# ROCSC 2022

*Author: Fărcășanu Tudor Andrei*

## Summary

## minipwn: Pwn

### Description

Desi pare o aplicatie simpla suspectam ca autorul a introdus un backdoor. Trebuie sa aflam daca acesta exista si daca il putem exploata pentru a obtine acces la steag (/flag.txt).
Flag format: ROCSC{sha256}

### Flag proof

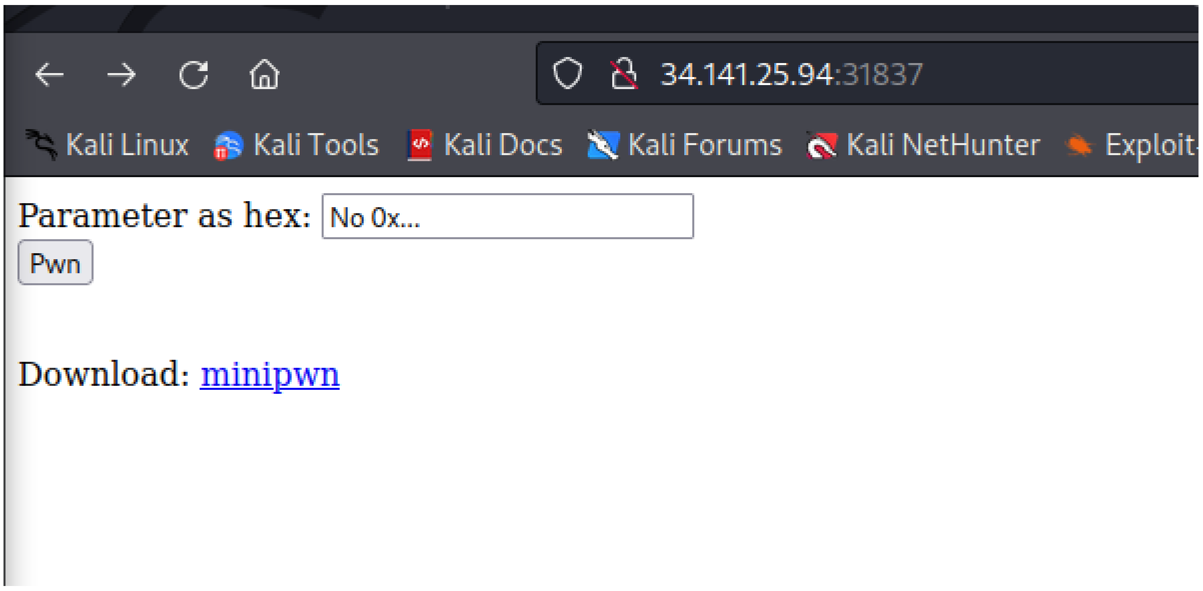
```
ROCSC{220c86cab87f8016f63660d369001d908b94df19ab406f01394e5c5c7eee88ac}
```

### Summary

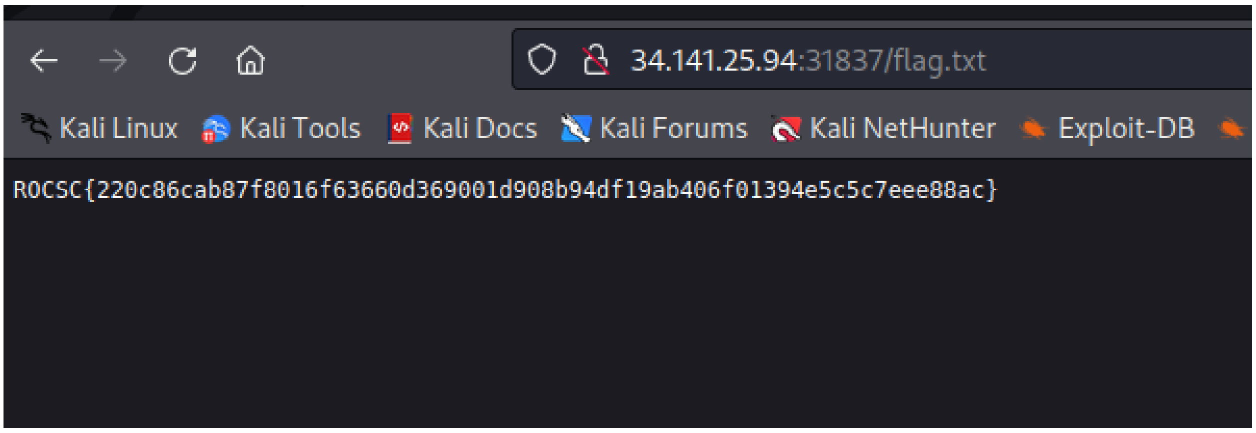
I went to http://34.141.25.94:31837 and then I followed the instructions in the challenge description, and so I went to http://34.141.25.94:31837/flag.txt where I found the flag.

### Details

Step 1:



Step 2:



```
ROCSC{220c86cab87f8016f63660d369001d908b94df19ab406f01394e5c5c7eee88ac}
```

## weird-noise: Forensics

### Description

We have received this message, but unfortunately, we can't understand anything.
The only thing we know is that the message has the following tag: ROBOT 36 Slow Something.

### Flag proof

```
FL4g147#?!23
```

### Summary

I used https://github.com/xdsopl/robot36 to decode the given .wav into a .ppm

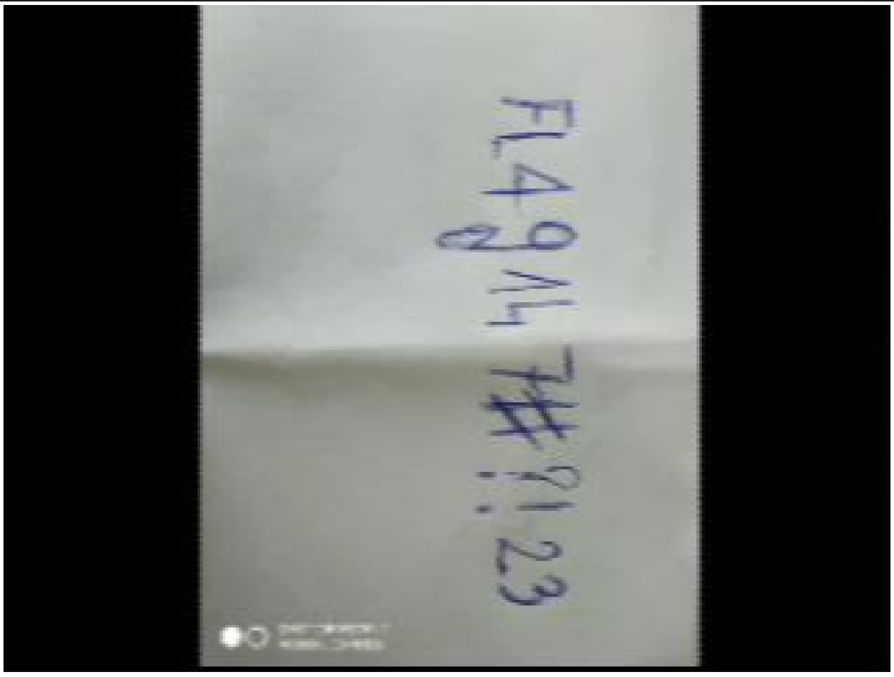
### Details

I googled the tag provided in the challenge description, and then googled robot 36 wav because the given file was in .wav format. Then I cloned and compiled https://github.com/xdsopl/robot36, using the included README. For my linux distro, I also needed to install libasound2-dev and libsdl2-dev libsdl1.2-dev.

```
git clone https://github.com/xdsopl/robot36.git
cd robot36
sudo make
sudo apt-get install libasound2-dev
sudo apt install libsdl2-dev libsdl1.2-dev
```

Then I used the following to extract the image, which contained the flag:

```
./decode ../weird-noise.wav out.ppm
```



## malicious-internet-traffic: Network, Forensics

### Description

During a cybersecurity incident, some malicious network traffic was captured. We need to find answers related to the following questions.
NOTE! The present challenge contains documents infected with real malware. Be cautious and solve this challenge in a virtual environment only.

### Flag proof

1. From which site the malicious Office document was downloaded?

```
http://danielbrink.dk
```

2. Which is the IP of the machine where the malicious document was downloaded?

```
10.12.5.102
```

3. Which is the name of the malicious document downloaded?

```
eForm-869337384710242.doc
```

4. With which malware family was the compromised machine infected?

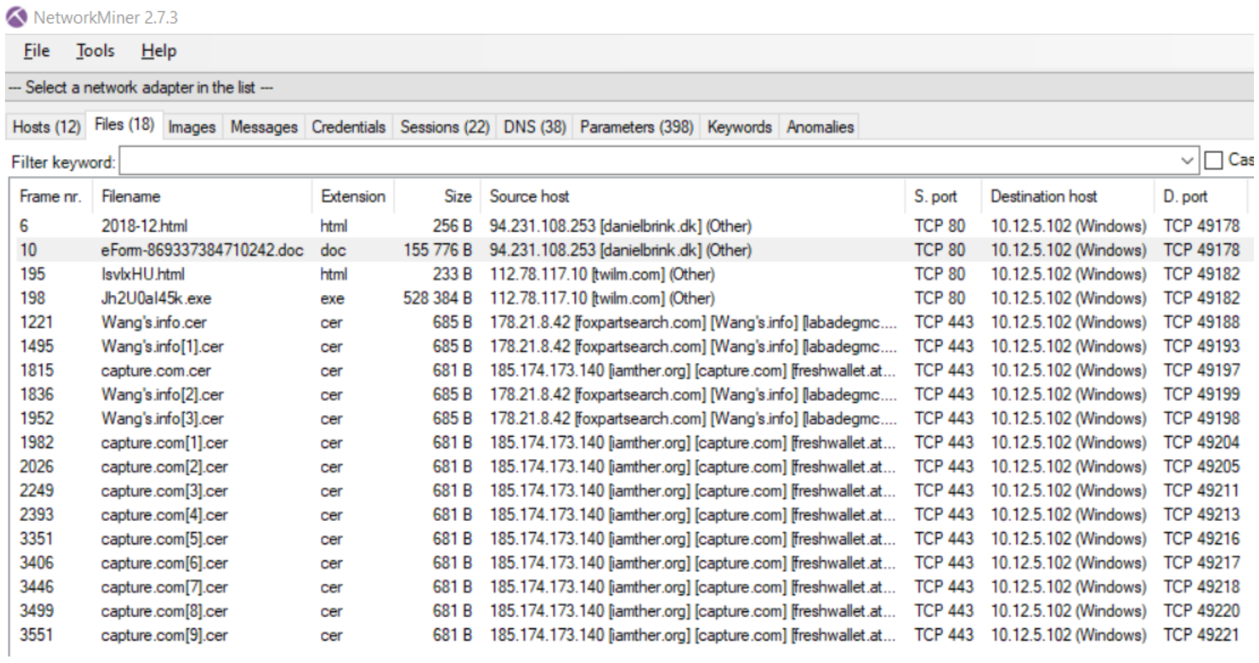
```
emotet
```

### Summary

For questions 1-3 I used NetworkMiner to analyze the packet capture. To answer question 4 I uploaded the malicious document to VirusTotal, which identified the malware family.

### Details

For question 1, after loading the packet capture in NetworkMiner, I looked at the Files tab:

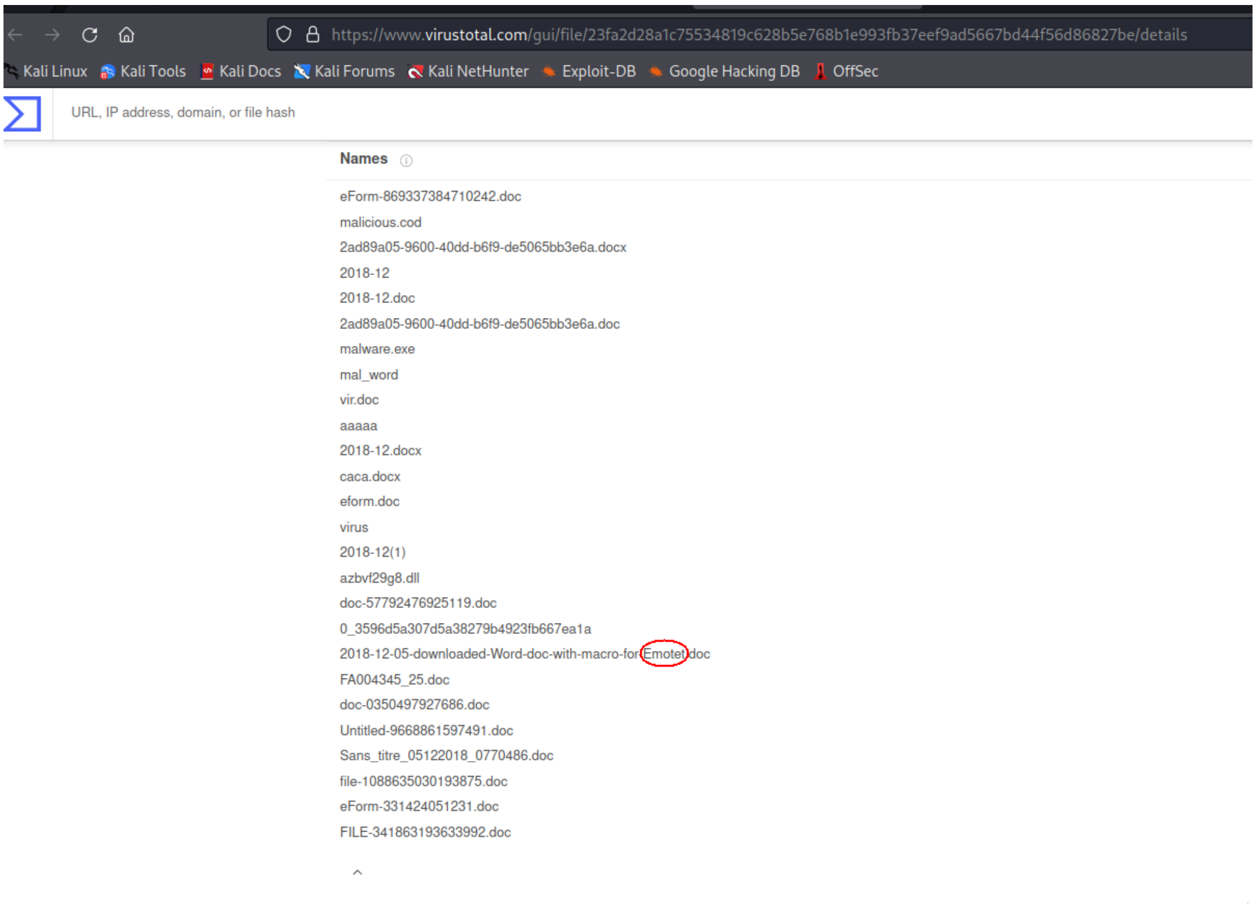


| Frame nr. | Filename | Extension | Size | Source host | S. port | Destination host | D. port |
|---|---|---|---|---|---|---|---|
| 6 | 2018-12.html | html | 256 B | 94.231.108.253 [danielbrink.dk] (Other) | TCP 80 | 10.12.5.102 (Windows) | TCP 49178 |
| 10 | eForm-869337384710242.doc | doc | 155 776 B | 94.231.108.253 [danielbrink.dk] (Other) | TCP 80 | 10.12.5.102 (Windows) | TCP 49178 |
| 195 | IsvlxHU.html | html | 233 B | 112.78.117.10 [twilm.com] (Other) | TCP 80 | 10.12.5.102 (Windows) | TCP 49182 |
| 198 | Jh2U0al45k.exe | exe | 528 384 B | 112.78.117.10 [twilm.com] (Other) | TCP 80 | 10.12.5.102 (Windows) | TCP 49182 |
| 1221 | Wang's.info.cer | cer | 685 B | 178.21.8.42 [foxpartsearch.com] [Wang's.info] [labadegmc.... | TCP 443 | 10.12.5.102 (Windows) | TCP 49188 |
| 1495 | Wang's.info[1].cer | cer | 685 B | 178.21.8.42 [foxpartsearch.com] [Wang's.info] [labadegmc.... | TCP 443 | 10.12.5.102 (Windows) | TCP 49193 |
| 1815 | capture.com.cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49197 |
| 1836 | Wang's.info[2].cer | cer | 685 B | 178.21.8.42 [foxpartsearch.com] [Wang's.info] [labadegmc.... | TCP 443 | 10.12.5.102 (Windows) | TCP 49199 |
| 1952 | Wang's.info[3].cer | cer | 685 B | 178.21.8.42 [foxpartsearch.com] [Wang's.info] [labadegmc.... | TCP 443 | 10.12.5.102 (Windows) | TCP 49198 |
| 1982 | capture.com[1].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49204 |
| 2026 | capture.com[2].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49205 |
| 2249 | capture.com[3].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49211 |
| 2393 | capture.com[4].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49213 |
| 3351 | capture.com[5].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49216 |
| 3406 | capture.com[6].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49217 |
| 3446 | capture.com[7].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49218 |
| 3499 | capture.com[8].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49220 |
| 3551 | capture.com[9].cer | cer | 681 B | 185.174.173.140 [iamther.org] [capture.com] [freshwallet.at... | TCP 443 | 10.12.5.102 (Windows) | TCP 49221 |

Since questions 1-3 mentioned that the file was a “document”, I focused on eForm-869337384710242.doc, and after some trial and error while submitting the flag, figured out it was http://danielbrink.dk.

Question 2 and 3 were answered using the Destination host and respectively Filename columns in the screenshot above.

For question 4 I uploaded the file to VirusTotal, and found the malware family in the Details tab, after expanding the Names section:



https://www.virustotal.com/gui/file/23fa2d28a1c75534819c628b5e768b1e993fb37eef9ad5667bd44f56d86827be/details

Names

eForm-869337384710242.doc
malicious.cod
2ad89a05-9600-40dd-b6f9-de5065bb3e6a.docx
2018-12
2018-12.doc
2ad89a05-9600-40dd-b6f9-de5065bb3e6a.doc
malware.exe
mal_word
vir.doc
aaaaa
2018-12.docx
caca.docx
eform.doc
virus
2018-12(1)
azbvf29g8.dll
doc-57792476925119.doc
0_3596d5a307d5a38279b4923fb667ea1a
2018-12-05-downloaded-Word-doc-with-macro-for-Emotet.doc
FA004345_25.doc
doc-0350497927686.doc
Untitled-9668861597491.doc
Sans_titre_05122018_0770486.doc
file-1088635030193875.doc
eForm-331424051231.doc
FILE-341863193633992.doc

## i-am-root: Forensics

### Description

We need the following information ASAP, a Linux workstation within our organization is manifesting weird behavior.
_NOTE! The present challenge contains documents infected with real malware.
Be cautious and solve this challenge in a virtual environment only.

### Flag proof

1. Which CPU models used the compromised machine?

```
i9-10885H
```

2. Which rootkit managed the attacker to launch on the compromised workstation?

```
diamorphine
```

3. Who developed the rootkit?

```
Victor-Ramos-Mello
```

4. Which is the MAC address of the compromised computer?

```
08:00:27:C6:76:FC
```

5. From which path location on the compromised workstation, the rootkit was launched?

```
/home/ubuntu/Diamorphine
```

### Summary

For question 2 and 3 I used Autopsy, while for 1,4,5 I used grep on the provided image.

### Details

For question 1, I first looked at the output of lscpu on my machine, and inspired by it, I searched for “processor” using the following command (-n for line number, -a to get output from binary files, -e for pattern):

```
grep -na -e "processor" i-am-root.bin > grepoutproc.txt
```

Then, I searched for the two mainstream manufacturers, using:

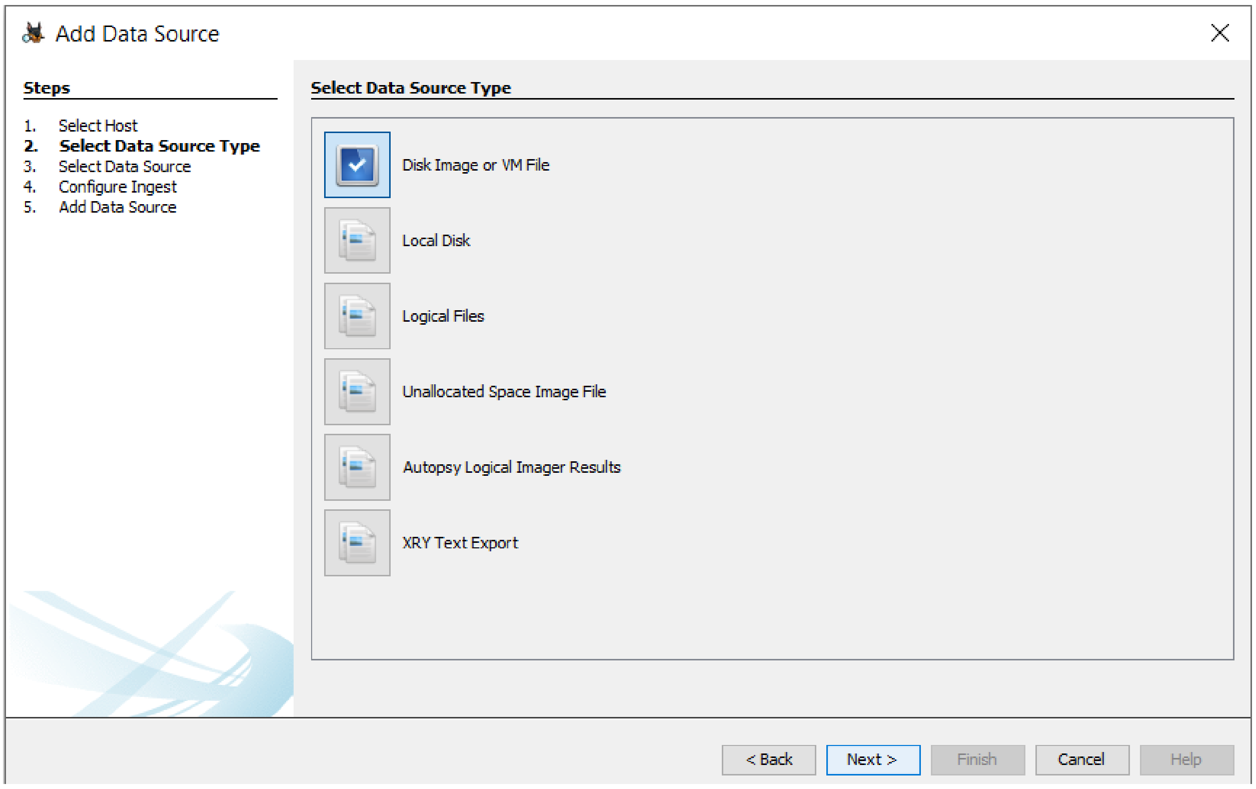
```
grep -ia "intel" grepoutproc.txt
```

```
grep -ia "amd" grepoutproc.txt
```

Somewhere in the output for intel, I found:

```
LSM: Security Framework initializing9&('Yama: becoming
mindful.vb0'AppArmor: AppArmor initialized''TD'Mount-cache hash table
entries: 8192 (order: 4, 65536 bytes, linear)n'\I'Mountpoint-cache hash
table entries: 8192 (order: 4, 65536 bytes, linear)'('*** VALIDATE tmpfs
***tz('*** VALIDATE proc ***''(▒'*** VALIDATE cgroup1 ***''(▒'***
VALIDATE cgroup2 ***Ô4$'process: using mwait in idle threadsã@-'Last
level iTLB entries: 4KB 64, 2MB 8, 4MB 8''D4'Last level dTLB entries:
4KB 64, 2MB 0, 4MB 0, 1GB 4''dQ'Spectre V1 : Mitigation: usercopy/swapgs
barriers and __user pointer sanitizationM'@/'Spectre V2 : Mitigation:
Full generic retpoline'`N'Spectre V2 : Spectre v2 / SpectreRSB
mitigation: Filling RSB on context switch''4$'Speculative Store Bypass:
Vulnerable''4"'MDS: Mitigation: Clear CPU buffers''4$'Freeing SMP
alternatives memory: 40K''
18709653:tb'smpboot: CPU0: Intel(R) Core(TM) i9-10885H CPU @ 2.40GHz
(family: 0x6, model: 0xa5, stepping: 0x2)'0
      hU'Performance Events: unsupported p6 CPU model 165 no PMU driver,
software events only.4'
```

To answer question 2, I loaded the file in Autopsy as as a disk image:


Add Data Source

Steps

1. Select Host
2. **Select Data Source Type**
3. Select Data Source
4. Configure Ingest
5. Add Data Source

**Select Data Source Type**

[✓ Disk Image or VM File]

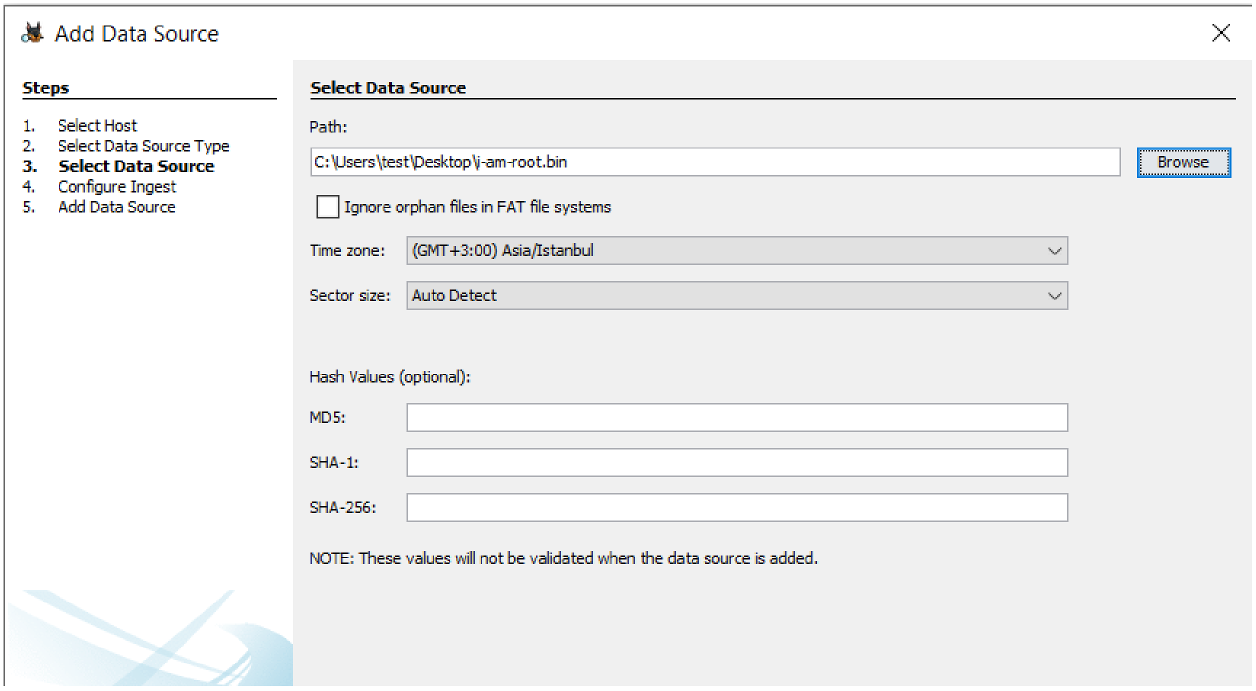
Local Disk

Logical Files

Unallocated Space Image File

Autopsy Logical Imager Results

XRY Text Export


Add Data Source

Steps

1. Select Host
2. Select Data Source Type
3. **Select Data Source**
4. Configure Ingest
5. Add Data Source

**Select Data Source**

Path: C:\Users\test\Desktop\j-am-root.bin

☐ Ignore orphan files in FAT file systems

Time zone: (GMT+3:00) Asia/Istanbul

Sector size: Auto Detect

Hash Values (optional):

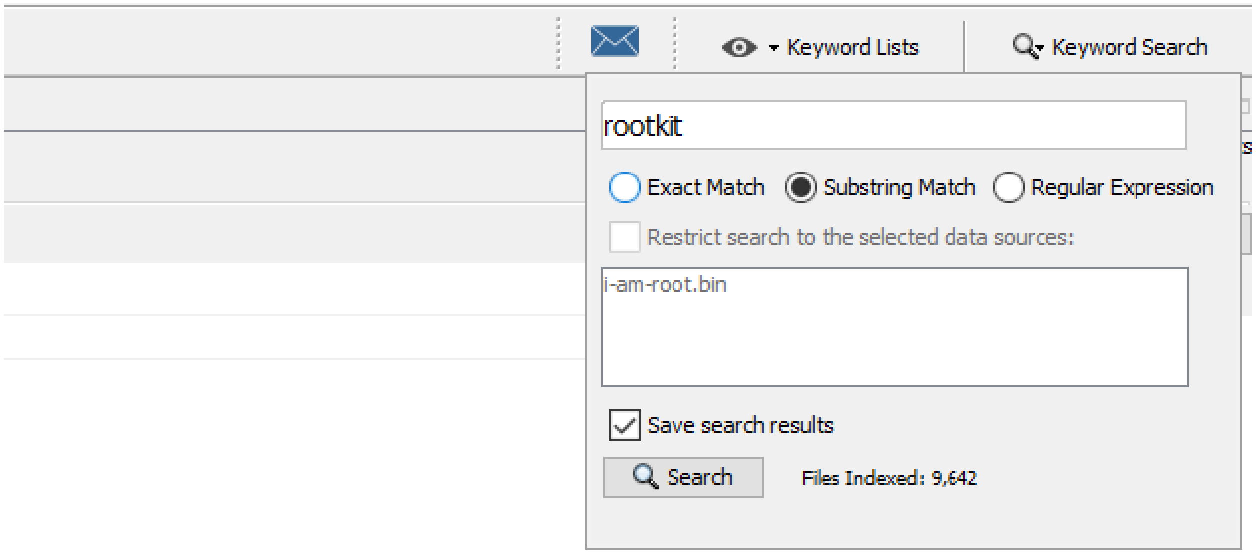
MD5:

SHA-1:

SHA-256:

NOTE: These values will not be validated when the data source is added.

Then, I searched for the substring "rootkit" in the extracted files:



rootkit

Exact Match [✓ Substring Match] Regular Expression

Restrict search to the selected data sources:

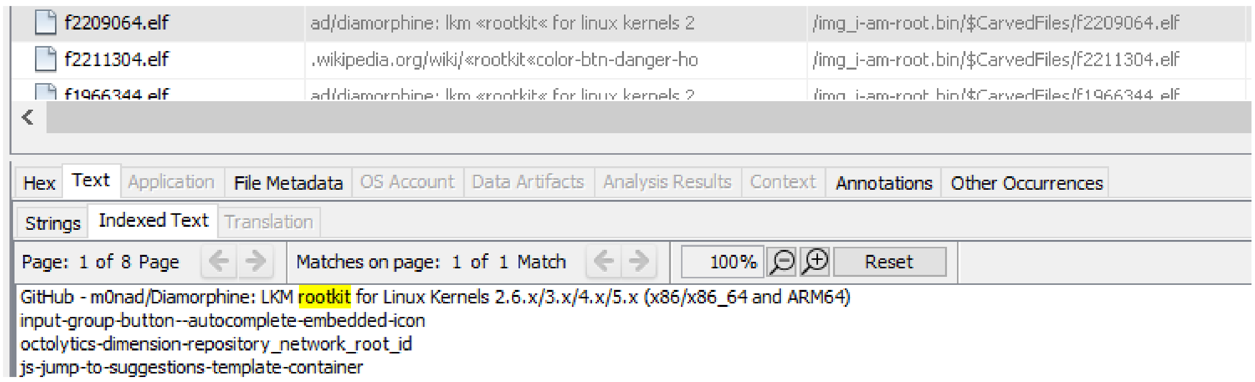
i-am-root.bin

[✓] Save search results

Search

Files Indexed: 9,642

I looked at the results and found the following, and I submitted "diamorphine":

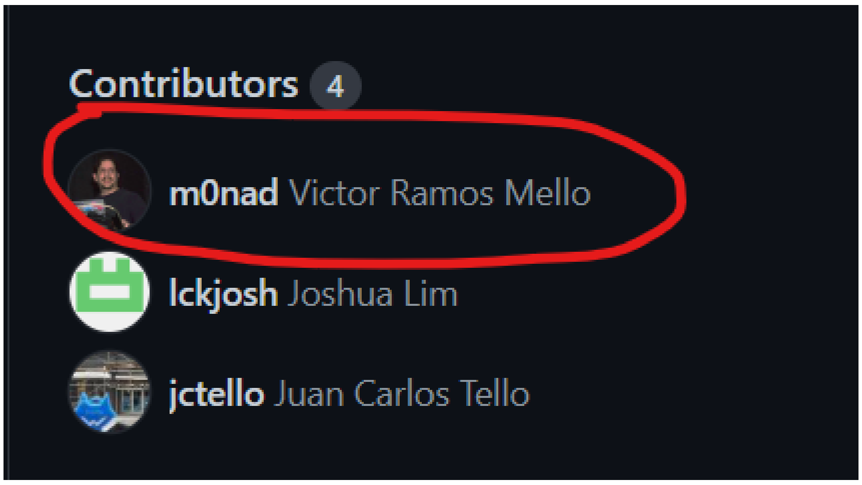


| | | |
|---|---|---|
| f2209064.elf | ad/diamorphine: lkm «rootkit« for linux kernels 2 | /img_i-am-root.bin/$CarvedFiles/f2209064.elf |
| f2211304.elf | .wikipedia.org/wiki/«rootkit«color-btn-danger-ho | /img_i-am-root.bin/$CarvedFiles/f2211304.elf |
| f1966344.elf | ad/diamorphine: lkm «rootkit« for linux kernels 2 | /img_i-am-root.bin/$CarvedFiles/f1966344.elf |

Page: 1 of 8 Page    Matches on page: 1 of 1 Match    100%

```
GitHub - m0nad/Diamorphine: LKM rootkit for Linux Kernels 2.6.x/3.x/4.x/5.x (x86/x86_64 and ARM64)
input-group-button--autocomplete-embedded-icon
octolytics-dimension-repository_network_root_id
js-jump-to-suggestions-template-container
```

For question 3, I googled the github repo found in the screenshot above, and identified the author's name:



Contributors 4

m0nad Victor Ramos Mello

lckjosh Joshua Lim

jctello Juan Carlos Tello

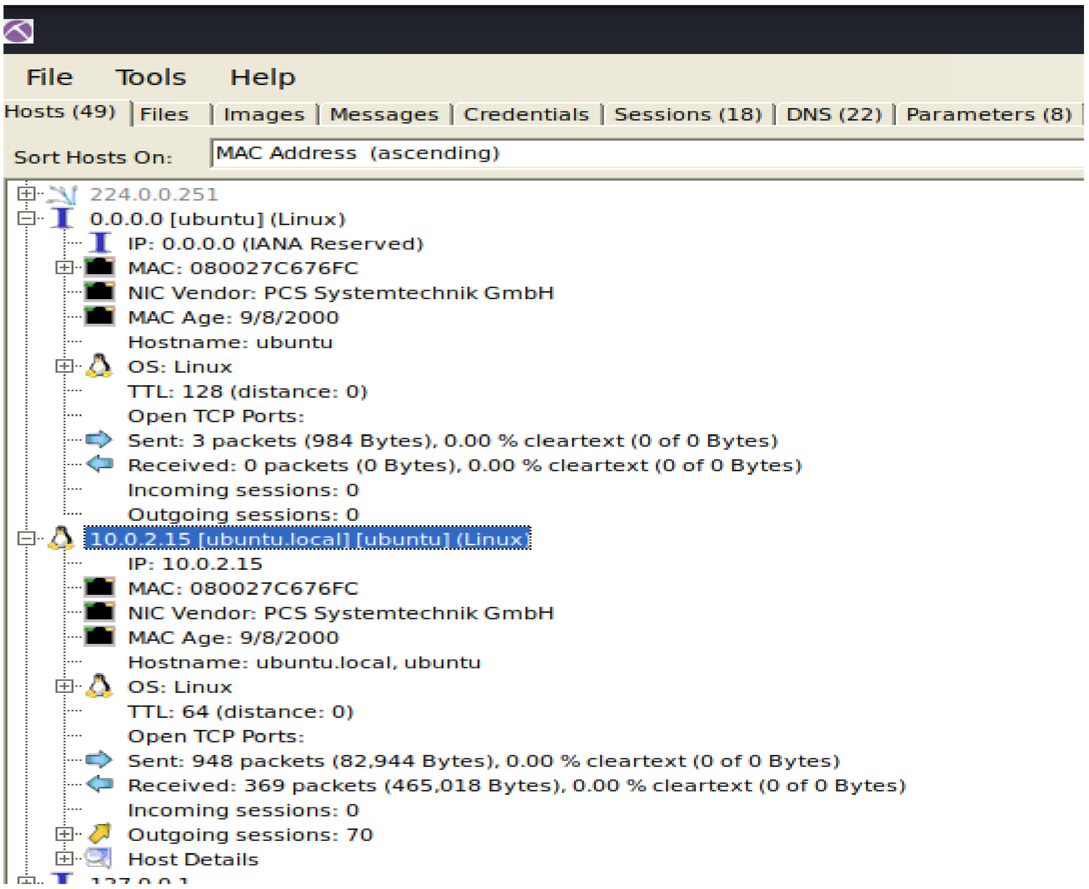
For question 4, I used bulk_extractor like so:

```
bulk_extractor i-am-root.bin -o bulkextoutp
cd bulkextoutp
cat ether.txt
```

In ether.txt, I found 2 different mac addresses:

```
# BANNER FILE NOT PROVIDED (-b optioxn)
# BULK_EXTRACTOR-Version: 2.0.0
# Feature-Recorder: ether
# Filename: i-am-root.bin
# Feature-File-Version: 1.1
48726208    08:00:27:C6:76:FC    (ether_dhost)
48726208    52:54:00:12:35:02    (ether_shost)
48730048    08:00:27:C6:76:FC    (ether_dhost)
48730048    52:54:00:12:35:02    (ether_shost)
48731968    08:00:27:C6:76:FC    (ether_dhost)
48731968    52:54:00:12:35:02    (ether_shost)
48733888    08:00:27:C6:76:FC    (ether_dhost)
48733888    52:54:00:12:35:02    (ether_shost)
58144064    08:00:27:C6:76:FC    (ether_dhost)
58144064    52:54:00:12:35:02    (ether_shost)
58145984    08:00:27:C6:76:FC    (ether_dhost)
58145984    52:54:00:12:35:02    (ether_shost)
58147904    08:00:27:C6:76:FC    (ether_dhost)
58147904    52:54:00:12:35:02    (ether_shost)
58149824    08:00:27:C6:76:FC    (ether_dhost)
```

I also used NetworkMiner to inspect the extracted packet capture:

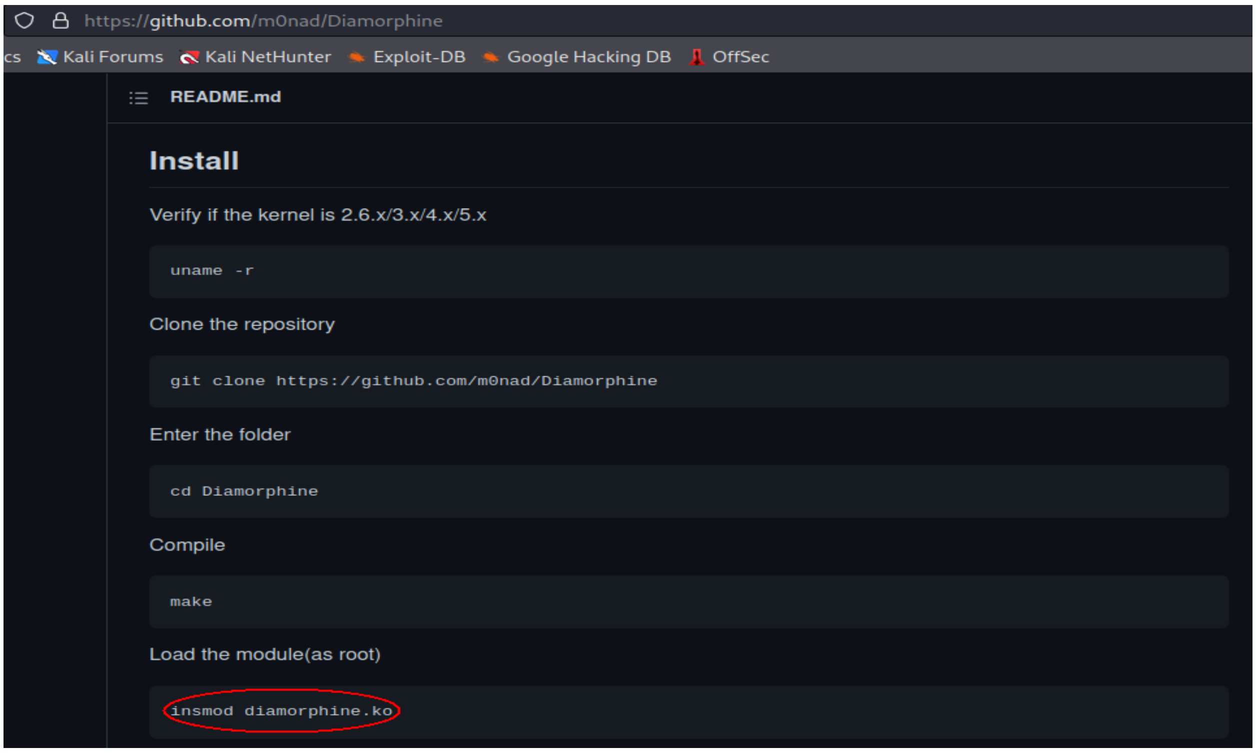


Sort Hosts On: MAC Address (ascending)

```
224.0.0.251
0.0.0.0 [ubuntu] (Linux)
    IP: 0.0.0.0 (IANA Reserved)
    MAC: 080027C676FC
    NIC Vendor: PCS Systemtechnik GmbH
    MAC Age: 9/8/2000
    Hostname: ubuntu
    OS: Linux
    TTL: 128 (distance: 0)
    Open TCP Ports:
    Sent: 3 packets (984 Bytes), 0.00 % cleartext (0 of 0 Bytes)
    Received: 0 packets (0 Bytes), 0.00 % cleartext (0 of 0 Bytes)
    Incoming sessions: 0
    Outgoing sessions: 0
10.0.2.15 [ubuntu.local] [ubuntu] (Linux)
    IP: 10.0.2.15
    MAC: 080027C676FC
    NIC Vendor: PCS Systemtechnik GmbH
    MAC Age: 9/8/2000
    Hostname: ubuntu.local, ubuntu
    OS: Linux
    TTL: 64 (distance: 0)
    Open TCP Ports:
    Sent: 948 packets (82,944 Bytes), 0.00 % cleartext (0 of 0 Bytes)
    Received: 369 packets (465,018 Bytes), 0.00 % cleartext (0 of 0 Bytes)
    Incoming sessions: 0
    Outgoing sessions: 70
    Host Details
```

Because it was mentioned in the challenge description the workstation was running Linux, I submitted “08:00:27:C6:76:FC”, as for the other MAC address found in ether.txt an OS couldn’t be identified.

For question 5, I looked at the Install section in the README on the rootkit’s github. The instructions of the mentioned using “insmod diamorphine.ko” to load the module:


https://github.com/m0nad/Diamorphine
cs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
README.md

## Install

Verify if the kernel is 2.6.x/3.x/4.x/5.x

```
uname -r
```

Clone the repository

```
git clone https://github.com/m0nad/Diamorphine
```

Enter the folder

```
cd Diamorphine
```

Compile

```
make
```

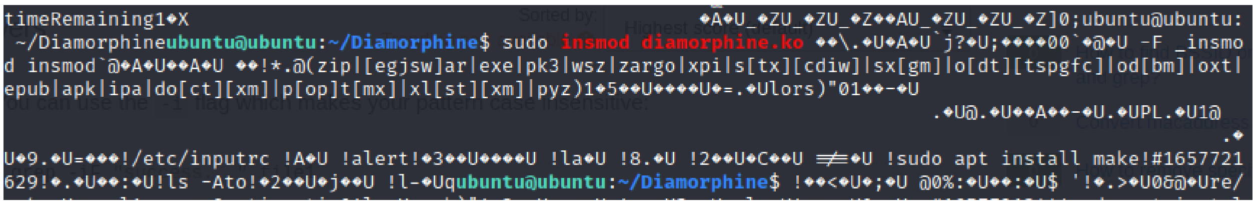
Load the module(as root)

```
insmod diamorphine.ko
```

As such, I used the following command:

```
grep -na -e "insmod diamorphine.ko" i-am-root.bin
```

In its output, I found highlighted by zsh the user and machine name, as well as the path where the load command was run:



```
timeRemaining1◆X                                           ◆A◆U_◆ZU_◆ZU_◆Z◆◆AU_◆ZU_◆ZU_◆Z]0;ubuntu@ubuntu:
 ~/Diamorphineubuntu@ubuntu:~/Diamorphine$ sudo insmod diamorphine.ko ◆◆\.◆U◆A◆U`j?◆U;◆◆◆◆00`◆@◆U -F _insmo
d insmod`@◆A◆U◆◆A◆U ◆◆!*.@(zip|[egjsw]ar|exe|pk3|wsz|zargo|xpi|s[tx][cdiw]|sx[gm]|o[dt][tspgfc]|od[bm]|oxt|
epub|apk|ipa|do[ct][xm]|p[op]t[mx]|xl[st][xm]|pyz)1◆5◆◆U◆◆◆◆U◆=.◆Ulors)"01◆◆-◆U
                                                                                     .◆U@.◆U◆◆A◆◆-◆U.◆UPL.◆U1@
                                                                                                            .◆
U◆9.◆U=◆◆◆!/etc/inputrc !A◆U !alert!◆3◆◆U◆◆◆◆U !la◆U !8.◆U !2◆◆U◆C◆◆U ≢◆U !sudo apt install make!#1657721
629!◆.◆U◆◆:◆U!ls -Ato!◆2◆◆U◆j◆◆U !l-◆Uqubuntu@ubuntu:~/Diamorphine$ !◆◆<◆U◆;◆U @0%:◆U◆◆:◆U$ '!◆.>◆U0δ@◆Ure/
```

With this information, I assembled the path (~=/home/ubuntu): `/home/ubuntu/Diamorphine`