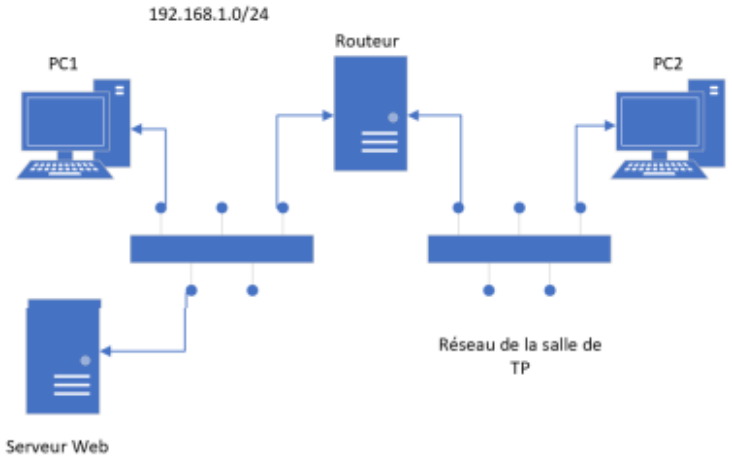
Randall ETZWEILER A1
Bastien HELEC A1

R401 - TP 4 NAT et Filtrage sous linux



Adressage IP :

Réseau interne :
PC1 : 192.168.1.1/24
Web : 192.168.1.80/24
Routeur : 192.168.1.254/24

Réseau externe :

Routeur : 10.213.0.203
PC2 : 10.203.0.98

Exercice 1 :
Sur quel hook doit se placer le hook du SNAT ?

Les règles du SNAT doit se placer sur le hook postrouting :

Exercice 2 Configurer les tables, chaînes et règles pour faire du SNAT afin de changer l'adresse IP du PC1 lorsqu'il essaie de s'adresser au PC2 et ainsi rendre la communication possible.
Vous indiquerez dans votre compte-rendu les commandes saisies.

Commandes nécessaires pour configurer le SNAT sur R1 :

Effacement toutes les règles existantes :
iptables -F
iptables -t nat -F
iptables -t mangle -F
iptables -X

On active le forward :

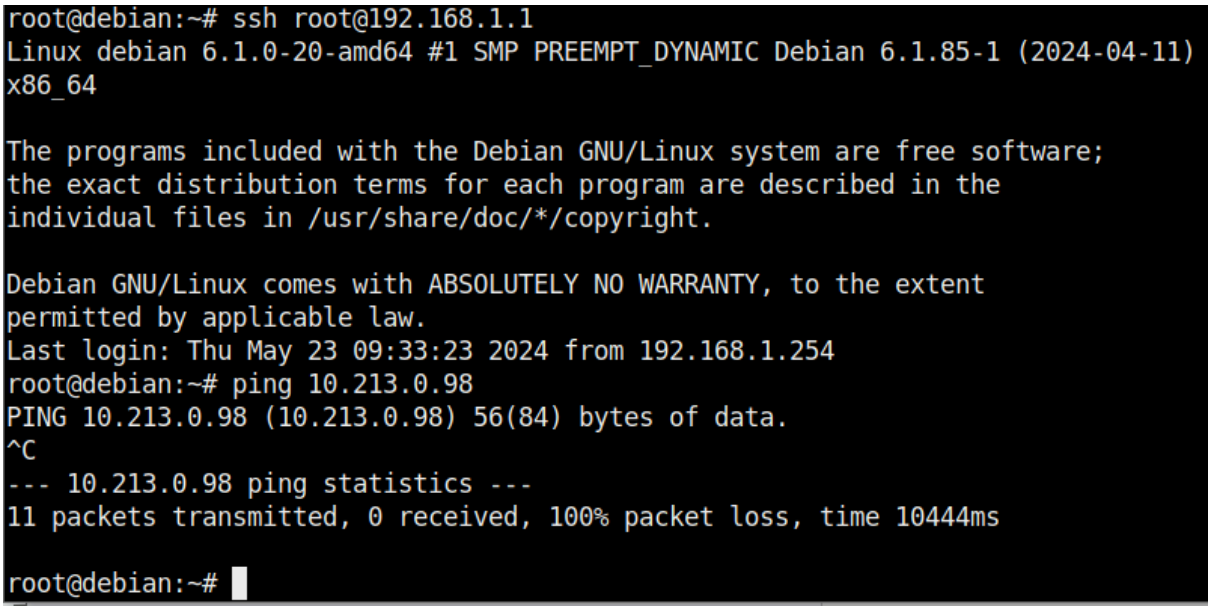
echo 1 > /proc/sys/net/ipv4/ip_forward

Configuration SNAT en nftables :

nft add table SNATTP4
nft add chain SNATTP4 postroute {type nat hook postrouting priority 0 \;}
nft add rule SNATTP4 postroute ip saddr 192.168.1.0/24 oif eth0 snat 10.213.0.203

Exercice 3 Proposer une méthode pour tester votre configuration et effectuer les relevés nécessaires pour illustrer le bon fonctionnement de votre configuration.

Depuis PC1 (192.168.1.1), on essaie de pinguer PC2 (10.213.0.98) avant d'appliquer les règles iptables :



Puis on essaie de refaire le ping après avoir appliqué les règles iptables :

```
test@213-14:~$ ssh root@10.213.0.203
root@10.213.0.203's password:
Linux debian 6.1.0-20-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.85-1 (2024-04-11)
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: Thu May 23 08:46:44 2024 from 10.213.0.98
root@debian:~# ping 10.213.0.98
PING 10.213.0.98 (10.213.0.98) 56(84) bytes of data.
64 bytes from 10.213.0.98: icmp_seq=1 ttl=64 time=0.154 ms
64 bytes from 10.213.0.98: icmp_seq=2 ttl=64 time=0.811 ms
64 bytes from 10.213.0.98: icmp_seq=3 ttl=64 time=0.653 ms
^C
--- 10.213.0.98 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2120ms
rtt min/avg/max/mdev = 0.154/0.539/0.811/0.280 ms
root@debian:~#
```

1.2 MASQUERADE

Lorsque l'adresse IP de sortie du réseau n'est pas fixe, il n'est pas possible de réaliser du SNAT avec une règle précisant en dur l'adresse de sortie. On met en place alors une règle de type SNAT qui s'appelle MASQUERADE.

Exercice 4 Supprimer la règle précédente et la remplacer par une règle de type MASQUERADE.

Suppression de la règle précédente :

nft delete rule SNATTP4 postroute ip saddr 192.168.1.0/24 oif eth0 snat 10.213.0.203

Ajout de la règle MASQUERADE :

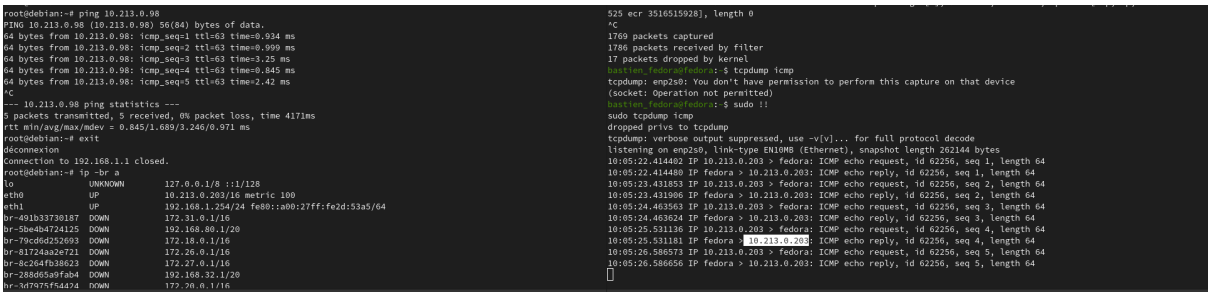
nft add rule SNATTP4 postroute oif eth0 masquerade

Proposer une méthode pour tester votre configuration et faire les relevés pour illustrer son bon fonctionnement.

Avant de configurer MASQUERADE, on s'assure que PC1 ne peut pas communiquer directement avec PC2 via un ping de PC2 depuis PC1 :

[oublie du ping]

Après avoir ajouté la règle MASQUERADE avec nftables, on essaie de nouveau de pinger PC2 depuis PC1 :



Ici on envoi sur la gauche un ping sur le PC1 (192.168.1.1) vers le PC2 (10.213.0.98) Toujours sur la gauche on se connecte ensuite sur le routeur, on obtient l'adresse ip du routeur(10.213.0.203) .

Sur la droite on voit le ping du PC1 vers le PC2 avec l'adresse IP du routeur

On peut vérifier encore une fois ce bon fonctionnement via la visualisation de la sortie du serveur web (192.168.1.80/24) vers le web avec un traceroute sur celui-ci :

```
root@debian:~# ip -br a |grep eth0
eth0             UP             192.168.1.80/24
root@debian:~# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.254)  0.471 ms  0.490 ms  0.446 ms
 2  10.213.255.254 (10.213.255.254)  2.959 ms  3.279 ms  3.558 ms
 3  gw.iutbeziers.fr (194.199.227.254)  1.078 ms  0.960 ms  0.859 ms
 4  100.77.22.254 (100.77.22.254)  2.086 ms  1.991 ms  1.896 ms
 5  100.74.101.65 (100.74.101.65)  3.203 ms  3.109 ms  3.112 ms
 6  100.77.255.243 (100.77.255.243)  1.974 ms  2.208 ms  2.121 ms
 7  100.77.255.241 (100.77.255.241)  2.159 ms  2.248 ms  2.181 ms
 8  100.77.255.113 (100.77.255.113)  2.046 ms  2.207 ms  2.127 ms
 9  100.77.255.114 (100.77.255.114)  2.364 ms  2.706 ms  2.619 ms
10  193.55.200.140 (193.55.200.140)  2.993 ms  2.589 ms  2.655 ms
11  et-3-1-7-ren-nr-marseille1-rtr-131.noc.renater.fr (193.51.180.100)  5.055 ms  4.949 ms  6.039 ms
12  et-0-1-0-ren-nr-marseille2-rtr-131.noc.renater.fr (193.51.177.127)  4.634 ms  4.807 ms  4.742 ms
13  72.14.218.132 (72.14.218.132)  4.950 ms  4.894 ms  4.581 ms
14  192.178.105.27 (192.178.105.27)  5.546 ms 192.178.105.171 (192.178.105.171)  5.496 ms 192.178.105.209 (192.178.105.
209)  5.721 ms
15  142.251.78.91 (142.251.78.91)  4.887 ms  4.802 ms 72.14.232.43 (72.14.232.43)  5.750 ms
16  dns.google (8.8.8.8)  4.889 ms  4.566 ms  4.464 ms
root@debian:~#
```

1.3 DNAT

Le DNAT est utilisé quand une machine de l'Internet essaie de joindre un serveur se trouvant dans une zone avec un adressage privé. Il faut donc que le client sur internet passe par le routeur et que celui-ci relaie le message vers le serveur interne.

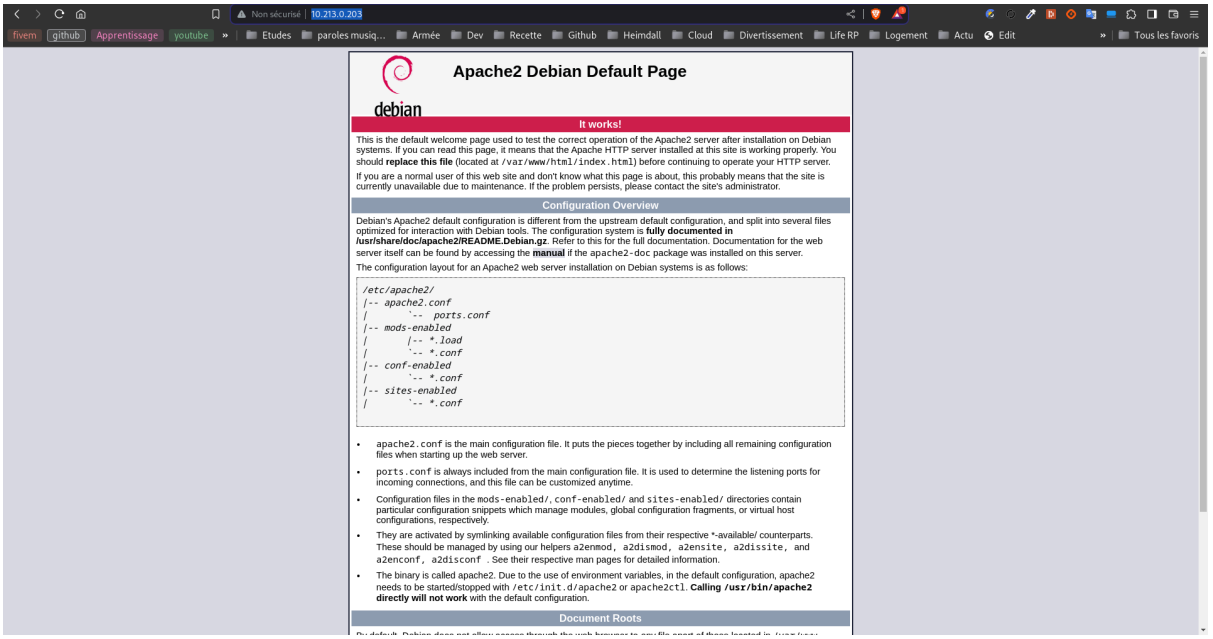
Exercice 5 Sur quel hook doit se placer la règle de DNAT ?
Sur le hook prerouting

Exercice 6 Configurer les tables, chaînes et règles pour faire du DNAT afin de permettre au PC2 d'accéder au serveur web interne. Vous indiquerez dans votre compte-rendu les commandes saisies.

nft add table DNATTP4
nft add chain DNATTP4 preroute { type nat hook prerouting priority 0 \; }
nft add rule DNATTP4 preroute iif eth0 tcp dport 80 dnat 192.168.1.80

Exercice 7 Proposer une méthode pour tester votre configuration et effectuer les relevés nécessaires pour illustrer le bon fonctionnement de votre configuration.

Sur le web on tape 10.213.0.203:80



2 Filtrage

Le module NETFILTER et son interface avec nftable peut également être utilisé pour filtrer les paquets entrants, sortants ou traversants le routeur. Dans cette deuxième partie, nous testerons différents filtres permettant de restreindre le trafic réseau.

2.1 Sans état : Stateless

La première partie consiste en du filtrage simple de paquets en fonction de critères sur les adresses ou sur les ports.

Exercice 8 Récupérer l'adresse IP d'un site web et proposer une règle permettant d'empêcher les consultations de ce site. Vous indiquerez dans votre compte-rendu les commandes saisies.

Récupération de l'IP d'un site :

dig facebook

```
bastien_fedora@fedora:~$ dig  facebook.com

; <<>> DiG 9.18.26 <<>> facebook.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33989
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;facebook.com.                  IN      A

;; ANSWER SECTION:
facebook.com.           50      IN      A       157.240.202.35

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Thu May 23 10:25:09 CEST 2024
;; MSG SIZE  rcvd: 57
```

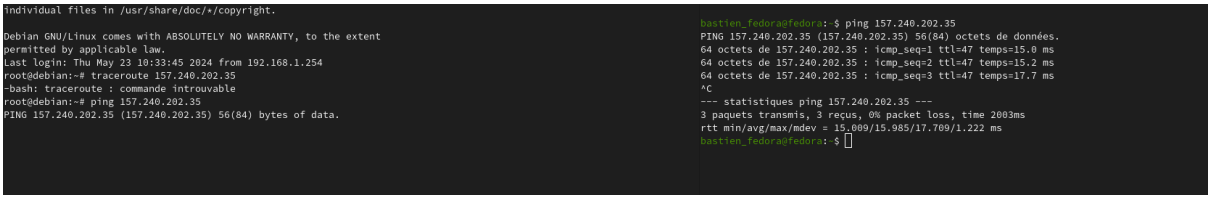
Adresse Ip de facebook : **157.240.202.35**

Règle permettant d'empêcher les consultations de ce site :
Création d'une table et une chaîne de filtrage telle que :

nft add table inet filter
nft add chain inet filter input { type filter hook input priority 0 \; }
nft add chain inet filter forward { type filter hook forward priority 0 \; }
nft add chain inet filter output { type filter hook output priority 0 \; }

Ajout d'une règle pour bloquer l'accès à l'adresse IP du site web :

nft add rule inet filter output ip daddr 157.240.202.35 drop



Quel peut être la limite de cette approche (on pourra penser aux adresses IP utilisées par des serveurs comme Facebook ou d'autres grosses entreprises qui gèrent beaucoup de trafic) ?

La limite de cette approche est que les grands sites comme Facebook utilisent de nombreuses adresses IP dynamiques et des CDN, rendant inefficace le blocage complet en

se basant sur une seule adresse IP, car ils peuvent rapidement changer et utiliser une large plage d'adresses.

Exercice 9 On activera un serveur SSH sur le routeur pour que des personnes du réseau local puissent l'administrer. Le premier filtre que nous allons mettre en place est de bloquer le trafic sur le port 22 pour des messages provenant du réseau du PC2. Vous indiquerez dans votre compte-rendu les commandes saisies.

apt install ssh*

nft add rule inet filter input ip saddr 10.213.0.0/16 tcp dport 22 drop

Exercice 10 Si on suppose que la passerelle ou le réseau interne dispose de plusieurs services, est-il raisonnable de spécifier un par un les ports accessibles depuis l'intérieur, mais pas l'extérieur ? Quel est le risque d'une telle approche ?

Gérer et maintenir une longue liste de règles pour chaque port peut devenir complexe et sujet à des erreurs, augmentant la probabilité de failles de sécurité. De plus, il est facile d'oublier de bloquer certains ports ou services, laissant des ouvertures non sécurisées accessibles depuis l'extérieur.

Que faudrait-il préciser dans la déclaration de la chaîne pour se simplifier la vie et assurer une meilleure sécurité ?

Pour simplifier la gestion des règles et améliorer la sécurité, il est préférable de suivre une approche par défaut de blocage et d'autoriser explicitement uniquement les services nécessaires. Cela se fait en utilisant une politique de refus par défaut (default deny) et en spécifiant les exceptions.

Exercice 11 Proposer une règle permettant de rejeter les paquets entrants sur l'interface de sortie du routeur (celle reliée au réseau de la salle) correspondant à des ECHO_REQUEST ICMP. Tester la différence entre les 2 actions "REJECT" et "DROP". Vous illustrerez ceci avec des captures des résultats obtenus sur vos consoles.

Chaîne de filtrage pour les paquets forwards (traversants) :
nft add chain inet filter forward { type filter hook forward priority 0 \; }

Règle pour rejeter les paquets ICMP ECHO_REQUEST avec l'action REJECT :
nft add rule inet filter forward oif eth1 ip protocol icmp icmp type echo-request reject

Règle pour bloquer les paquets ICMP ECHO_REQUEST avec l'action DROP :
nft add rule inet filter forward oif eth1 ip protocol icmp icmp type echo-request drop

Exercice 12 Donner le fichier de règles permettant de
▷ **bloquer par défaut tout le trafic sur le routeur ;**
▷ **autoriser les ping sur le routeur**
▷ **n'autoriser que les connexions entrantes en SSH provenant du réseau interne (gauche**
sur la figure 1)
▷ **autoriser les connexions traversant le routeur à destination de serveurs web (port destination 80)**

```
#!/usr/sbin/nft -f

table inet filter {
    chain input {
        type filter hook input priority 0; policy drop;

        # Autoriser les paquets loopback
        iif "lo" accept

        # Autoriser les pings
        ip protocol icmp icmp type echo-request accept

        # Autoriser SSH depuis le réseau interne
        ip saddr 192.168.1.0/24 tcp dport 22 accept
    }

    chain forward {
        type filter hook forward priority 0; policy drop;

        # Autoriser les connexions traversant le routeur à destination de serveurs web (port 80)
        tcp dport 80 accept
    }

    chain output {
        type filter hook output priority 0; policy accept;
    }
}

# Appliquer les règles immédiatement
add table inet filter
add chain inet filter input { type filter hook input priority 0 \; policy drop \; }
add chain inet filter forward { type filter hook forward priority 0 \; policy drop \; }
add chain inet filter output { type filter hook output priority 0 \; policy accept \; }

add rule inet filter input iif "lo" accept
add rule inet filter input ip protocol icmp icmp type echo-request accept
add rule inet filter input ip saddr 192.168.1.0/24 tcp dport 22 accept
```

```
add rule inet filter forward tcp dport 80 accept
```