

Snort - Détection d'intrusion (IDS)

Lien supplémentaire : CNIL ... <https://all-it-network.com/snort/>

Ce laboratoire a été développé pour le cadre Labtainer par le Naval Postgraduate School, Center for Cybersecurity and Cyber Operations dans le cadre du National Science Foundation Award No. 1438893.

Ce site travail est dans le domaine public, et ne peut être protégé par des droits d'auteur.

Insert text

1 Présentation

Cet exercice présente l'utilisation du système snort pour fournir une détection d'intrusion dans un environnement Linux. Les étudiants configureront des règles snort simples et expérimenteront avec un système de détection d'intrusion réseau (IDS).

2 Environnement du laboratoire

Ce laboratoire fonctionne dans l'environnement Labtainer, disponible sur <http://my.nps.edu/web/c3o/labtainers>. Ce site comprend des liens vers une machine virtuelle préconstruite sur laquelle Labtainer est installé, mais Labtainer peut être exécuté sur n'importe quel hôte Linux prenant en charge Docker.

À partir de votre répertoire labtainer-student, démarrez le laboratoire en utilisant :

```
export DISPLAY=:0
labtainer snort
```

Un lien vers ce manuel de laboratoire s'affichera

3 Configuration du réseau

Ce laboratoire comprend plusieurs ordinateurs en réseau comme le montre la **Figure 1**. Lorsque le laboratoire démarre, vous obtenez plusieurs terminaux virtuels, un connecté à chaque composant.

La passerelle est configurée avec iptables pour utiliser NAT pour traduire les adresses sources du trafic à partir des adresses IP internes, par exemple **192.168.2.1**, vers notre adresse externe, c'est-à-dire **203.0.113.10**. Les règles iptables de la passerelle redirigent également le trafic web (ports 80 et 443) vers le composant serveur web en traduisant l'adresse de destination visible de l'extérieur en l'adresse interne du serveur web.

La passerelle est également configurée pour refléter le trafic qui entre dans la passerelle via le lien 203.0.113.10, ou le lien vers le serveur web. Ce trafic miroir est acheminé vers le composant snort. Cette mise en miroir permet au composant **snort** de reconstruire les sessions TCP entre le serveur web et les adresses externes.

Le composant snort comprend l'utilitaire Snort IDS. Il inclut également Wireshark pour vous aider à observer le trafic mis en miroir vers le composant snort.

Le serveur Web exécute Apache et est configuré pour prendre en charge SSL pour les pages Web du domaine www.example.com.

Le composant remote_ws comprend le navigateur Firefox, ainsi qu'un fichier /etc/hosts local qui mappe www.example.com à l'adresse externe de la passerelle, c'est-à-dire 203.0.113.10. La station de travail interne (ws2) comprend également Firefox et une entrée dans /etc/hosts pour www.example.com. Les deux stations de travail l'utilitaire nmap.

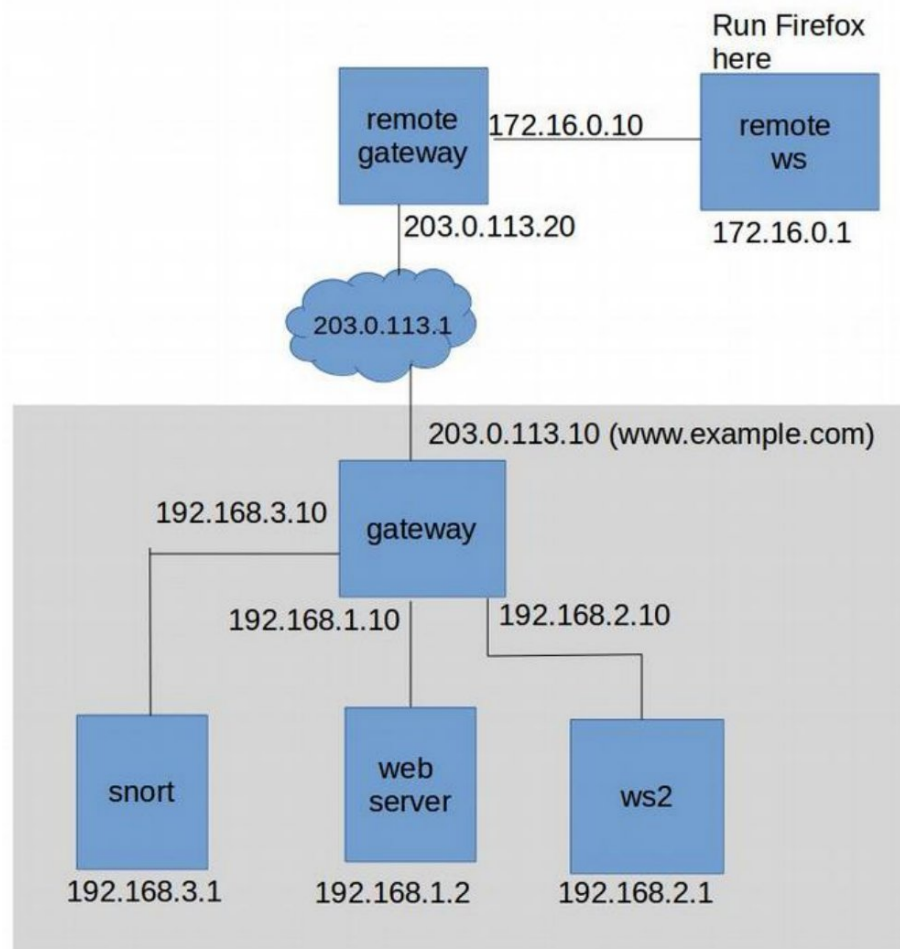


Figure 1 : topologie du réseau pour le laboratoire snort

4 Tâches de laboratoire

Il est supposé que l'étudiant a reçu un enseignement ou une étude indépendante sur le fonctionnement de base de Snort, ainsi que sur les objectifs généraux et les mécanismes de la détection des intrusions dans le réseau.

Passez en revue la topologie du réseau. En particulier, considérez les paramètres iptables sur la passerelle. Ceux-ci peuvent être vus en examinant les commandes dans `/etc/rc.local`, qui sont utilisés pour définir les traductions NAT et, ce qui est essentiel pour ce laboratoire, à refléter le trafic vers le composant snort.

Liens vers une ressource expliquant le port mirroring avec iptables :

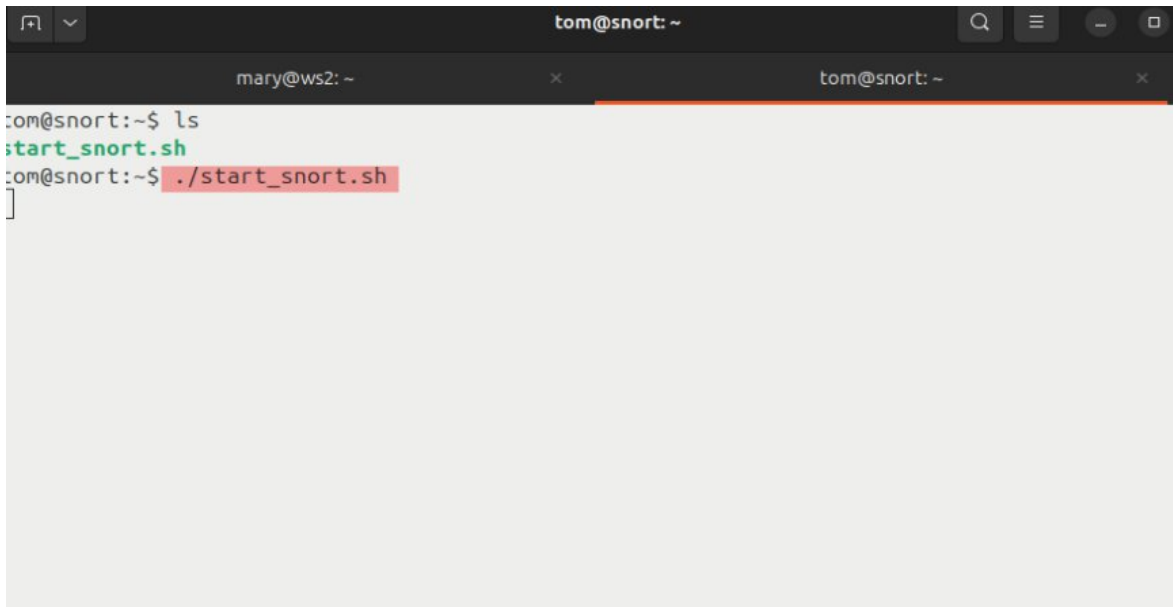
<https://www.it-connect.fr/port-mirroring-sous-linux-avec-iptables/>

4.1 Démarrer et arrêter snort

L'utilitaire Snort est installé sur le composant snort. Le répertoire d'accueil comprend un script `start_snort.sh` qui démarre l'utilitaire en mode de détection d'intrusion réseau et affiche les alertes sur la console. Pour ce laboratoire, vous devez démarrer snort avec :

`./start_snort.sh`

Lorsque le moment est venu d'arrêter snort, par exemple pour ajouter des règles, utilisez simplement CTRL+C.



A terminal window titled 'tom@snort: ~' with two tabs: 'mary@ws2: ~' and 'tom@snort: ~'. The active tab shows the following commands and output:

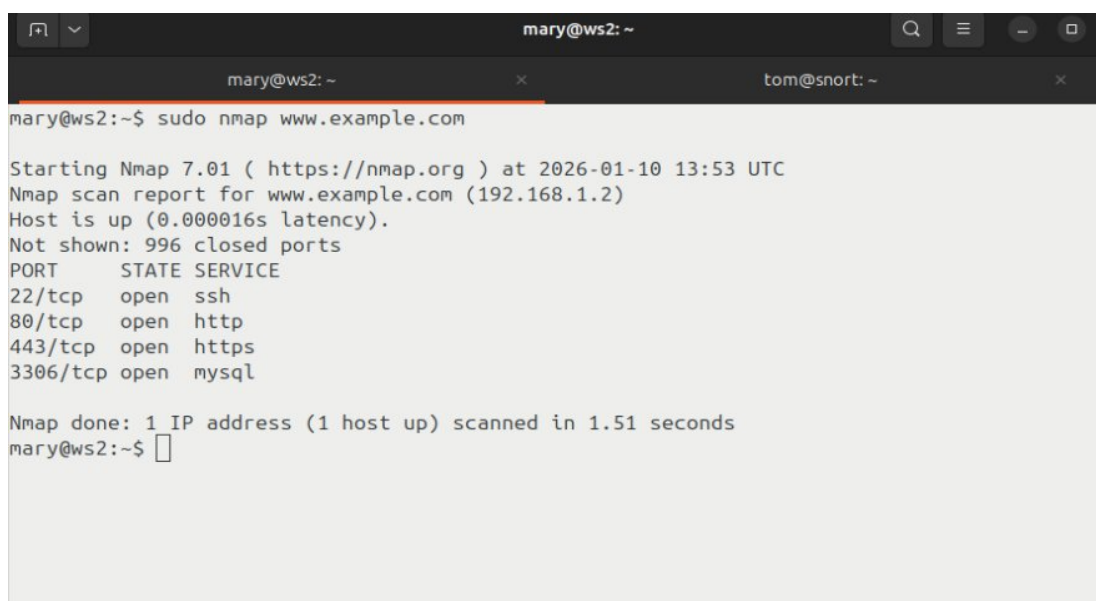
```
tom@snort:~$ ls
start_snort.sh
tom@snort:~$ ./start_snort.sh
```

Démarrage du service Snort sur la machine Snort

4.2 Règles Snort préconfigurées

L'utilitaire Snort comprend un ensemble de règles préconfigurées qui créent des alertes pour les activités réseau suspectes connues. La configuration du composant snort est en grande partie celle qui existe après l'installation initiale de l'utilitaire snort. Pour voir un exemple de certaines des règles préconfigurées, effectuez un scan nmap de www.example.com à partir de la station de travail distante :

`sudo nmap www.example.com`



A terminal window titled 'mary@ws2: ~' with two tabs: 'mary@ws2: ~' and 'tom@snort: ~'. The active tab shows the following commands and output:

```
mary@ws2:~$ sudo nmap www.example.com

Starting Nmap 7.01 ( https://nmap.org ) at 2026-01-10 13:53 UTC
Nmap scan report for www.example.com (192.168.1.2)
Host is up (0.000016s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https
3306/tcp   open  mysql

Nmap done: 1 IP address (1 host up) scanned in 1.51 seconds
mary@ws2:~$
```

Scan Nmap lancé depuis la station distante (terminal remote_ws) afin de générer du trafic analysé par Snort.

```
tom@snort:~$ ./start_snort.sh
01/10-13:53:25.515712  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity]
[Priority: 3] {ICMP} 192.168.1.2 -> 192.168.2.1
01/10-13:53:25.515836  [**] [1:451:5] ICMP Timestamp Reply [**] [Classification: Misc activ
ity] [Priority: 3] {ICMP} 192.168.1.2 -> 192.168.2.1
```

Alertes générées par Snort sur le terminal snort suite au scan Nmap effectué depuis la station ws2 (mary).

Notez les alertes affichées sur la console snort. Les règles qui génèrent ces alertes sont visibles, ainsi que toutes les règles, dans le fichier /etc/snort/rules/.

4.3 Écrire une (mauvaise) règle simple

Les règles personnalisées sont généralement ajoutées au fichier /etc/snort/rules/local.rules. une règle qui génère une alerte pour chaque paquet dans un flux TCP. Par exemple :

```
alert tcp any any -> any any (msg: "TCP detected" ; sid:00002 ;)
```

Cette règle peut être lue comme suit : "Générer une alerte chaque fois qu'un paquet TCP provenant de n'importe quelle adresse sur n'importe quel port est envoyé à n'importe quelle adresse sur n'importe quel port, et afficher le message 'TCP detected', et donner à la règle un identifiant 00002."

```
GNU nano 2.5.3      File: /etc/snort/rules/local.rules      Modified
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures.  Put your local
# additions here.

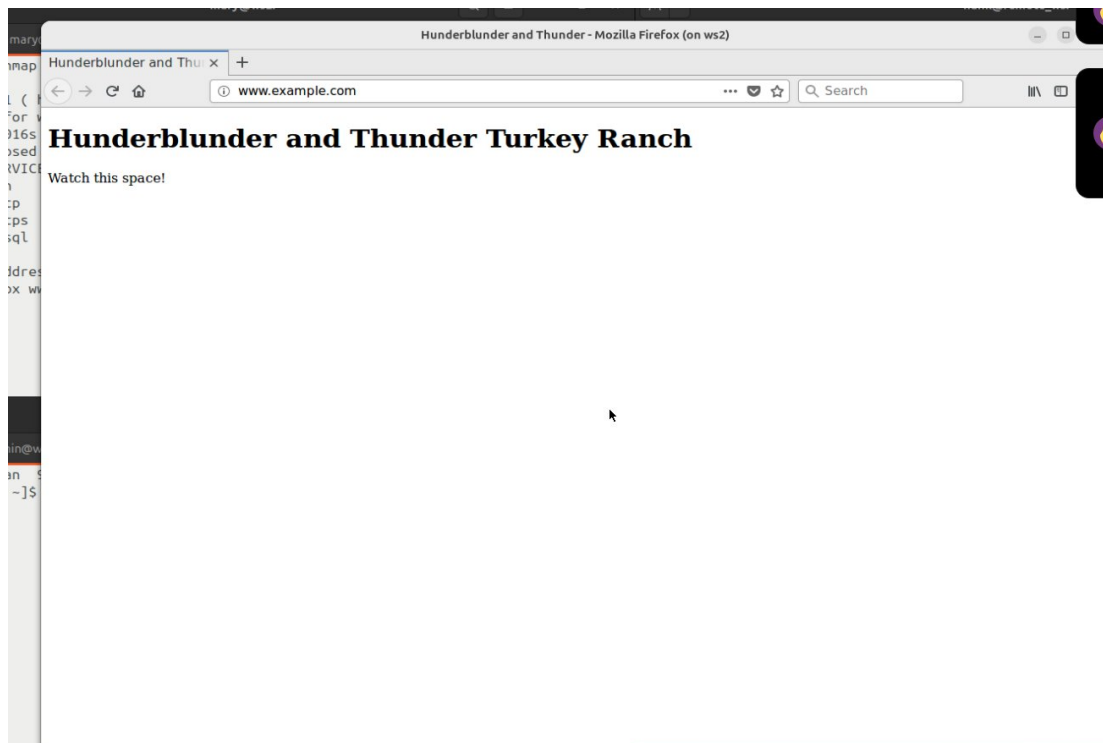
alert tcp any any -> any any (msg:"TCP detected"; sid:00002;)

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify     ^C Cur Pos
^X Exit          ^R Read File    ^\ Replace      ^U Uncut Text   ^T To Spell    ^_ Go To Line
```

Ajout d'une règle Snort personnalisée trop permissive dans le fichier local.rules (terminal snort).

Redémarrez ensuite snort. Testez cette règle en démarrant Firefox sur la station de travail distante :

firefox www.example.com



Firefox ouvert sur www.example.com depuis la station distante afin de générer du trafic TCP.

Comme vous pouvez le constater, la règle que vous avez écrite va vous submerger d'informations inutiles. Donc, arrêtez snort et supprimez la règle.

```
01/10-14:02:27.963860 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.963181 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.963321 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.963519 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.963628 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.963802 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964028 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964159 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964356 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964568 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964640 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964797 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.964972 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.965111 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.965293 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.965473 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.965750 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.965910 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.966053 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.966221 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.966391 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.966561 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.966734 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.966900 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.967057 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.967418 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.967610 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.967631 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.967836 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.967968 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.968469 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.968485 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970098 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970158 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970306 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970342 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970483 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970705 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.970842 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.971014 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.971175 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.971338 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.971546 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
01/10-14:02:27.971662 [**] [1:2:0] TCP detected [**] [Priority: 0] {TCP} 172.217.18.202:443 -> 203.0.113.10:35526
```

Beaucoup d'alertes générées par Snort à causé d'une règle trop générale (TCP detected)

4.4 Règle personnalisée pour le trafic CONFIDENTIEL

Au niveau du navigateur Firefox, qui devrait afficher la page web de www.example.com, nous allons afficher une page web non publiée dont nous savons qu'elle existe sur le site web. En particulier, nous avons entendu dire que l'entreprise a placé des plans confidentiels à l'adresse www.example.com/plan.html.

Jetez-y un coup d'œil.

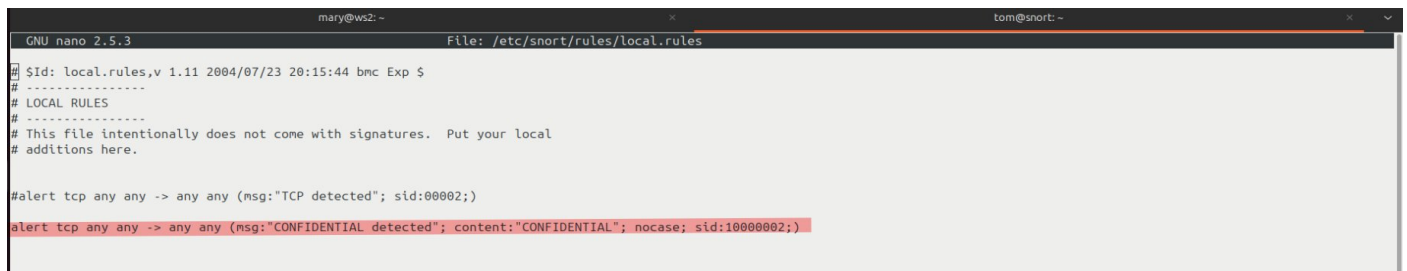
Ajoutez maintenant une règle à votre fichier `local.rules` de snort qui générera une alerte chaque fois que le texte "CONFIDENTIAL" sera envoyé sur Internet. Référez-vous au manuel de snort

<https://www.snort.org/documents/snort-users-manual> (Payload Detection Rule Options) ou aux règles existantes pour comprendre comment qualifier les alertes en fonction de leur contenu. Veillez à inclure le mot "CONFIDENTIAL" dans le message d'alerte, et donnez à la règle un sid unique. Après avoir ajouté la règle, redémarrez snort.

Sur le navigateur du poste de travail distant, effacez l'historique (Menu / Préférences Sécurité et confidentialité), puis rafraîchissez la page `plan.html`. Vous devriez voir une alerte dans la console snort.

```
alert tcp any any -> any any (msg:"CONFIDENTIAL detected"; content:"CONFIDENTIAL"; nocase; sid:10000002;)
```

`nocase` signifie insensible à la casse



```
GNU nano 2.5.3 File: /etc/snort/rules/local.rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures. Put your local
# additions here.

#alert tcp any any -> any any (msg:"TCP detected"; sid:00002;)
alert tcp any any -> any any (msg:"CONFIDENTIAL detected"; content:"CONFIDENTIAL"; nocase; sid:10000002;)
```

Ajout d'une règle Snort personnalisée détectant le mot « CONFIDENTIAL » dans les flux TCP (terminal snort).

4.5 Effets du chiffrement

De retour au navigateur Firefox, effacez à nouveau l'historique du navigateur. Modifiez maintenant l'URL pour utiliser la fonction SSL du serveur web. Changez l'URL en <https://www.example.com/plan.html>. Voyez-vous une nouvelle alerte snort ?

Pourquoi ?

On efface l'historique (cache) du navigateur pour obliger Firefox à recharger la page depuis le serveur web et ainsi générer du trafic réseau analysable par Snort

Une solution à ce problème consiste à utiliser un proxy inverse devant le serveur Web. Ce reverse proxy va gérer le trafic entrant et gérer les connexions SSL. Le serveur Web ne recevrait alors que du trafic HTTP en clair, et le trafic sortant du serveur web pourrait alors être reflété vers l'IDS. Nous ne poursuivons pas cette solution dans ce laboratoire

4.6 Surveillance du trafic interne

Allez sur le composant `ws2` (mary) et lancez `nmap` :

sudo nmap www.example.com

```
:-S ./start snort.sh
18:01.484579  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.1.2 -> 192.168.2.1
18:01.484653  [**] [1:451:5] ICMP Timestamp Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.1.2 -> 192.168.2.1
```

Alertes ICMP visibles sur le terminal snort

Que voyez-vous sur le composant snort ? Inclut-il l'alerte ICMP PING NMAP que vous avez vue lorsque le poste de travail distant a exécuté nmap ? Pourquoi ?

Sur Snort, seules quelques alertes ICMP sont visibles. L'alerte ICMP PING NMAP n'est pas détectée car le trafic de ws2 n'est pas transmis à Snort sans mirroring.

Allez dans le composant passerelle et modifiez le script /etc/rc.local afin que le trafic de la station de travail de Mary soit reflété dans le composant snort. Pour ce faire, ajoutez la ligne suivante à la section de ce fichier qui définit la mise en miroir des paquets :

```
iptables -t mangle -A PREROUTING -i $lan2 -j TEE --gateway 192.168.3.1
```

```
GNU nano 2.5.3 File: /etc/rc.local
sudo iptables -t nat -A POSTROUTING -o $lan1 -p tcp --dport 22 -d 192.168.1.2 -j SNAT --to-source 192.168.1.10
sudo iptables -t nat -A PREROUTING -i $wan -p tcp --dport 4444 -j DNAT --to-destination 192.168.1.2
sudo iptables -t nat -A POSTROUTING -o $lan1 -p tcp --dport 4444 -d 192.168.1.2 -j SNAT --to-source 192.168.1.10
#
# restart the DNS forwarder (rely on DNS provided by ISP)
#
/etc/init.d/dnsmasq restart
echo "nameserver 203.0.113.1" > /etc/resolv.conf
route del -host 172.17.0.1
iptables -t mangle -A PREROUTING -i $lan2 -j TEE --gateway 192.168.3.1
```

Ajout d'une règle iptables sur la passerelle (machine ubuntu@gateway) pour qu'il soit refléter le trafic de ws2 vers la machine Snort.

Ensuite, exécutez le script pour remplacer les règles iptables par vos nouvelles règles :

```
sudo /etc/rc.local
```

```
ubuntu@gateway:~$ sudo nano /etc/rc.local
ubuntu@gateway:~$ sudo /etc/rc.local
[ ok ] Restarting dnsmasq (via systemctl): dnsmasq.service.
SIOCDELRT: No such process
ubuntu@gateway:~$
```

Exécution du script rc.local sur la passerelle afin d'appliquer les nouvelles règles iptables

Maintenant, redémarrez snort et exécutez à nouveau nmap depuis l'ordinateur ws2 de Mary.

4.7 Distinguer le trafic par adresse

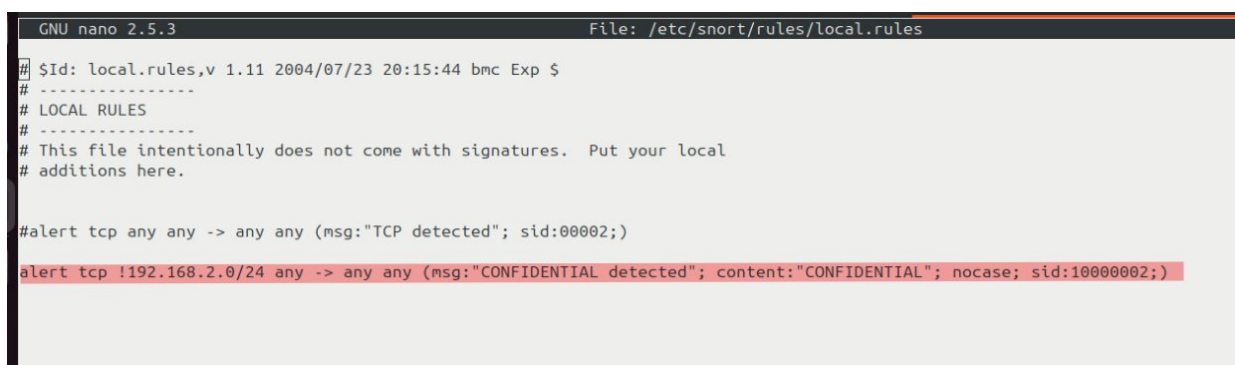
Lancez Firefox sur l'ordinateur ws2 de Mary pour visualiser le plan d'affaires confidentiel :

```
firefox www.example.com/plan.html
```

Puis observez la console snort. Cela ne suffira pas ! Les esprits vifs de la startup doivent pouvoir consulter leur plan d'affaires confidentiel sans que les alertes IDS ne se déclenchent. Mais ils veulent surveiller les ordinateurs internes pour détecter tout trafic suspect, par exemple les scans nmap. Dans cette tâche, vous allez ajuster votre règle snort de sorte que l'alerte CONFIDENTIEL ne se déclenche que lorsque le plan est accessible par des adresses extérieures au site.

Si vous examinez les règles trouvées dans le répertoire /etc/snort/rules, vous constaterez que les règles ont la forme générale de :

```
alert <protocole> <adr_source> <src_port> -> \
      <adr_dest> <port_dest> <options de la règle entre parenthèses>.
```



```
GNU nano 2.5.3 File: /etc/snort/rules/local.rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures. Put your local
# additions here.

#alert tcp any any -> any any (msg:"TCP detected"; sid:000002;)
alert tcp 192.168.2.0/24 any -> any any (msg:"CONFIDENTIAL detected"; content:"CONFIDENTIAL"; nocase; sid:10000002;)
```

Modification de la règle Snort afin de distinguer le trafic interne (192.168.2.0/24) et externe pour la détection du contenu confidentiel.

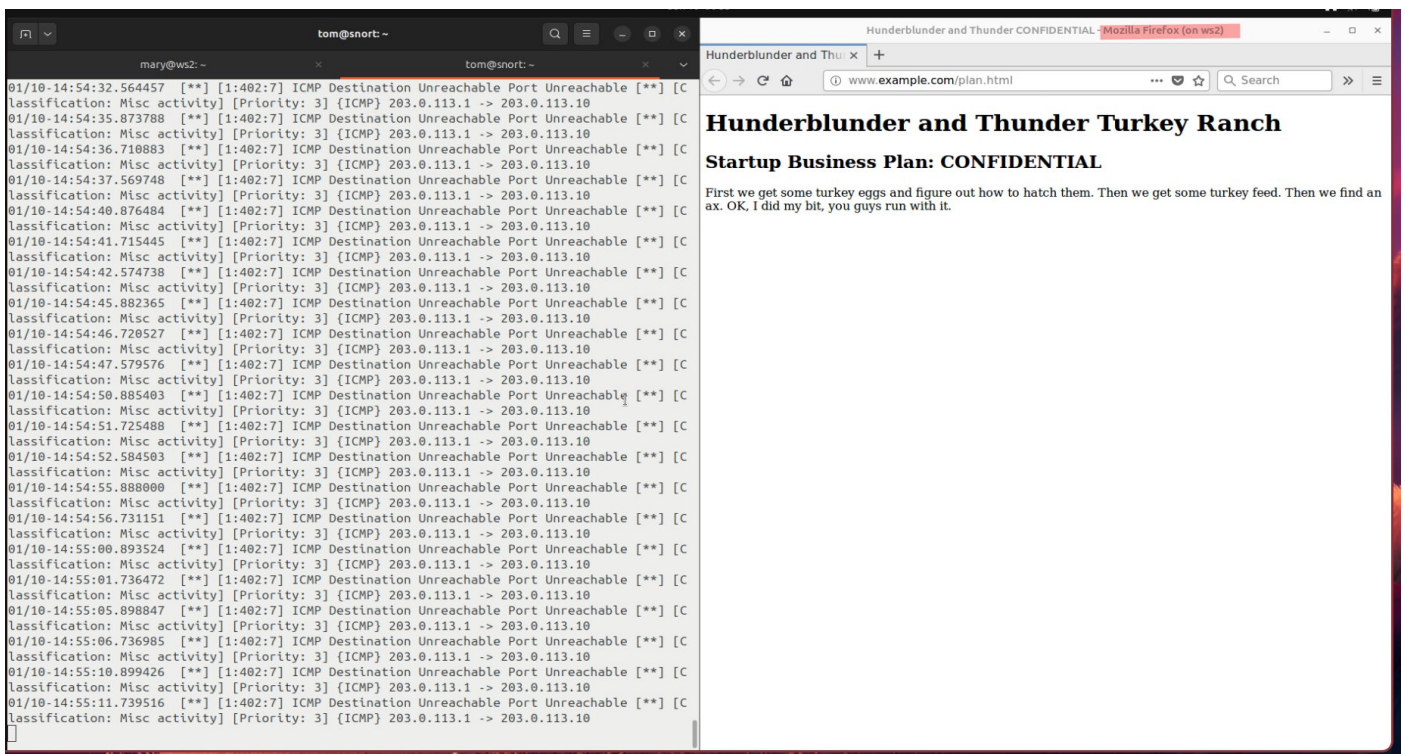
Les règles snort comprennent deux champs d'adresse : `adr_source` et `adr_dest`. Ces adresses sont utilisées pour vérifier la source d'où provient le paquet et la destination du paquet. L'adresse peut être une adresse IP unique ou une adresse réseau. Vous avez probablement utilisé le mot-clé `any` pour appliquer une règle à toutes les adresses. Pour les adresses réseau, l'adresse est suivie d'une barre oblique et du nombre de bits du masque de réseau. Par exemple, l'adresse réseau `192.168.2.0/24` représente le réseau de classe C `192.168.2.0` avec 24 bits dans le masque réseau.

Notez qu'en raison de l'utilisation de la NAT, tout le trafic provenant du serveur web et destiné à une adresse externe aura pour adresse de destination la passerelle (c'est-à-dire `192.168.1.10`), tandis que le trafic web destiné aux utilisateurs internes aura une adresse de destination correspondant à l'utilisateur interne.

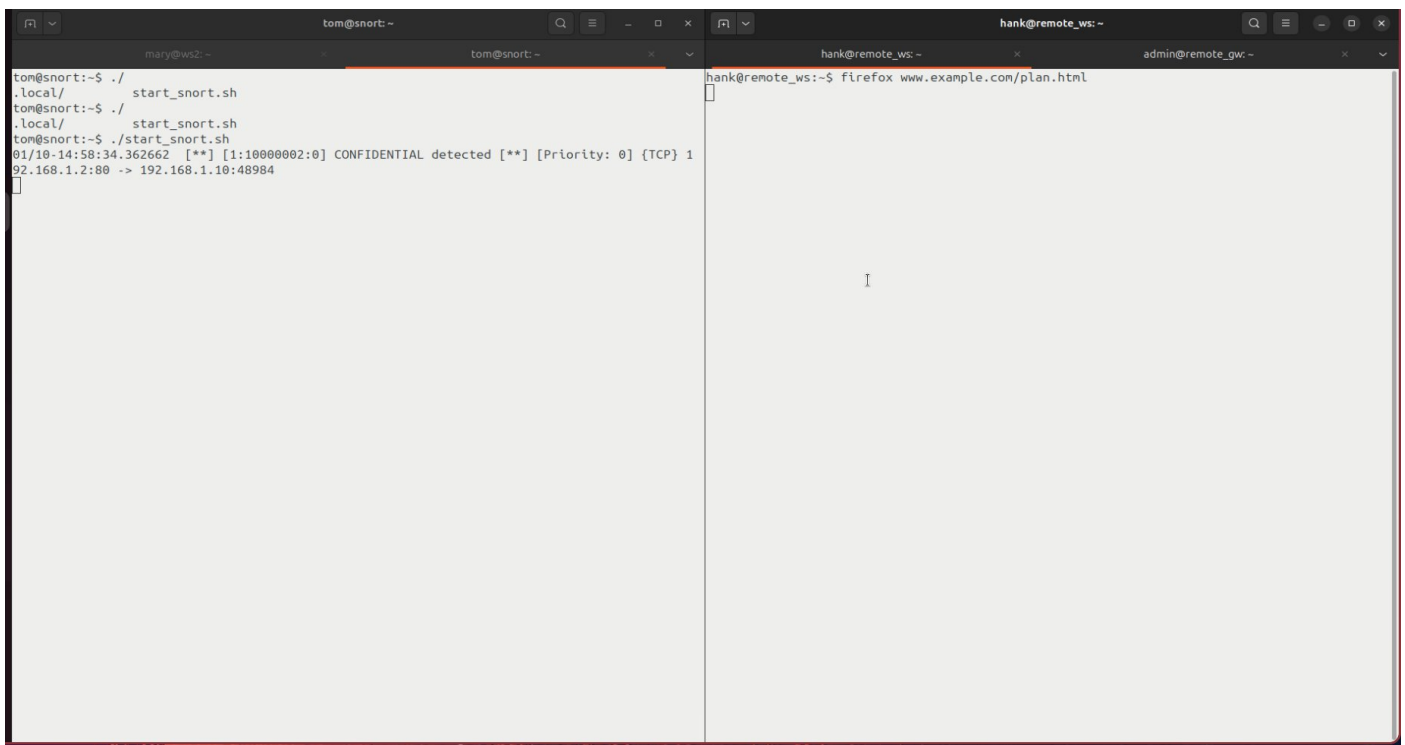
Pour cette tâche, vous devez définir vos règles snort et la mise en miroir du trafic de telle sorte que :

1. L'accès externe au plan d'affaires génère une alerte ;
2. L'accès interne au plan d'affaires ne génère pas d'alerte ;
3. L'utilisation externe ou interne de nmap génère une alerte ICMP NMAP PING.

Vous devez tester chacun de ces critères au cours d'une seule session snort, c'est-à-dire que si vous modifiez une règle snort, ou la mise en miroir des ports, vous devez recommencer vos tests.



Accès interne à la page plan.html depuis la station ws2 (mary) sans alerte CONFIDENTIAL, seules des alertes ICMP sont générées.



Détection par Snort d'un accès externe à la page plan.html, déclenchant l'alerte CONFIDENTIAL.

```
tom@snort: ~  
mary@ws2: ~  
tom@snort:~$ ./start_snort.sh  
01/10-15:06:44.060749  [**] [1:469:3] ICMP PING NMAP [**] [Classification: Attempted Inform  
ation Leak] [Priority: 2] {ICMP} 192.168.2.1 -> 192.168.1.2  
01/10-15:06:44.060749  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Prior  
ity: 3] {ICMP} 192.168.2.1 -> 192.168.1.2  
01/10-15:06:44.060778  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity]  
[Priority: 3] {ICMP} 192.168.1.2 -> 192.168.2.1  
01/10-15:06:44.060889  [**] [1:453:5] ICMP Timestamp Request [**] [Classification: Misc act  
ivity] [Priority: 3] {ICMP} 192.168.2.1 -> 192.168.1.2  
01/10-15:06:44.060899  [**] [1:451:5] ICMP Timestamp Reply [**] [Classification: Misc activ  
ity] [Priority: 3] {ICMP} 192.168.1.2 -> 192.168.2.1  
01/10-15:06:45.389979  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attem  
pted Information Leak] [Priority: 2] {TCP} 192.168.2.1:42027 -> 192.168.1.2:705  
01/10-15:06:45.400825  [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted In  
formation Leak] [Priority: 2] {TCP} 192.168.2.1:42027 -> 192.168.1.2:161  
]
```

Alertes ICMP NMAP PING générées sur la machine Snort lors d'un scan Nmap effectué depuis la station ws2 (mary).

```
tom@snort: ~  
mary@ws2: ~  
hank@remote_ws: ~  
admin@remote_gw: ~  
tom@snort:~$ ./start_snort.sh  
01/10-15:08:23.497901  [**] [1:469:3] ICMP PING NMAP [**] [Classification: Attempted Inform  
ation Leak] [Priority: 2] {ICMP} 203.0.113.20 -> 203.0.113.10  
01/10-15:08:23.497901  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Prior  
ity: 3] {ICMP} 203.0.113.20 -> 203.0.113.10  
01/10-15:08:23.498055  [**] [1:453:5] ICMP Timestamp Request [**] [Classification: Misc act  
ivity] [Priority: 3] {ICMP} 203.0.113.20 -> 203.0.113.10  
01/10-15:08:24.842105  [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted In  
formation Leak] [Priority: 2] {TCP} 203.0.113.20:42540 -> 203.0.113.10:161  
01/10-15:08:24.894850  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attem  
pted Information Leak] [Priority: 2] {TCP} 203.0.113.20:42540 -> 203.0.113.10:705  
]  
hank@remote_ws:~$ sudo nmap www.example.com  
Starting Nmap 7.01 ( https://nmap.org ) at 2026-01-10 15:08 UTC  
Nmap scan report for www.example.com (203.0.113.10)  
Host is up (0.000029s latency).  
Not shown: 996 closed ports  
PORT      STATE SERVICE  
22/tcp    open  ssh  
53/tcp    open  domain  
80/tcp    open  http  
443/tcp   open  https  
  
Nmap done: 1 IP address (1 host up) scanned in 1.53 seconds  
hank@remote_ws:~$
```

Détection par Snort d'un scan Nmap lancé depuis la station distante remote_ws, avec génération d'alertes ICMP NMAP PING.

5. Soumission

Après avoir terminé le laboratoire, allez dans le terminal de votre système Linux qui a été utilisé pour commencer le laboratoire et tapez :

```
stoplab snort
```

Quand vous arrêtez le labo, le système affichera un chemin vers les résultats zippés du labo sur votre système Linux. Fournissez ce fichier à votre instructeur, par exemple, via le site Sakai.