SSH SECURISE

PREREQUIS

2 machines linux sur le même réseau avec un accés sur internet

Partie 1 Installation

Dans cette partie nous allons vérifier que le serveur SSH est bien installé :

- Vérifier si le serveur est installé :

```
root@deb11-cours1:~# dpkg -l openssh-server
Souhait=inconnU/Installé/suppRimé/Purgé/H=à garder
| État=Non/Installé/fichier-Config/dépaqUeté/échec-conFig/H=semi-installé/W=attend-traitement-
déclenchements
|/ Err?=(aucune)/besoin Réinstallation (État,Err: majuscule=mauvais)
||/ Nom            Version           Architecture Description
+++-==============-=================-============-
=================================================================
ii  openssh-server 1:8.4p1-5+deb11u1 amd64        secure shell (SSH) server, for secure access
from remote machines
root@deb11-cours1:~#
```

- Si le serveur n'est pas installé alors :

```
apt install openssh-server
```

- Une fois installé vous pouvez tenter de vous y connecter avec le user que vous avez créé au départ pour ma part c'est kvega :

```
ssh
kvega@192.168.21.200


                              ✔  10217  08:33:36
kvega@192.168.21.200's password:
Linux deb11-cours1 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64
```

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov  8 08:32:48 2022 from 192.168.21.1
kvega@deb11-cours1:~$
```

A cette étape nous pouvons nous connecter sur le premier serveur.

Partie 2 Connection entre les deux serveurs en SSH

Dans cette partie nous verrons comment faire communiquer deux serveurs en SSH sans avoir a taper le mot de passe

- Revenir a la PARTIE 1 et vérifier que sur l'autre serveur SSH est bien installé
- Générer une paire clés sur le serveur 1

> Pour les besoin du TP il faudrat être **root** sur le serveur.

```
root@deb11-cours1:~# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ed25519
Your public key has been saved in /root/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:rcrV5tIU6L4ZNEOGDF9Wom9M2YMEoEsBSTb/dDDOwRU root@deb11-cours1
The key's randomart image is:
+--[ED25519 256]--+
|o*. +=oE*..      |
|o ooo=+* =       |
|  o. +=.*.o      |
| . .o .*....     |
|  .  . .S ..     |
|       o.=.      |
|       .+oo      |
```

```
|     . oo=.      |
|      o oo.      |
+----[SHA256]-----+
```

Il faut tout laisser par défault et ne pas mettre de mot de passe: Quand il est demandé simplement taper ENTER.

- Permettre la connection avec le user root sur le serveur B

```
root@deb11-cours2:~# grep PermitRootLogin /etc/ssh/sshd_config
#PermitRootLogin prohibit-password
# the setting of "PermitRootLogin without-password".
```

Sur le serveur on vois que le fichier de configuration du serveur SSH ne permet pas la connexion au serveur via l'utilisateur ROOT. Pour lui donner accès nous allons donc devoir autoriser l'utilisateur root à se connecter :

```
root@deb11-cours2:~# sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/g'
/etc/ssh/sshd_config
root@deb11-cours2:~# grep PermitRootLogin /etc/ssh/sshd_config
PermitRootLogin prohibit-password
```

On reload le service SSH

```
service ssh reload
```

- Tenter une connexion du serveur A au serveur B:

```
root@deb11-cours1:~# ssh root@192.168.21.201
root@192.168.21.201's password:
Linux deb11-cours2 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov  8 08:16:03 2022
root@deb11-cours2
```

On vois que la connexion se fait bien. Nous allons donc pouvoir partager la clé ssh du serveur A au serveur B.

- partage de clé ssh

```
root@deb11-cours1:~# ssh-copy-id root@192.168.21.201
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_ed25519.pub"
/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@192.168.21.201's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'root@192.168.21.201'"
and check to make sure that only the key(s) you wanted were added.
```

La clé ssh Publique vient d'être copier. on peut s'en rendre compte en se rendant sur le serveur B et en regardant dans le fichier /root/.ssh/authorized_keys que notre clé est bien présente. **Sur le serveur A:**

```
root@deb11-cours1:~# cat /root/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFEE2zBCapNve8gCCRR7xV7PzinFjckSi8iOb+corG4M root@deb11-cours1
```

Sur le serveur B:

```
root@deb11-cours2:~# cat /root/.ssh/authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFEE2zBCapNve8gCCRR7xV7PzinFjckSi8iOb+corG4M root@deb11-cours1
```

Une fois la connexion établie il faudrat resécuriser le serveur B :

```
root@deb11-cours2:~# sed -i 's/PermitRootLogin yes/#PermitRootLogin yes/g' /etc/ssh/sshd_config
root@deb11-cours2:~# grep PermitRootLogin /etc/ssh/sshd_config
#PermitRootLogin yes
# the setting of "PermitRootLogin without-password".
root@deb11-cours2:~# service ssh reload
```

Répeter la partie 2 sur le serveur A

Tester si la connexion se fait bien sans mot de passe:

1. Créer un fichier sur le serveur B

```
root@deb11-cours2:~# touch toto
```

2. Envoyer le fichier sur le serveur A

```
scp toto root@192.168.21.200:/root/
```

3. Vérifier sur le serveur A que le fichier s'y trouve bien

```
root@deb11-cours1:~# cd /root
root@deb11-cours1:~# ls
toto
```

On peut aussi se connecter en ssh du serveur A au serveur B et inversement pour tester :

```
ssh root@192.168.21.201
Linux deb11-cours2 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov  8 08:57:01 2022 from 192.168.21.200
root@deb11-cours2:~#
```

Aucun mot de passe ne m'a été demandé.

EXRECICE

créer un script qui envoie un fichier vers le serveur distant Ex de script :

```
#!/bin/bash
test=$(ping -c1 8.8.8.8)
if [ $? -eq 0 ]
then
        echo "c'est super !!"
        #Je verifie que le fichier toto existe si oui
```

```
        #je l'envoie
        #sinon je le créer
else
        echo "ça marche pas !!"
fi
```

Partie 3 sécurisation de SSH sur le réseau

On va changer le port d'écoute du serveur SSH

```
root@deb11-cours2:~# sed -i 's/#Port 22/Port 2222/g' /etc/ssh/sshd_config
root@deb11-cours2:~# service ssh restart
root@deb11-cours2:~# netstat -lntp
Connexions Internet actives (seulement serveurs)
Proto Recv-Q Send-Q Adresse locale          Adresse distante        Etat        PID/Program
name
tcp        0      0 0.0.0.0:2222            0.0.0.0:*               LISTEN      633/sshd:
/usr/sbin
tcp6       0      0 :::2222
```

Ne pas oublier de changer le script en spécifiant le bon port.

Partie 4 test avec KALI et attaque brutte force

Installer Kali

1. Télécharger Kali https://www.kali.org/get-kali/#kali-virtual-machines
    - le mot de passe et le User par défaut de kali sont : User: kali Mdp: kali

On va mettre à jour le clavier dans Kali pour le metre en français:

```
sudo dpkg-reconfigure keyboard-configuration
#laisser par défaut
#choisir Other et choisir french
#tout laisser par défaut
reboot
```

On va mettre à jour kali:

```
sudo apt update
sudo apt upgrade
sudo apt install open-vm-tools
```

tester le Réseau avec Kali

1. On va tester les IP/Hosts présent sur le réseau:

```
nmap -sP 192.168.21.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2022-11-14 04:22 EST
Nmap scan report for fr-lap10398 (192.168.21.1)
Host is up (0.00035s latency).
Nmap scan report for 192.168.21.2
Host is up (0.00036s latency).
Nmap scan report for 192.168.21.132
Host is up (0.000092s latency).
Nmap scan report for 192.168.21.200
Host is up (0.00026s latency).
Nmap scan report for 192.168.21.201
Host is up (0.00023s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.98 seconds
```

2. On va scanner chaque IPs sur le réseau:

```
nmap -sV  192.168.21.201
Starting Nmap 7.93 ( https://nmap.org ) at 2022-11-14 04:27 EST
Nmap scan report for 192.168.21.201
Host is up (0.00028s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
2222/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (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: 1 IP address (1 host up) scanned in 0.51 seconds
```

3. Dans un premier temps, sur l'un des deux serveurs SSH, analyser le cache ARP respectif des deux machines à l’aide de la commande (pour avoir des informations dans la cache arp, vous devrez peut-être au préalable lancer un ping sur chaque machine présente dans le réseau ou relancer les commandes nmap) : *Voici mon résultat:*

```
ip neigh show
192.168.21.1 dev ens160 lladdr 00:50:56:c0:00:08 REACHABLE
192.168.21.2 dev ens160 lladdr 00:50:56:fd:96:81 REACHABLE
192.168.21.132 dev ens160 lladdr 00:0c:29:76:db:c7 STALE
```

4. Sur Kali, à l'aide de l'outil Metasploit, On va brut forcer le mot de passe d'un utilisateur: a. création d'un nouvel utilisateur sur le serveur B:

```
useradd -s /bin/bash toto
```

b. Mise à jour du mot de passe de toto:

```
#mettre azerty comme mot de passe a toto
passwd toto
```

c. Créer le fichier le mot de passe

```
cd
pwgen -sBn 9 24 -1 > /root/Motdepasse.txt
echo "azerty" >> /root/Motdepasse.txt
```

d. Se servir de Metasploit et le configurer: **A cette étape chaque ligne est testée par chaque ligne du fichier** `/usr/share/wordlists/sqlmap.txt`. Pour aller plus vite nous allons spécifier le User:

```
msfconsole
show options
use auxiliary/scanner/ssh/ssh_login
set RPORT 2222
set RHOSTS 192.168.21.201
set USERNAME toto
set PASS_FILE /root/Motdepasse.txt
set THREADS 4
set VERBOSE true
run
```

Partie 5 sécurisation avec Iptables

> Attention il ne faut pas faire cette étape sur le serveur qui se fait brute-forcer ! {.is-warning}

1. Installation de Iptables:

```
apt install iptables
```

On peut lister les tables:

```
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
```

2. Ajout d'une régle de filtrage *Ici je vais autoriser seulement mon post sur le port 22 donc SSH de la machine*

```
iptables -A INPUT -m state --state NEW,ESTABLISHED,RELATED --source 192.168.21.1 -p tcp --
dport ssh -j ACCEPT
iptables -A INPUT -j DROP
iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  192.168.21.1         anywhere             state NEW,RELATED,ESTABLISHED
tcp dpt:ssh
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

Pour remttre Iptables à "zero" il faut:

```
iptables -F
```

Ps: Après redémarrage à ce stade les tables ne vont pas rester en l'état. Il faudrat les remettre en place à chaque redémarrage pour le moment.

Revision #2
Created 2024-01-05 09:43:49 UTC by kvega
Updated 2024-01-05 09:50:12 UTC by kvega