

Projet SSH Secure SHell

I. Installation puis configuration d'un serveur SSH (OpenSSH) sur Ubuntu

1. Dans un premier temps, nous allons mettre à jour la liste des fichiers dans les dépôts APT avec la commande suivante à saisir dans le terminal : `sudo apt-get update`

```
samy@ubuntu:~$ sudo apt-get update
[sudo] password for samy:
Hit:1 http://us.archive.ubuntu.com/ubuntu impish InRelease
Hit:2 http://security.ubuntu.com/ubuntu impish-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu impish-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu impish-backports InRelease
Reading package lists... Done
```

2. Ensuite, nous allons pouvoir installer le serveur SSH(OpenSSH) avec la commande : `sudo apt-get install openssh-server`
3. Si le serveur SSH ne s'active pas automatiquement, il faut saisir la commande suivante : `sudo service ssh start`
4. Nous allons maintenant vérifier le statut du serveur SSH avec cette commande : `sudo service ssh status`

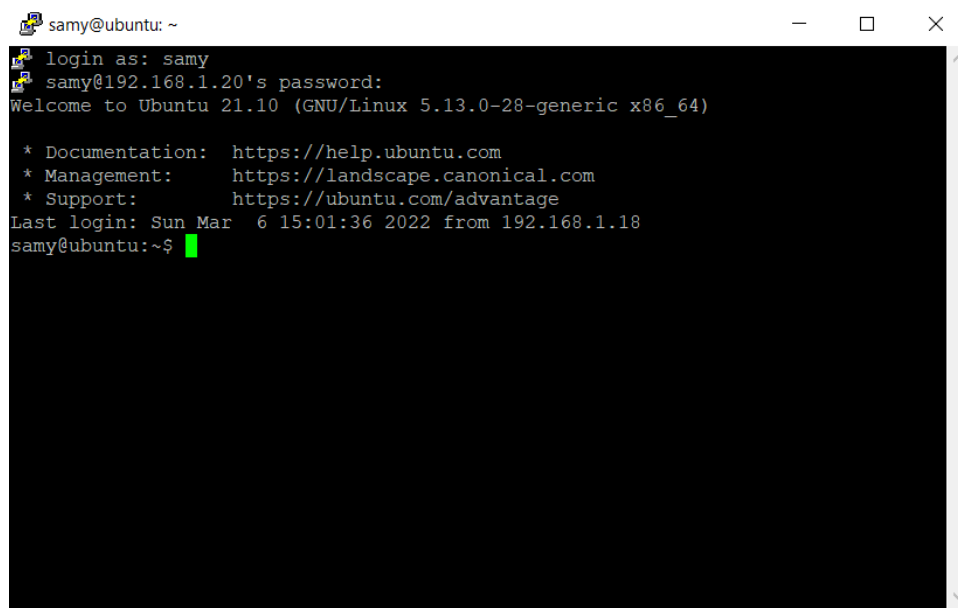
```
samy@ubuntu:~$ sudo service ssh status
[sudo] password for samy:
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2022-03-06 14:14:18 CET; 8min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 46775 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
  Main PID: 46776 (sshd)
    Tasks: 1 (limit: 4583)
   Memory: 1.0M
      CPU: 15ms
   CGroup: /system.slice/ssh.service
           └─46776 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Mar 06 14:14:18 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Mar 06 14:14:18 ubuntu sshd[46776]: Server listening on 0.0.0.0 port 22.
lines 1-15...skipping...
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2022-03-06 14:14:18 CET; 8min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 46775 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
  Main PID: 46776 (sshd)
    Tasks: 1 (limit: 4583)
   Memory: 1.0M
      CPU: 15ms
   CGroup: /system.slice/ssh.service
           └─46776 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Mar 06 14:14:18 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Mar 06 14:14:18 ubuntu sshd[46776]: Server listening on 0.0.0.0 port 22.
Mar 06 14:14:18 ubuntu sshd[46776]: Server listening on :: port 22.
Mar 06 14:14:18 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
```

Cela signifie que le serveur SSH est en cours d'exécution sur le système, Active (running).

Nous pouvons désormais accéder au serveur avec le logiciel PuTTY dans lequel nous allons saisir l'adresse IP présente sur Ubuntu ainsi que le port 22, port ssh par défaut.



```
samy@ubuntu: ~
login as: samy
samy@192.168.1.20's password:
Welcome to Ubuntu 21.10 (GNU/Linux 5.13.0-28-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage
Last login: Sun Mar  6 15:01:36 2022 from 192.168.1.18
samy@ubuntu:~$
```

5. Nous pouvons désormais modifier la configuration du serveur SSH avec la commande suivante : `sudo nano /etc/ssh/sshd_config`

Nous pouvons par exemple modifier le port de SSH, afficher une bannière, empêcher les connexions SSH en root.

```
# $OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $

# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

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

#Port 22
#AddressFamily anys
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
```

II. Installation et configuration de fail2ban

1. Installer fail2ban avec la commande à saisir dans le terminal : `sudo apt install fail2ban`
2. Nous pouvons dès à présent paramétrer fail2ban dans le fichier de configuration : `nano jail.local` dans le répertoire `etc/fail2ban`

Ici j'ai déterminé le nombre maximum d'essais à 2 avant un bannissement dans la jail sshd qui concerne le serveur SSH.

```
# SSH servers
#

[sshd]

# To use more aggressive sshd modes set filter parameter "mode" in jail.local:
# normal (default), ddos, extra or aggressive (combines all).
# See "tests/files/logs/sshd" or "filter.d/sshd.conf" for usage example and details.
#mode = normal
enabled = true
port = ssh
logpath = /var/log/auth.log
backend = %(sshd_backend)s
maxretry = 2
```

Nous constatons qu'après deux essais, nous sommes bien bannis par fail2ban.

```
samy@ubuntu:~/Desktop$ python3 ssh.py
hostname :192.168.163.130
username :samy
password :azrzar
Traceback (most recent call last):
  File "/home/samy/Desktop/ssh.py", line 11, in <module>
    ssh_client.connect(hostname=n,username=m,password=a)
  File "/usr/lib/python3/dist-packages/paramiko/client.py", line 368, in connect
    raise NoValidConnectionsError(errors)
paramiko.ssh_exception.NoValidConnectionsError: [Errno None] Unable to connect to port 22 on 192.168.163.130
```

```
samy@ubuntu:~/Desktop$ sudo fail2ban-client status sshd
Status for the jail: sshd
|- Filter
| |- Currently failed: 0
| |- Total failed:     5
| `-- File list:      /var/log/auth.log
`- Actions
   |- Currently banned: 1
   |- Total banned:    2
   `-- Banned IP list: 192.168.163.130
```

III. Programme Python pour SSH

J'ai fait un programme python à l'aide du module Paramiko, qui permet d'intégrer des fonctions ssh, pour pouvoir se connecter à mon serveur automatiquement. Nous devons auparavant installer python et le module Paramiko avec le terminal (sudo apt-get install -y python-paramiko)

```
import paramiko

n=str(input("hostname (IP) :"))
m=str(input("username :"))
a=str(input("password :"))

command = "ls"

ssh_client =paramiko.SSHClient()
ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
ssh_client.connect(hostname=n,username=m,password=a)

commands = [
    "pwd",
    "id",
    "uname -a",
    "df -h"
]
print("Vous êtes connecté au serveur SSH")
for command in commands:
    print("="*50, command, "="*50)
    stdin, stdout, stderr = ssh_client.exec_command(command)
    print(stdout.read().decode())
    err = stderr.read().decode()
    if err:
        print(err)
lines = stdout.readlines()
print(lines)
```

On peut donc soit saisir manuellement les informations nécessaires pour se connecter au serveur ou les saisir directement à la place des variables string pour plus d'optimisation.

Je peux me connecter sur le serveur ssh si j'ai saisi un hostname, un username et un password correcte.

```
samy@ubuntu:~/Desktop$ python3 ssh.py
hostname (IP) :192.168.163.130
username :samy
password :
['Desktop\n', 'Documents\n', 'Downloads\n', 'jail.local\n', 'Music\n', 'Pictures\n', 'Public\n', 'server.js\n', 'snap\n', 'Templates\n', 'Videos\n']
Vous êtes connecté au serveur SSH
```

Si je me trompe deux fois dans la saisit des informations concernant la connexion (username, password), je serais bannie pendant un certain temps par fail2ban (je peux vérifier cela avec la commande fail2ban status sshd).

```
samy@ubuntu:~/Desktop$ python3 ssh.py
hostname (IP) :192.168.163.130
username :samy
password :jesaispas
Traceback (most recent call last):
  File "/home/samy/Desktop/ssh.py", line 11, in <module>
    ssh_client.connect(hostname=n,username=m,password=a)
  File "/usr/lib/python3/dist-packages/paramiko/client.py", line 368, in connect
    raise NoValidConnectionsError(errors)
paramiko.ssh_exception.NoValidConnectionsError: [Errno None] Unable to connect to port 22 on 192.168.163.130
```