

PROJET : DETECTION ET BLOCAGE D'ATTAQUES

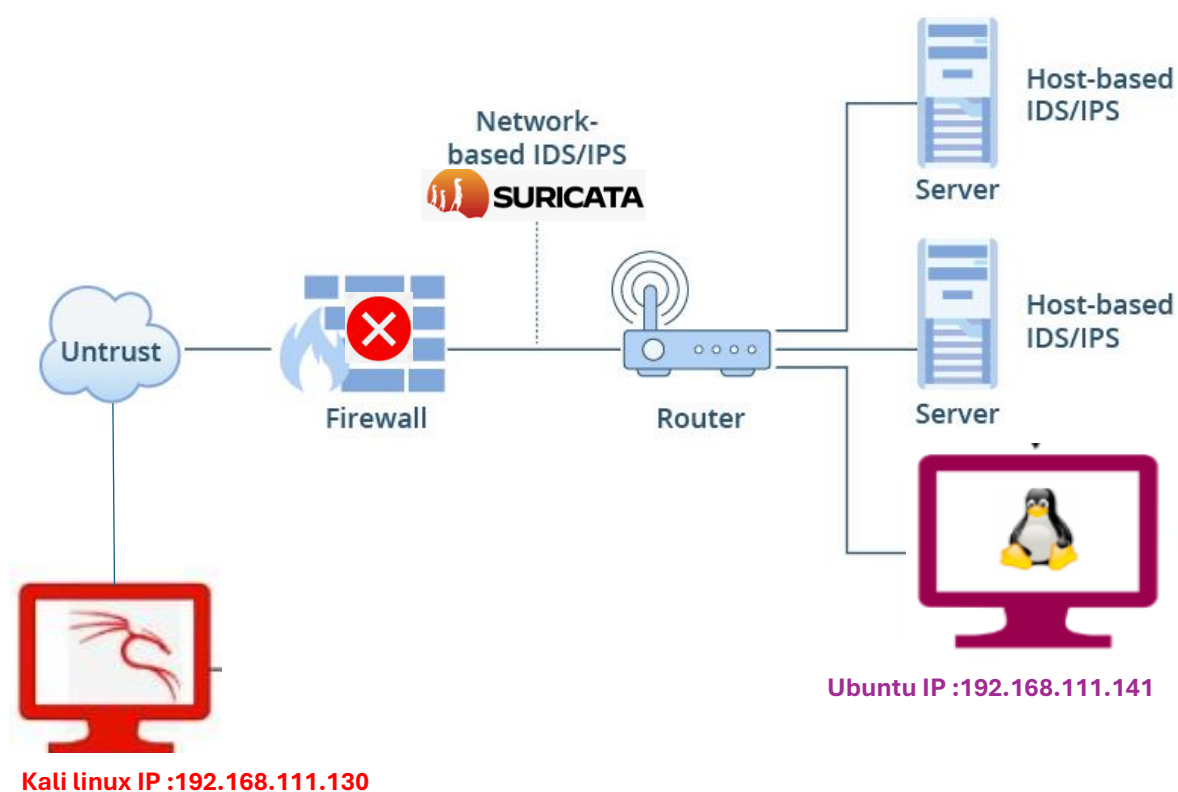
AVEC SURICATA IDS/IPS



Dans ce projet nous allons tenter de mieux comprendre les outils de détection d'attaque et blocage de certaines attaques sur les réseaux informatiques ainsi que des tentatives de récupérations d'informations (scan de réseau avec nmap).

Quelques scenarii d'attaques/scans seront étudiés à savoir du scanning réseaux avec nmap, une tentative d'attaque type brute force, des pings (protocole ICMP) pour savoir une machine répond une sollicitation d'une autre machine inconnue du réseau.

Pour mener à bien ce projet nous utiliserons SURICATA comme IDS (systèmes de détection des intrusions) dans un premier temps pour détecter les scans puis comme IPS (systèmes de prévention des intrusions) dans un second temps pour bloquer les attaques.



NB : Firewall déconnecté dans cette architecture

1 . C'est quoi un IDS/IPS

Les systèmes de détection des intrusions (IDS) analysent le trafic réseau pour détecter des signatures correspondant à des cyberattaques connues.

Les systèmes de prévention des intrusions (IPS) analysent également les paquets, mais ils peuvent aussi les bloquer en fonction du type d'attaques qu'ils détectent, ce qui contribue à stopper ces attaques.

2. Fonctionnement d'un IDS/IPS

Les IDS et les IPS font tous deux parties de l'infrastructure réseau. Les IDS/IPS comparent les paquets de réseau à une base de données de cybermenaces contenant des signatures connues de cyberattaques et repèrent tous les paquets qui concordent avec ces signatures.

La principale différence entre les deux tient au fait que l'IDS est un système de surveillance, alors que l'IPS est un système de contrôle.

L'IDS ne modifie en aucune façon les paquets réseau, alors que l'IPS empêche la transmission du paquet en fonction de son contenu, tout comme un pare-feu bloque le trafic en se basant sur l'adresse IP.

2.a. Les IDS (Intrusion Detection Systems)

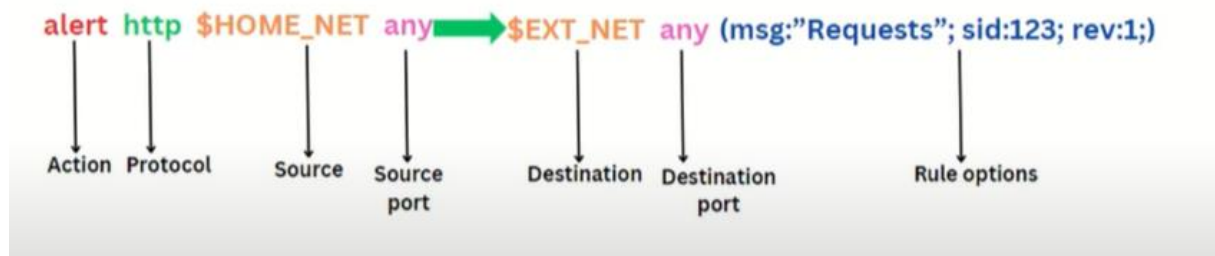
Ils analysent et surveillent le trafic réseau pour détecter des signes indiquant que des hackers utilisent une cybermenace connue afin de s'infiltrer dans votre réseau ou y voler des données. Les systèmes d'IDS comparent l'activité réseau en cours avec une base de données d'attaques connues afin de détecter divers types de comportements tels que les violations de la politique de sécurité, les malwares et les scanners de port.

2.b. Les IPS (Intrusion Prevention Systems)

Ils agissent dans la même zone du réseau qu'un pare-feu, entre le monde extérieur et le réseau interne. Les IPS rejettent de façon proactive les paquets réseau en fonction d'un profil de sécurité si ces paquets représentent une menace connue.

3. Structure des règles de détection et de blocages dans SURICATA

Une règle SURICATA se présente d'une façon générale sous la forme suivante avec plusieurs options :



Une règle/signature se compose des éléments suivants :

L'action, qui détermine ce qui se passe lorsque la règle correspond.

L'en-tête, définissant le protocole, les adresses IP, les ports et la direction la règle.

Les options de la règle, qui définissent les spécificités de la règle.

*ACTION

L'action détermine la réponse à un match :

alert - Générer une alerte.

pass - Arrêter l'inspection ultérieure du paquet.

drop - supprimer le paquet et génerez une alerte.

reject - envoyer l'erreur RST/ICMP unreachable à l'expéditeur du paquet correspondant.

*PROTOCOLE

Cette section spécifie le protocole réseau à surveiller, tel que TCP, UDP, ICMP, etc. Il définit le type de trafic ciblé par la règle.

*SOURCE ET DESTINATION

La source et la destination permettent de spécifier la source du trafic « \$HOME_NET » et la destination du trafic « \$EXTERNAL_NET » respectivement. Sans oublier la flèche directionnelle.

*PORTS (source et destination)

Ces paramètres limitent la règle à des flux de données spécifiques en fonction des ports utilisés dans le trafic réseau.

*OPTIONS DE RÈGLES

Le reste de la règle se compose d'options. Ceux-ci sont entourés de parenthèses et séparés par des points-virgules. Certaines options ont des paramètres, qui sont spécifiés par le mot-clé de l'option, suivi d'un deux-points, suivi des paramètres.

4. Mise en œuvre des scénarii d'attaques et de scan réseau

4.1 . Scénario 1 : Détection d'un ping ICMP

Dans ce scénario nous allons lancer des pings du protocole ICMP depuis une machine Kali linux vers un poste Ubuntu client. Nous verrons si la machine cliente répond aux pings et élaborons une règle avec SURICATA pour nous alerter.

Pour détecter un PING (protocole ICMP) lancer par la machine kali linux, nous avons créé la règle SURICATA suivante :

```
alert icmp any any -> $HOME_NET any (msg:"DETECTION DE PING ICMP!!" ;flow:established; priority:1 ;rev:1 ;sid:10001;)
```

Action (alerte) : l'action spécifiée est « alert », indiquant que lorsqu'une correspondance est trouvée avec les conditions définies, une alerte doit être générée.

Protocole (icmp) : La règle s'applique spécifiquement à l'Internet Control Message Protocol (ICMP). ICMP est souvent utilisé pour les diagnostics réseau, y compris les messages ping.

Message (msg : « DETECTION DE PING ICMP !! ») : Le message à afficher lorsqu'une correspondance est trouvée est « DETECTION DE PING ICMP !! ». Cela donne une indication claire de la nature de l'alerte.

sid : 10001 : chaque règle SURICATA est associée à un identificateur de signature unique (sid). Dans ce cas, l'identifiant est défini comme 10001. Cela permet d'identifier de manière unique cette règle au sein d'un ensemble de règles.

rev :1 : désigne la révision de la règle (rev) indique la version de la règle. Dans cet exemple, la règle a la révision 1.

Flow :established :Correspondance sur les connexions établies.

4.1.1. RESULTAT

Les captures d'écran ci-dessous montrent que la règle SURICATA créée pour détecter un PING fonctionne. Nous pouvons également y voir les adresses IP de la Kali linux qui lance les ping (192.168.111.130) et la machine hôte Ubuntu 192.168.111.141.

```
kali@kali: ~  
File Actions Edit View Help  
root@kali: /home/kali x kali@kali: ~ x  
^C  
— 192.168.111.141 ping statistics —  
58 packets transmitted, 58 received, 0% packet loss, time 58338ms  
rtt min/avg/max/mdev = 0.376/0.521/1.003/0.104 ms  
  
(kali@kali)-[~]  
$ ping 192.168.111.141  
PING 192.168.111.141 (192.168.111.141) 56(84) bytes of data.  
64 bytes from 192.168.111.141: icmp_seq=1 ttl=64 time=0.443 ms  
64 bytes from 192.168.111.141: icmp_seq=2 ttl=64 time=0.448 ms  
64 bytes from 192.168.111.141: icmp_seq=3 ttl=64 time=0.483 ms  
64 bytes from 192.168.111.141: icmp_seq=4 ttl=64 time=0.532 ms  
64 bytes from 192.168.111.141: icmp_seq=5 ttl=64 time=0.398 ms  
64 bytes from 192.168.111.141: icmp_seq=6 ttl=64 time=0.476 ms  
64 bytes from 192.168.111.141: icmp_seq=7 ttl=64 time=0.516 ms  
64 bytes from 192.168.111.141: icmp_seq=8 ttl=64 time=0.548 ms
```

```
Open [ ] custom.rules [Read-Only] Save [ ] - [ ] X
/var/lib/suricata/rules

1 alert icmp any any -> $HOME_NET any {msg:"DETECTION DE PING ICMP!!";flow:established; priority:1 ;rev:1 ;sid:10001;}

mdiaw2@mdiaw2: /var/log/suricata

mdiaw2@mdiaw2:~$ cd /var/log/suricata
mdiaw2@mdiaw2: /var/log/suricata$ ls
certs  eve.json  files      suricata.log
core  fast.log  stats.log  suricata-start.log
mdiaw2@mdiaw2: /var/log/suricata$ sudo tail -f /var/log/suricata/fast.log
[sudo] password for mdiaw2:

^C
mdiaw2@mdiaw2: /var/log/suricata$ sudo tail -f /var/log/suricata/fast.log
^C
mdiaw2@mdiaw2: /var/log/suricata$ sudo tail -f /var/log/suricata/fast.log
[sudo] password for mdiaw2:
01/07/2025-15:24:02.148549 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.141:0 -> 192.168.111.130:0
01/07/2025-15:24:03.150501 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.141:0 -> 192.168.111.130:0
01/07/2025-15:24:03.150501 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.141:0 -> 192.168.111.130:0
01/07/2025-15:24:04.174483 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.130:8 -> 192.168.111.141:0
01/07/2025-15:24:04.174530 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.141:0 -> 192.168.111.130:0
01/07/2025-15:24:05.198813 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.130:8 -> 192.168.111.141:0
01/07/2025-15:24:05.198853 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.141:0 -> 192.168.111.130:0
01/07/2025-15:24:06.220543 10001:1 DETECTION DE PING ICMP!! [Classification: (null)] [Priority: 1] {ICMP} 192.168.111.141:0 -> 192.168.111.130:0
```

4.2 . Scenario 2 : Détection d'un scan avec nmap

Dans ce scenario nous lancer un scan de réseau avec nmap depuis notre kali linux vers la machine hôte Ubuntu.

L'objectif est de créer une règle SURICATA capable de nous alerter (sans bloquer) qu'une tentative de scan de la machine Ubuntu est en cours.

Pour détecter un scan nmap (flag SYN (S)) lancer par la machine kali linux, nous avons créé la règle SURICATA suivante :

```
alert tcp any any -> 192.168.111.141 any (msg:"DETECTION D'UNE TENTATIVE DE SCAN DE PORTS!!" ;flow:established;flags:S; priority:1 ;rev:1 ;sid:10003;)
```

flags : S : Spécifie les drapeaux TCP à surveiller, tels que SYN, FIN, PUSH, RESET, URG, ACK, etc. Dans notre exemple, la règle détecte les paquets TCP avec l'indicateur SYN défini. L'indicateur TCP SYN est utilisé pour initier une connexion TCP en envoyant une demande de synchronisation au point de terminaison distant.

4.2.1. RESULTAT

Les captures d'écran ci-dessous montrent que la règle SURICATA créée pour détecter un scan réseau avec nmap fonctionne.

```
(root@kali)-[/home/kali]
# nmap -sS 192.168.111.141
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-08 10:05 EST
Nmap scan report for 192.168.111.141
Host is up (0.00019s latency).
All 1000 scanned ports on 192.168.111.141 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 00:0C:29:94:96:1F (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
```

```
mdiaw2@mdiaw2: /var/log/suricata

[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 91.189.91.98:80 -> 192.168.111.141:5
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 91.189.91.98:80 -> 192.168.111.141:5
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 91.189.91.98:80 -> 192.168.111.141:5
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 91.189.91.98:80 -> 192.168.111.141:5
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 91.189.91.98:80 -> 192.168.111.141:5
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 185.125.190.49:80 -> 192.168.111.141
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 185.125.190.49:80 -> 192.168.111.141
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 185.125.190.49:80 -> 192.168.111.141
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 185.125.190.49:80 -> 192.168.111.141
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 185.125.190.49:80 -> 192.168.111.141
[1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 185.125.190.49:80 -> 192.168.111.141
```

4.3. Scenario 3 : Détection et blocage d'une attaque type brute force

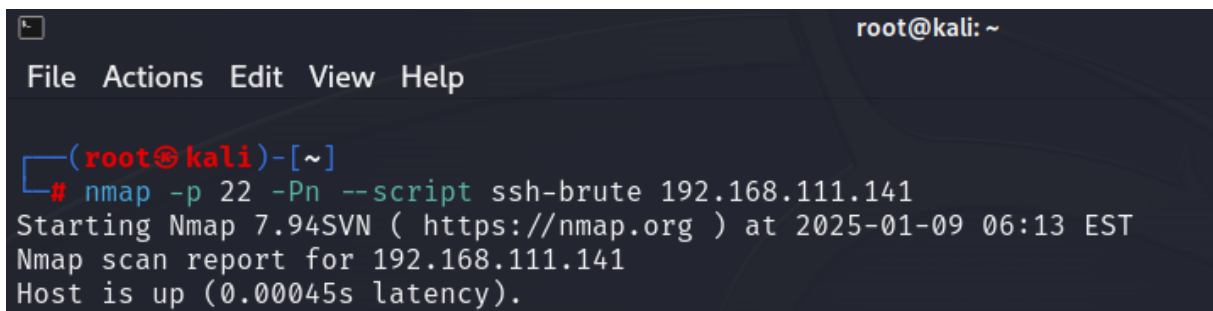
Dans ce dernier scenario nous allons utiliser SURICATA comme un IPS. Nous tenterons de bloquer une attaque de type brute force lancer depuis notre machine Kali linux vers la machine Ubuntu cliente.

Afin de détecter et d'éliminer l'attaque par brute force, nous avons élaborer la règle SURICATA suivante :

```
drop tcp any any -> $HOME_NET 22 (msg:"DROP INCOMING SSH BRUTE-FORCE ATTACK !!" ;flow:established; priority:1 ;rev:1 ;sid:10004;)
```

4.3.1. RESULTAT

Les captures d'écran ci-dessous montrent que la règle SURICATA créée ci-dessus détecte bien et supprime l'attaque par brute force via SSH.



```
root@kali: ~
File Actions Edit View Help
(root@kali)-[~]
# nmap -p 22 -Pn --script ssh-brute 192.168.111.141
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-09 06:13 EST
Nmap scan report for 192.168.111.141
Host is up (0.00045s latency).
```

```
[Drop] [**] [1:10004:1] DROP INCOMING SSH BRUTE-FORCE ATTACK !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.130:54484 -> 192.168.111.141:22
```

ANNEXES

```
mdiaw2@mdiaw2: ~  
mdiaw2@mdiaw2:~$ sudo iptables -I INPUT -p tcp -j NFQUEUE --queue-num 2  
[sudo] password for mdiaw2:  
mdiaw2@mdiaw2:~$ sudo iptables -I OUTPUT -p tcp -j NFQUEUE --queue-num 2  
mdiaw2@mdiaw2:~$ sudo iptables -vnL  
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)  
  pkts bytes target     prot opt in     out     source               destination  
      0      0 NFQUEUE   tcp  --  *      *       0.0.0.0/0            0.0.0.0/0  
      0      0 NFQUEUE num 2  
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  
      0      0 NFQUEUE   tcp  --  *      *       0.0.0.0/0            0.0.0.0/0  
      0      0 NFQUEUE num 2  
mdiaw2@mdiaw2:~$
```

```
32:50.190929 [Drop] [**] [1:10004:1] DROP INCOMING SSH BRUTE-FORCE ATTACK !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.130:54484 -> 192.168.111.141:22  
32:52.359533 [**] [1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.130:54484 -> 192.168.111.141:22  
32:52.359569 [Drop] [**] [1:10004:1] DROP INCOMING SSH BRUTE-FORCE ATTACK !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.130:54484 -> 192.168.111.141:22  
33:41.902678 [**] [1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.141:22 -> 192.168.111.130:54484  
33:41.903145 [**] [1:10003:1] DETECTION D'UNE TENTATIVE DE SCAN DE PORTS !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.130:54484 -> 192.168.111.141:22  
33:41.903222 [Drop] [**] [1:10004:1] DROP INCOMING SSH BRUTE-FORCE ATTACK !! [**] [Classification: (null)] [Priority: 1] {TCP} 192.168.111.130:54484 -> 192.168.111.141:22
```