Compte Rendu

TP 2 : IDS (SNORT) & Machine attaquante (Kali Linux)

Étudiante

SASSI Yosra

(ING3)

TP 2 : IDS (SNORT) & Machine attaquante (Kali Linux)

1. Installation et mise en place de l'environnement

L'objectif de cette première étape est de mettre en place un environnement de test permettant la réalisation de tests d'intrusion et la détection d'attaques réseau à l'aide de l'IDS Snort.

1.1 Outils et technologies utilisés

Les outils suivants ont été utilisés :

- GNS3 : simulateur de réseaux pour la création de la topologie
- VirtualBox : hyperviseur pour l'exécution des machines virtuelles
- Kali Linux : machine attaquante
- Ubuntu Server 20.04 LTS : machine IDS exécutant Snort
- VPCS : machine vulnérable simulée
- Ethernet Switch (GNS3) : interconnexion des équipements

1.2 Installation de la machine Kali Linux (attaquant) : **192.168.56.10/24**

```
┌──(root㉿kali)-[~]
└─# sudo ip addr flush dev eth0

┌──(root㉿kali)-[~]
└─# sudo ip addr add 192.168.2.10/24 dev eth0

┌──(root㉿kali)-[~]
└─# sudo ip link set eth0 up

┌──(root㉿kali)-[~]
└─# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
ault 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 noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP g
roup default qlen 1000
    link/ether 08:00:27:63:b0:05 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.10/24 scope global eth0
       valid_lft forever preferred_lft forever
```

sudo nmcli dev set eth0 managed no
sudo ip addr flush dev eth0
sudo ip addr add 192.168.2.10/24 dev eth0
sudo ip link set eth0 up

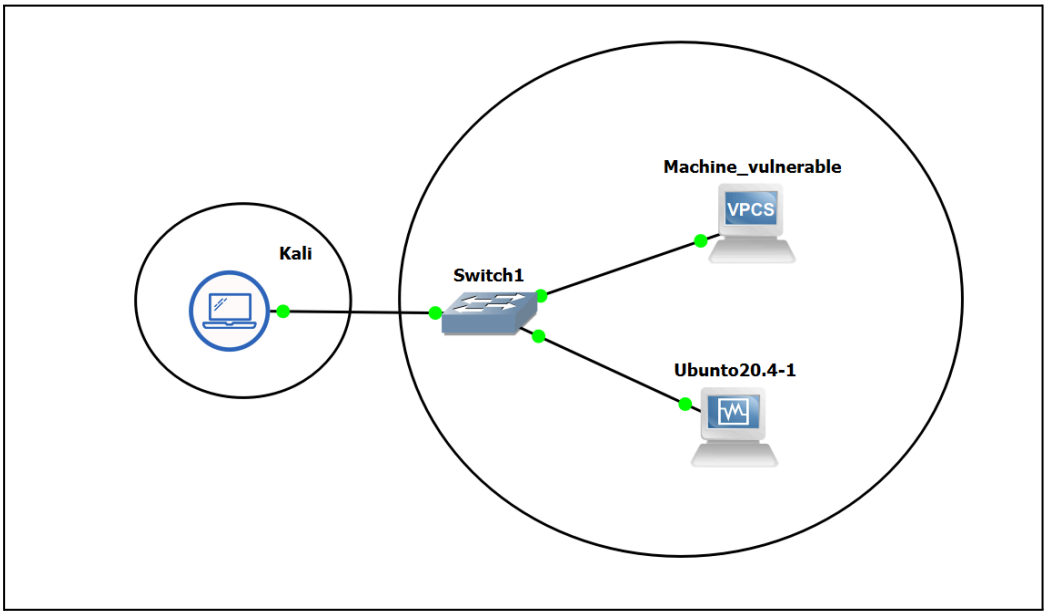
1.3 Mise en place de la machine vulnérable : **192.168.56.30/24**

```
VPCS> ip 192.168.2.30/24
Checking for duplicate address...
PC1 : 192.168.2.30 255.255.255.0

VPCS> save
Saving startup configuration to startup.vpc
.  done

VPCS> show ip

NAME        : VPCS[1]
IP/MASK     : 192.168.2.30/24
GATEWAY     : 0.0.0.0
DNS         :
MAC         : 00:50:79:66:68:00
LPORT       : 10006
RHOST:PORT  : 127.0.0.1:10007
MTU:        : 1500

VPCS>
```

1.4 Installation de la machine Ubuntu Server 20.04 (Snort) : **192.168.56.20/24**

```
ubunto@ubunto-VirtualBox:~$ sudo ip addr flush dev enp0s3
ubunto@ubunto-VirtualBox:~$ sudo ip addr add 192.168.2.20/24 dev enp0s3
ubunto@ubunto-VirtualBox:~$ sudo ip link set enp0s3 up
ubunto@ubunto-VirtualBox:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt 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 g
roup default qlen 1000
    link/ether 08:00:27:70:ba:cc brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.20/24 scope global enp0s3
       valid_lft forever preferred_lft forever
ubunto@ubunto-VirtualBox:~$
```

1.5 Mise en place de la topologie réseau



1.6 Vérification de la connectivité

Des tests de connectivité ont été réalisés à l'aide de la commande ping afin de valider la communication entre les machines.

Machine vulnérable ↔ Kali : OK

```
VPCS> ping 192.168.2.10
84 bytes from 192.168.2.10 icmp_seq=1 ttl=64 time=2.148 ms
84 bytes from 192.168.2.10 icmp_seq=2 ttl=64 time=1.414 ms
84 bytes from 192.168.2.10 icmp_seq=3 ttl=64 time=1.971 ms
84 bytes from 192.168.2.10 icmp_seq=4 ttl=64 time=2.021 ms
84 bytes from 192.168.2.10 icmp_seq=5 ttl=64 time=1.172 ms
```

Ubuntu (Snort) ↔ Kali : OK

```
ubunto@ubunto-VirtualBox:~$ ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
64 octets de 192.168.2.10 : icmp_seq=1 ttl=64 temps=1.57 ms
64 octets de 192.168.2.10 : icmp_seq=2 ttl=64 temps=11.0 ms
64 octets de 192.168.2.10 : icmp_seq=3 ttl=64 temps=1.04 ms
64 octets de 192.168.2.10 : icmp_seq=4 ttl=64 temps=0.984 ms
64 octets de 192.168.2.10 : icmp_seq=5 ttl=64 temps=1.28 ms
64 octets de 192.168.2.10 : icmp_seq=6 ttl=64 temps=5.30 ms
64 octets de 192.168.2.10 : icmp_seq=7 ttl=64 temps=5.22 ms
64 octets de 192.168.2.10 : icmp_seq=8 ttl=64 temps=2.61 ms
64 octets de 192.168.2.10 : icmp_seq=9 ttl=64 temps=5.48 ms
```

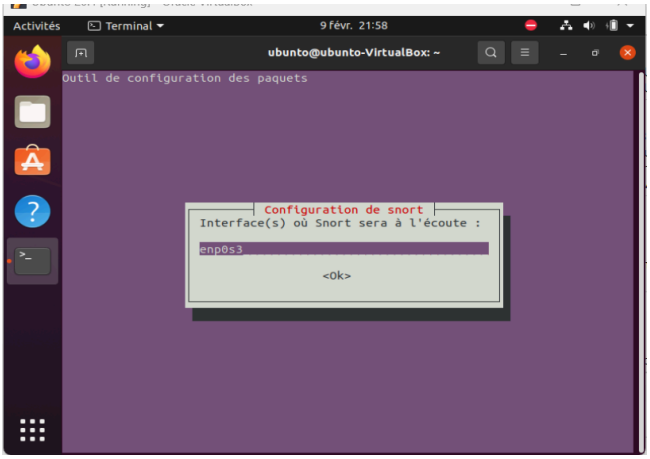
Ubuntu (Snort) ↔ Machine vulnérable : ok

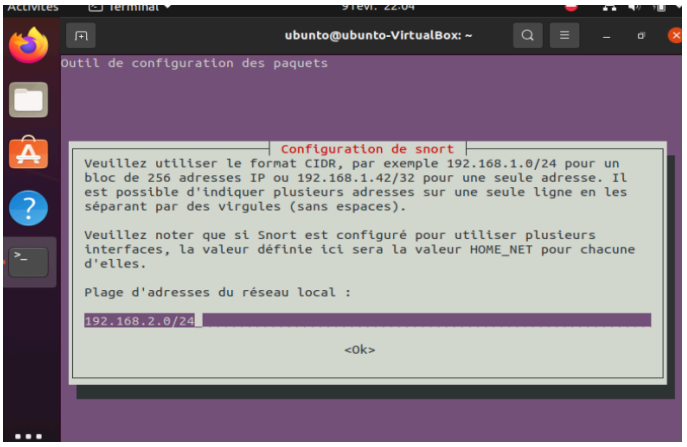
```
ubunto@ubunto-VirtualBox:~$ ping 192.168.2.30
PING 192.168.2.30 (192.168.2.30) 56(84) bytes of data.
64 octets de 192.168.2.30 : icmp_seq=1 ttl=64 temps=2.35 ms
64 octets de 192.168.2.30 : icmp_seq=2 ttl=64 temps=1.84 ms
64 octets de 192.168.2.30 : icmp_seq=3 ttl=64 temps=1.15 ms
64 octets de 192.168.2.30 : icmp_seq=4 ttl=64 temps=1.50 ms
64 octets de 192.168.2.30 : icmp_seq=5 ttl=64 temps=1.91 ms
64 octets de 192.168.2.30 : icmp_seq=6 ttl=64 temps=1.94 ms
64 octets de 192.168.2.30 : icmp_seq=7 ttl=64 temps=1.23 ms
64 octets de 192.168.2.30 : icmp_seq=8 ttl=64 temps=1.65 ms
64 octets de 192.168.2.30 : icmp_seq=9 ttl=64 temps=0.919 ms
64 octets de 192.168.2.30 : icmp_seq=10 ttl=64 temps=1.06 ms
64 octets de 192.168.2.30 : icmp_seq=11 ttl=64 temps=1.47 ms
64 octets de 192.168.2.30 : icmp_seq=12 ttl=64 temps=0.574 ms
```

⇒ Ces tests confirment que la topologie est correctement installée et fonctionnelle.

1.7 Installation et configuration de Snort (IDS) :

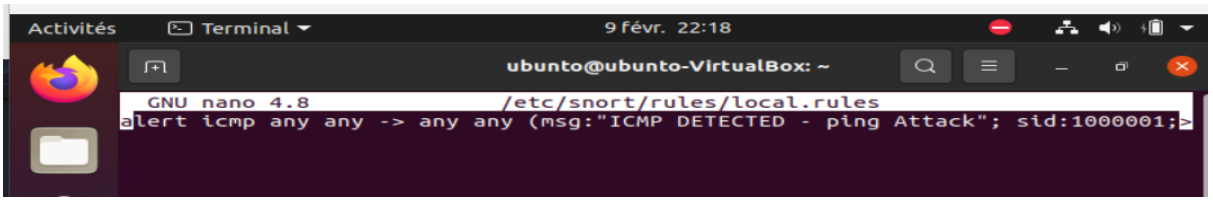
Lors de la configuration de Snort, l'interface enp0s3 a été définie comme interface d'écoute, correspondant à l'interface connectée au réseau interne du laboratoire.





Ajout d'une règle :

```
ubunto@ubunto-VirtualBox:~$ sudo nano /etc/snort/rules/local.rules
```



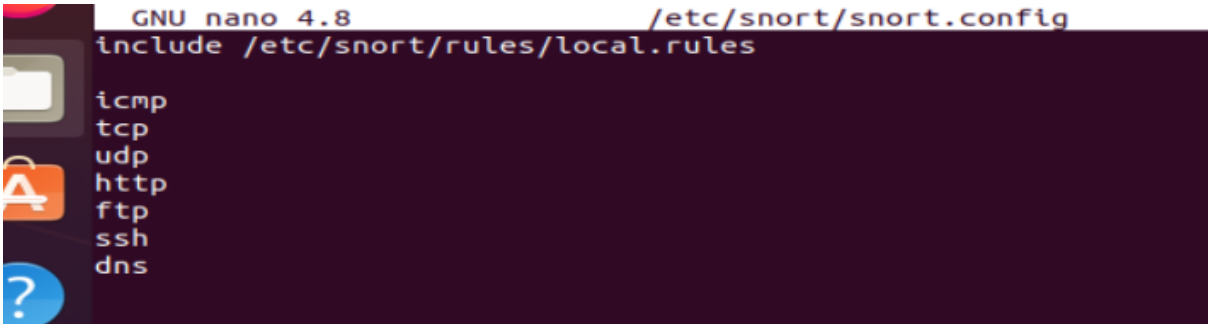
alert icmp any any -> any any (msg:"ICMP DETECTED - Ping Attack"; sid:1000001; rev:1;)

alert tcp any any -> any any (flags:S; msg:"NMAP Scan Detected - SYN"; sid:1000002; rev:1;)

⇒ Une règle ICMP personnalisée a été ajoutée à Snort, permettant la détection des requêtes ping envoyées depuis la machine Kali Linux vers la machine vulnérable.

2. Consultation des règles de détection dans le fichier snort.lua :

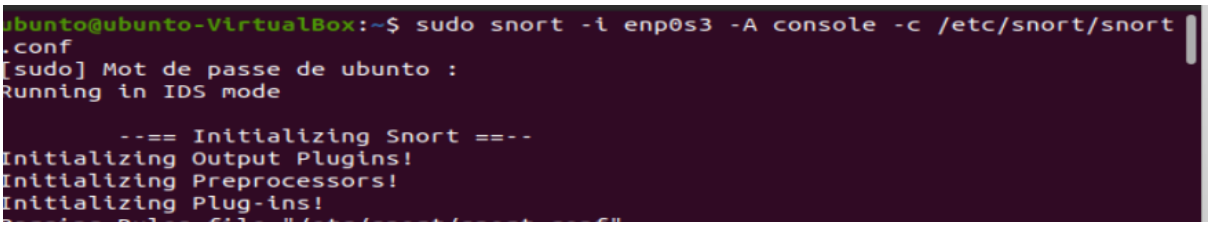
```
ubunto@ubunto-VirtualBox:~$ sudo nano /etc/snort/snort.config
ubunto@ubunto-VirtualBox:~$
```



3. Tests d'attaques Kali Linux et détection par Snort

3.1 Lancement de Snort en mode détection

Sur la machine Ubuntu (Snort), Snort est lancé manuellement afin d'analyser le trafic réseau en temps réel sur l'interface connectée au réseau interne :



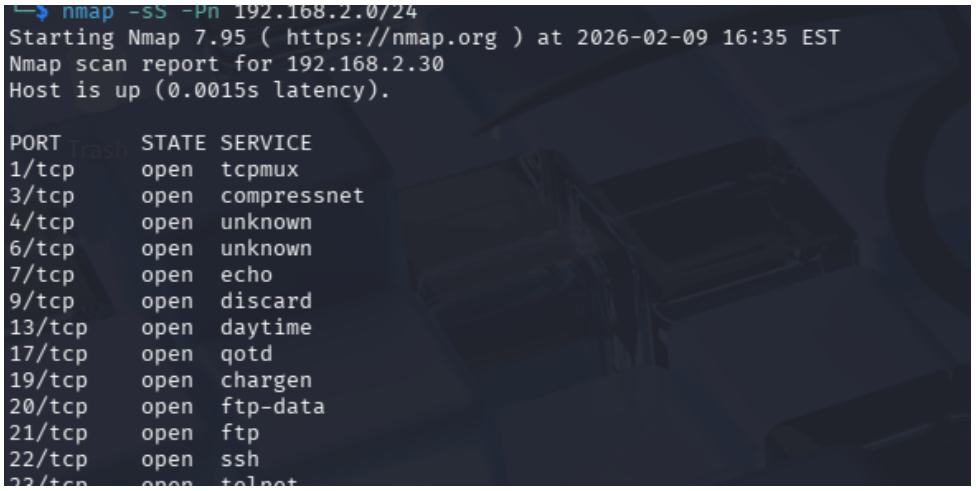
3.2 Outils d'attaque utilisés sur Kali Linux

Kali Linux dispose de nombreux outils dédiés aux tests d'intrusion. Dans ce TP, l'outil suivant a été utilisé :

- Nmap : outil de scan réseau permettant d'identifier les hôtes actifs, les ports ouverts, les services et les vulnérabilités potentielles.

3.3 Scan réseau avec Nmap (scan global)

Depuis la machine Kali Linux, la commande suivante a été exécutée :



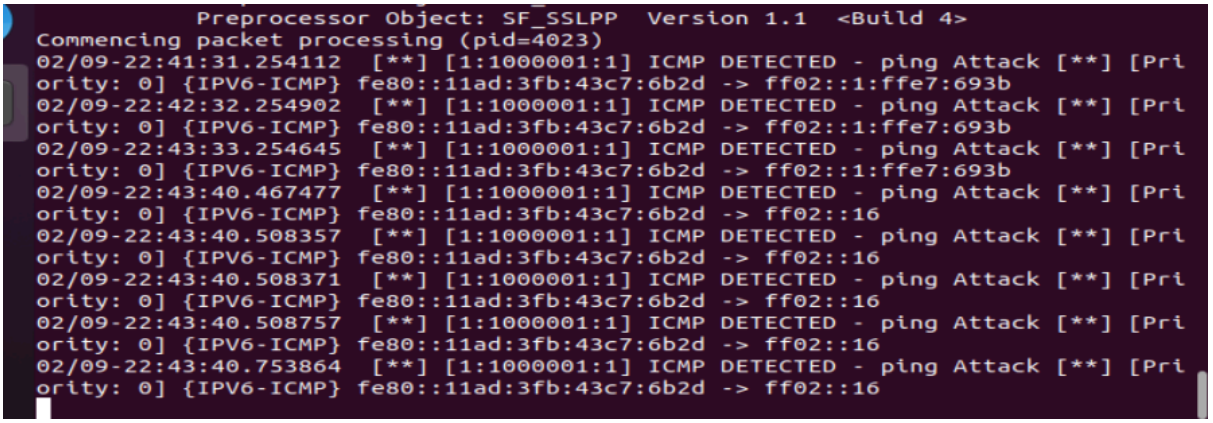
Explication de la commande	Résultat
• nmap : outil de scan réseau • -sS : scan SYN (scan furtif, semi-ouvert) • -Pn : désactive la phase de découverte par ping • 192.168.2.0/24 : cible l'ensemble du réseau interne	- Détection des hôtes actifs sur le réseau - Identification des ports ouverts sur les machines ciblées

3.4 Scan ciblé avec détection de vulnérabilités

nmap 192.168.2.30 -sV --script vuln

Explication de la commande	Résultat
• -sV : détection des versions des services actifs • --script vuln : exécution de scripts Nmap dédiés à la recherche de vulnérabilités • 192.168.2.30 : machine vulnérable ciblée	- Identification des services en cours d'exécution - Détection de potentielles failles de sécurité

Lors de ce scan avancé, Snort a généré plusieurs alertes:



⇒ Les tests d'attaques réalisés depuis Kali Linux ont été correctement détectés par Snort. Les scans Nmap, qu'ils soient globaux ou ciblés, ont généré des alertes en temps réel, confirmant le bon fonctionnement du système de détection d'intrusion.