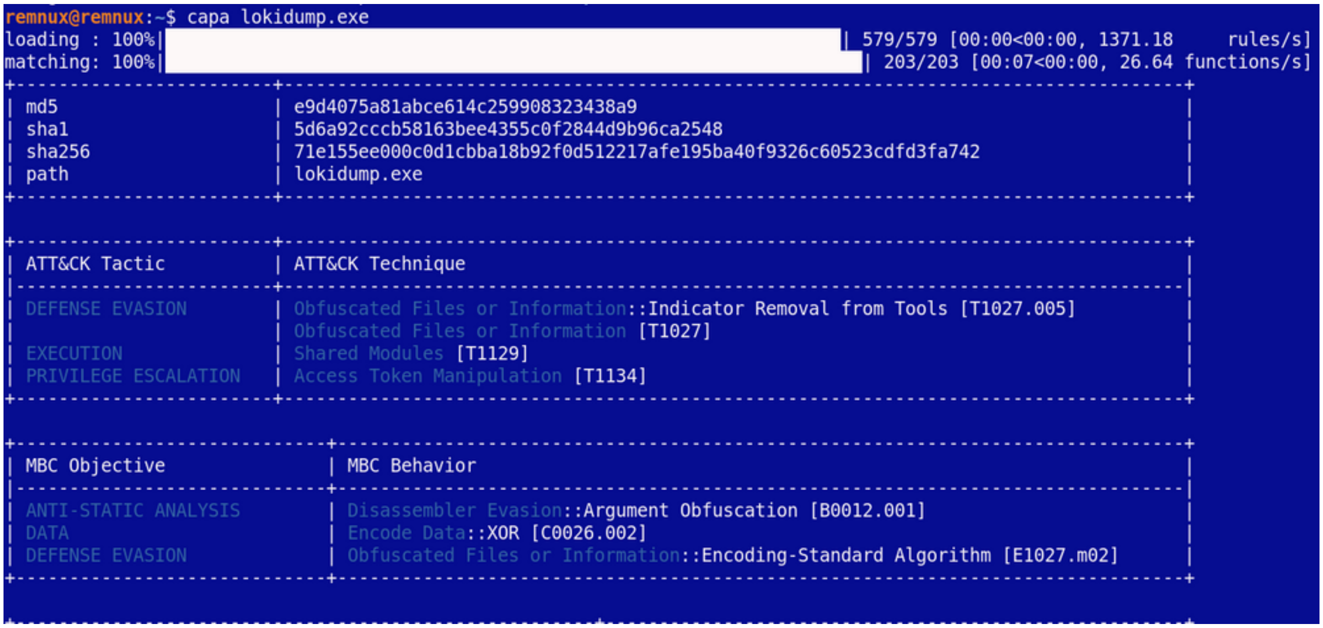
# Malware Headliners: LokiBot

atomicmatryoshka.com/post/malware-headliners-lokibot

z3r0day_504

January 28, 2022

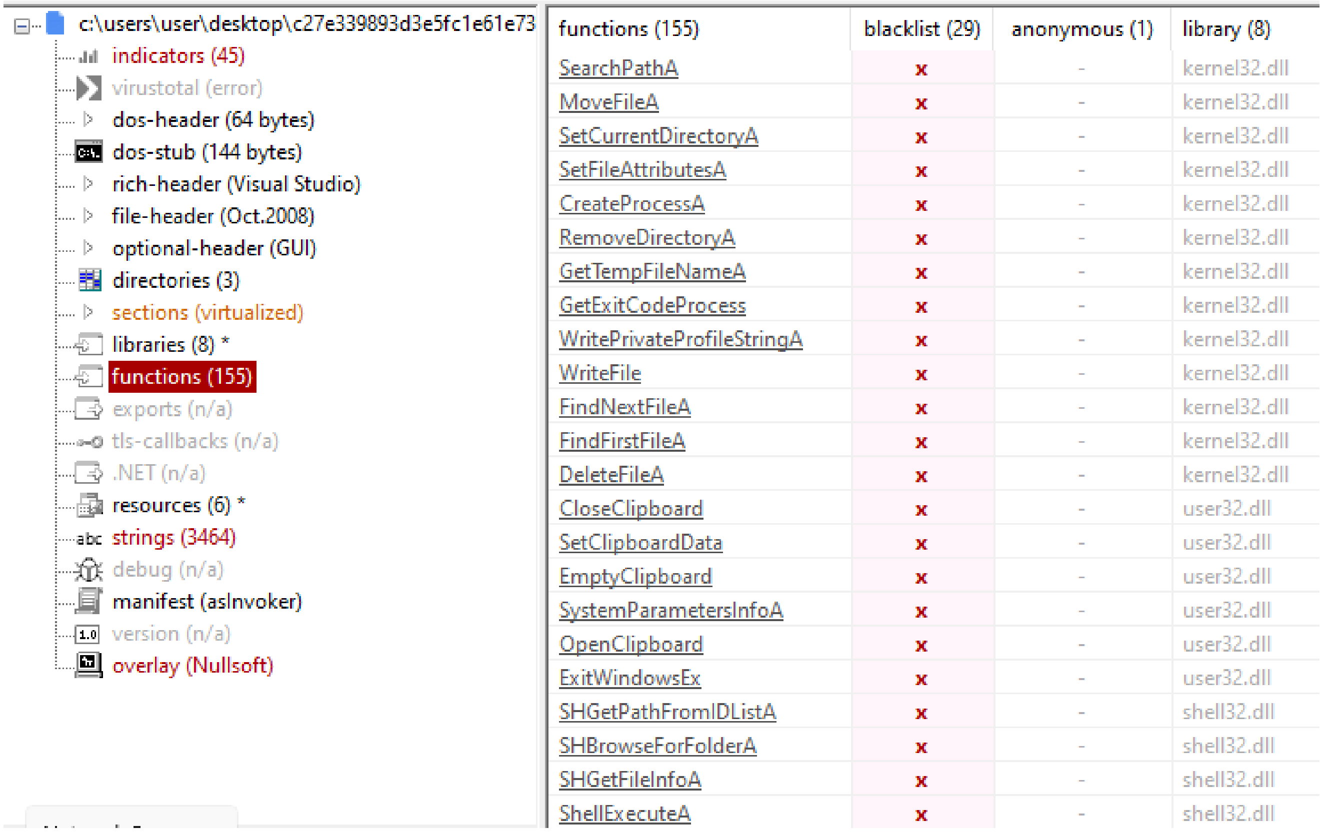



LokiBot, or Loki, is a password stealing malware and was considered the 8th most prevalent malware family in 2021 according to MalwareBazaar. Available for sale on underground forums, its within reach of anyone willing to pay the right price. As far as its use by organized actors, MITRE has linked its usage to the SilverTerrier threat group, known for having financial cybercrime motives.

In this blog post we'll conduct some static and dynamic analysis on a LokiBot sample to extract IOCs and characterize its behavior.

## STATIC ANALYSIS WITH PESTUDIO 9.27

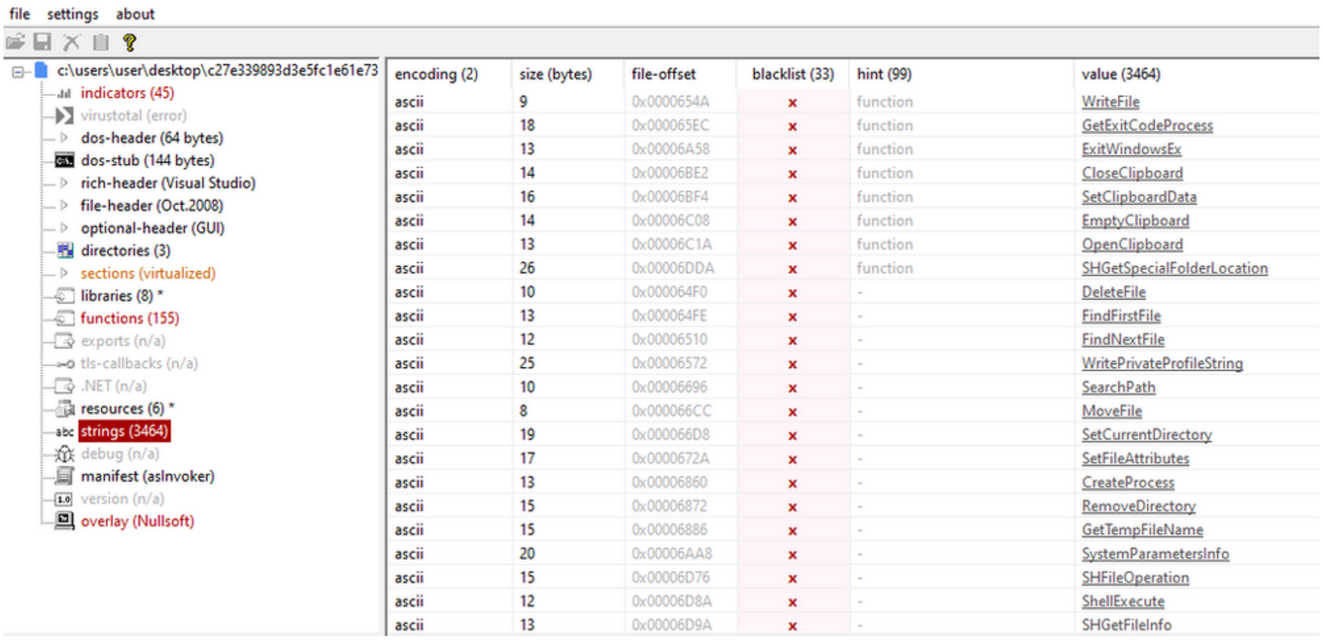
Using the latest version of PeStudio, we start to build a picture of what the specimen is capable of. Taking a look at the imports/functions category, we see the following:




Based on the imports, this sample shows potentially:

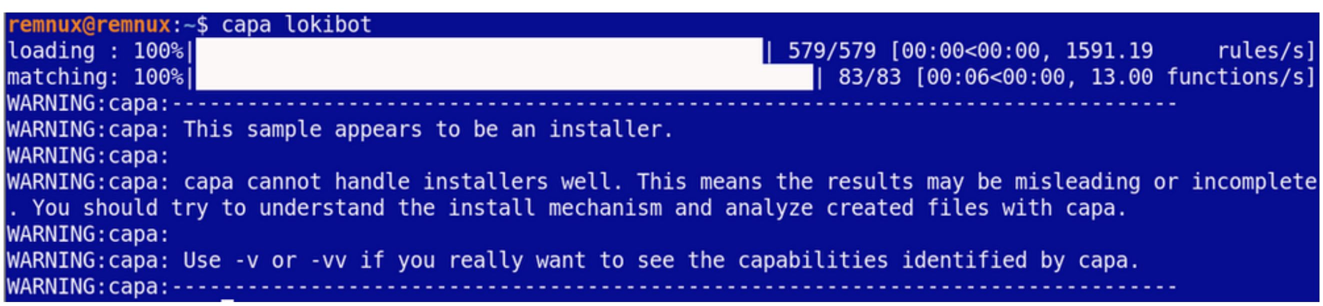
- Anti-debugging capabilities (EmptyClipboard, GetTickCount)
- Parsing through files and folders (FindFirstFileA, FindNextFileA, SearchPath, CreateFileA)
- Evasive behaviors/artifact destruction (DeleteFile, RemoveDirectory)
- File writing (CreateFileA, WriteFile, MoveFile)
- Registry interactions (RegCreateKey, RegDeleteKey, RegEnumKey, RegOpenKey, RegSetValue, RegQueryValue)

Looking at the strings tab, we see a lot of the same references to the API calls, especially if sorting for blacklist items to show first:




Outside of those, nothing is proving to be too conclusive here.

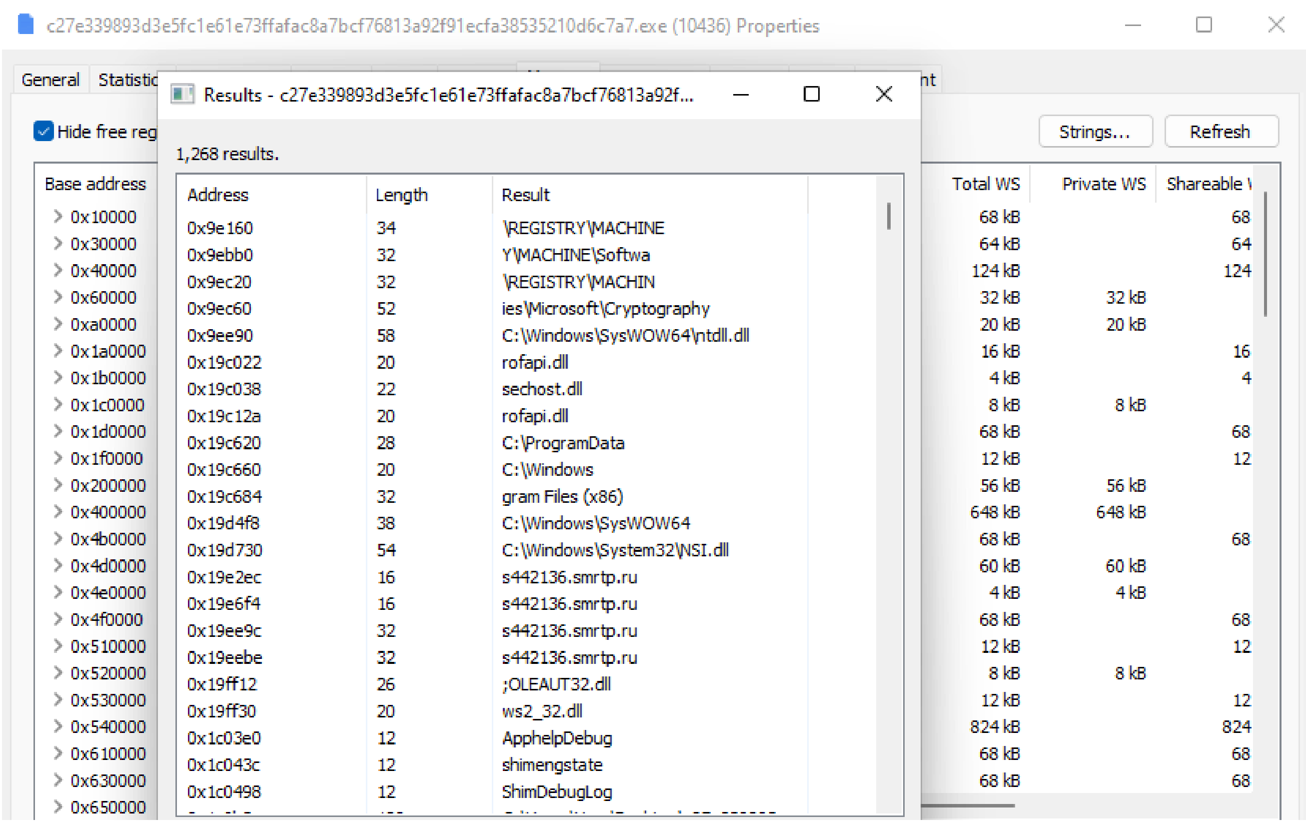
I pushed the file over to REMnux to give a stab at it with capa. Capa gave the following output:




In order to analyze further with capa, I'll need to dump the actual malware executable once it starts running during the dynamic analysis stage and rebuild it. I'll have those details after the dynamic analysis section.

## DYNAMIC ANALYSIS

Prior to detonating the sample, I had Process Hacker, Process Monitor, and WireShark running to capture any events. I was able to capture the following data:




The memory strings in Process Hacker offered some IOCs. We see a domain (s442136.smrtp[.]ru) as well as some registry interactions.

Process Monitor offered the following:

| | | | | | |
|---|---|---|---|---|---|
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Nichrome\User Data\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Nichrome\User Data\Default\Web Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalNichrome\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalNichrome\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\RockMelt\User Data\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\RockMelt\User Data\Default\Web Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalRockMelt\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalRockMelt\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Spark\User Data\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Spark\User Data\Default\Web Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalSpark\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalSpark\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Chromium\User Data\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Chromium\User Data\Default\Web Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalChromium\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalChromium\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Titan Browser\User Data\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Titan Browser\User Data\Default\Web Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalTitan Browser\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalTitan Browser\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Torch\User Data\Default\Login Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\Local\Torch\User Data\Default\Web Data | PATH NOT FOUND Desired Access: R... |
| 2:18:5... | c27e339893d3... | 10436 | CreateFile | C:\Users\User\AppData\LocalTorch\Login Data | PATH NOT FOUND Desired Access: R... |

We see a ton of "CreateFile" operations with browser file paths. It would be easy to be misled by the fact that the operation title is "CreateFile" and believe that the executable is attempting to generate files on the victim system. Reading Microsoft documentation offers some more context:

# CreateFileA function (fileapi.h)

Article • 10/13/2021 • 29 minutes to read

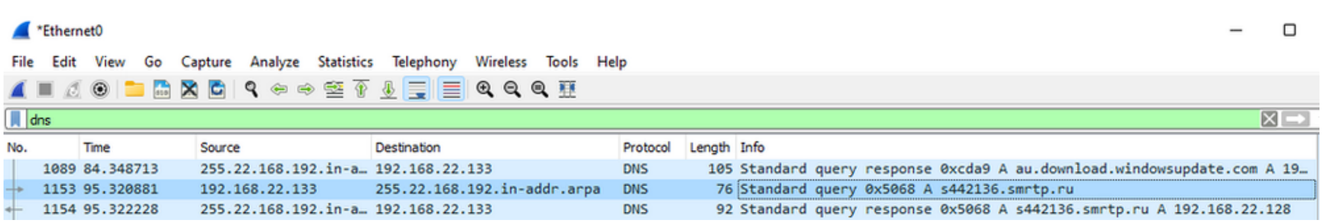
Is this page helpful?

Creates or opens a file or I/O device. The most commonly used I/O devices are as follows: file, file stream, directory, physical disk, volume, console buffer, tape drive, communications resource, mailslot, and pipe. The function returns a handle that can be used to access the file or device for various types of I/O depending on the file or device and the flags and attributes specified.

To perform this operation as a transacted operation, which results in a handle that can be used for transacted I/O, use the CreateFileTransacted function.

Not only does this function allow for the creation of files, but also opening them. On the right side of the Process Monitor screenshot we see the value "path not found," meaning that the malware tried to open or access the browser file paths and they did not exist. Being that the malware is a password stealer, it is likely checking these file paths for saved credentials.

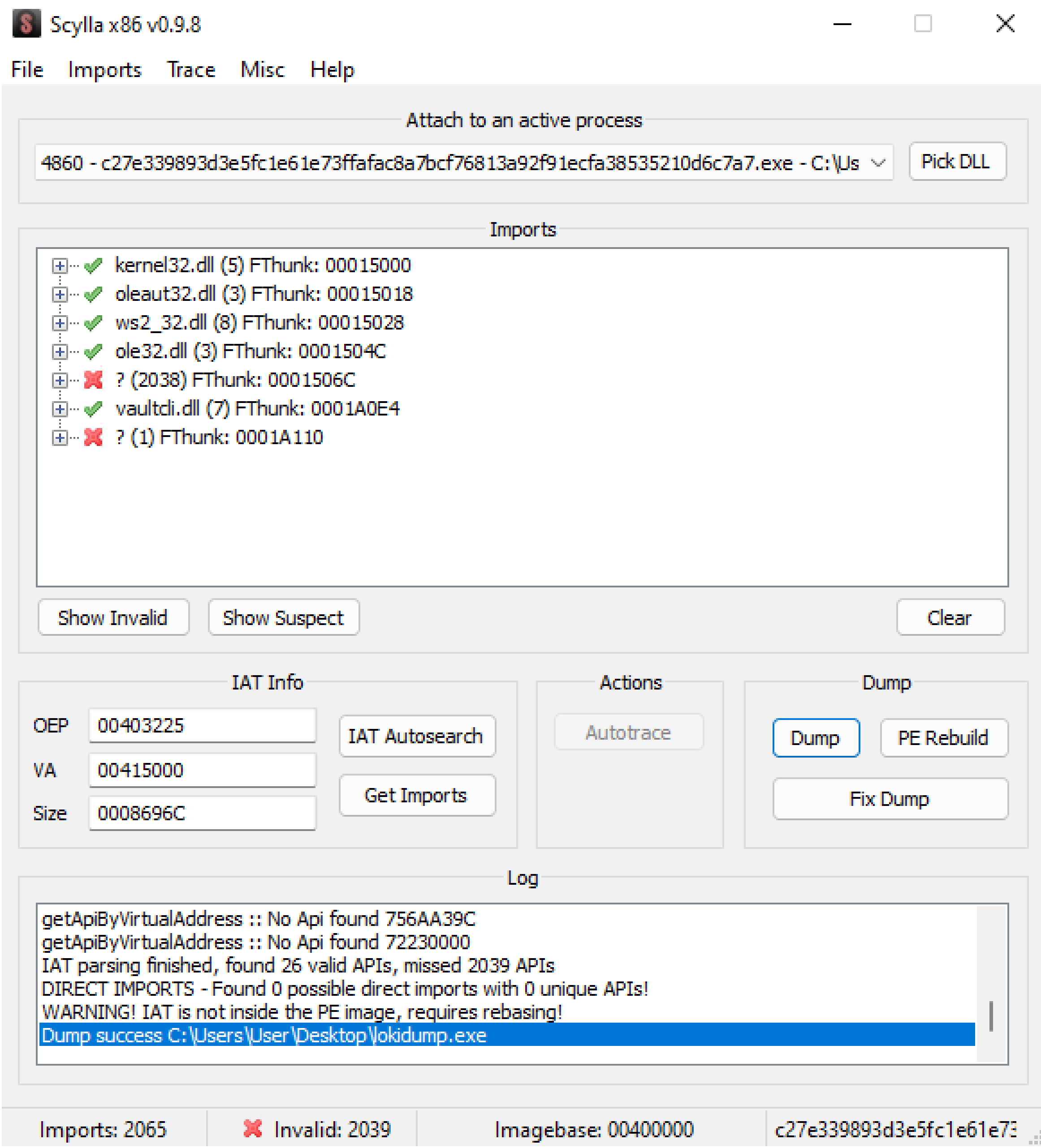
Seeing the WireShark output, we see information that corroborates earlier findings:



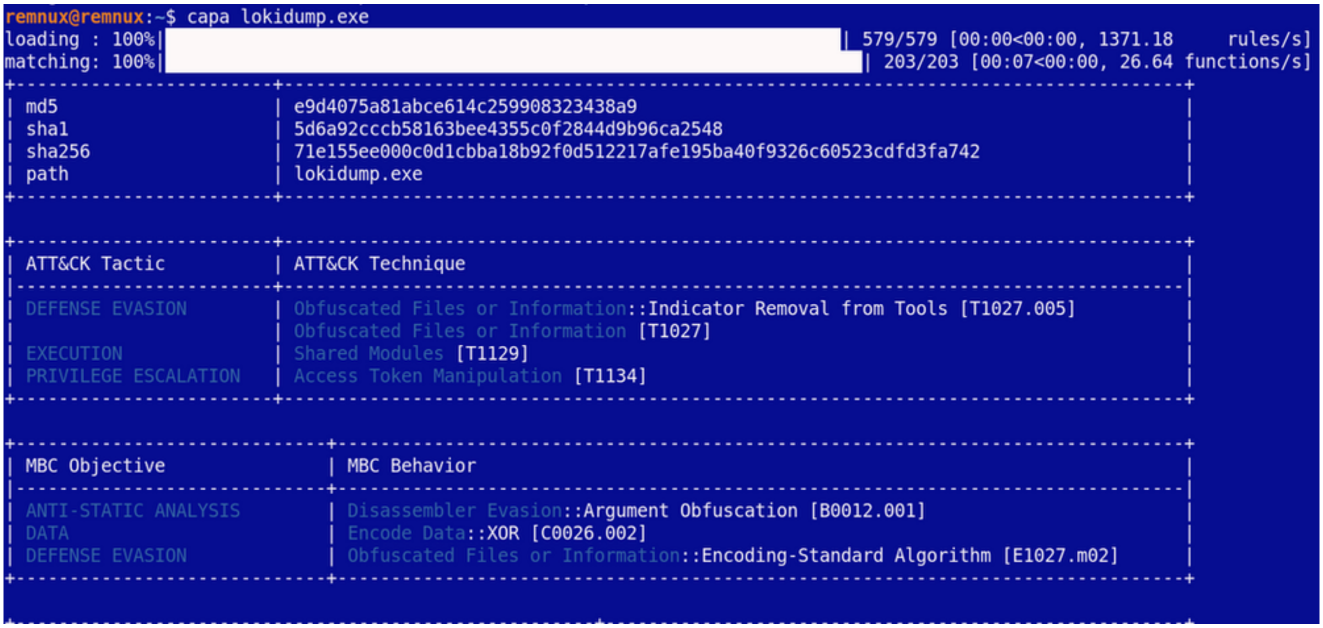
## REBUILDING WITH SCYLLA

So like I mentioned early on, the initial file is an installer and not the actual child process that we've analyzed in the dynamic stage. I'll demonstrate how one could actually get a "tangible" version of the malware that's executing to then analyze it with tools like capa. The tool we'll be using is Scylla, and imports reconstructor developed by NtQuery and available on GitHub.

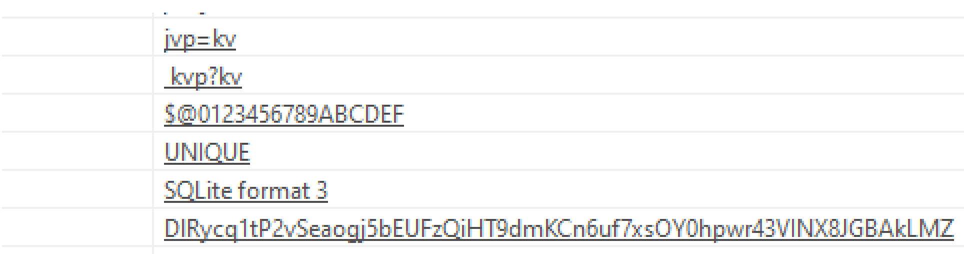
First, make sure the malware is already running. Next open Scylla x86 and attach it to the active process. Click "IAT autosearch," and then "get imports," followed by "dump." Name it something intuitive, and voila. This will reconstruct the executable.




I pushed this version back over to the REMnux box for analysis, and it worked fine with capa:




We can also feed this new executable back into PEStudio for its new assessment:




An interesting catch that I didn't catch before is the reference in strings to SQLite. A lot of browsers saved passwords in SQLite databases which, if we didn't know what this was ahead of time, we could safely lean towards it being a browser password stealer.

## IOCs

**File Hashes:**

Installer:

c27e339893d3e5fc1e61e73ffafac8a7bcf76813a92f91ecfa38535210d6c7a7

Dropped executable:
71e155ee000c0d1cbba18b92f0d512217afe195ba40f9326c60523cdfd3fa742

**Domains:**

s442136.smrtp[.]ru

REFERENCES:

LokiBot Malware (*CISA*)

New Campaign Sees LokiBot Delivered Via Multiple Methods (*TrendMicro*)

Scylla