Concepteur original : P. Mathieu

Étude d'un pare-feu

25 mars 2024

Vous devez rendre une version électronique de votre rapport à l'enseignant avec des copies d'écran.

But du TP

Vous devez disposer pour cette manipulation de 4 ordinateurs : deux binômes doivent travailler encsemble.

Les passerelles auront le rôle de pare-feu par rapport à l'extérieur (= le réseau de l'IUT). Les clients représentent le réseau intérieur à protéger.

Client gauche Passerelle gauche ① Passerelle droite ② Client droit

Illustration 1 Plan de la maquette

On a donc 2 étapes principales dans le TP : mise en place du logiciel de pare-feu, puis mise en place des règles d'accès.

Commandes de base (systemclt, apt, ip,etc) :
http://www.i3s.unice.fr/~urvoy/docs/M3105/Memo.pdf
Si vous voulez utiliser ifconfig, il faut installer le paquet net-tools.

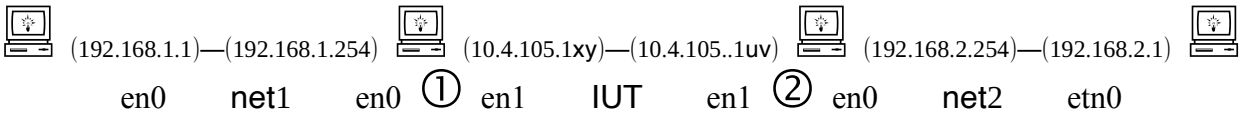
I. Préliminaires

1. Câblage et définition du réseau

On va utiliser des machines virtuelles Debian (11 ou supérieur) sur les clients et passerelles car il faut être super utilisateur pour faire ces manipulations.

Important : Pour passer root, faites un **`su - l (la lettre l, pas le chiffre 1)`** et non un simple su sinon vous ne chargerez pas les variables d'environnement de root, et notamment la commande iptables sera introuvable.

Reproduire le schéma ci-dessous avec les instructions ci après :



Important :

- Les adresses des passerelles sont du 10.4.105.1xy car les VMs récupèrent des adresses du type 10.4.105.101, 10.4.105.102, etc.
- adapter avec des adresses en 110 au lieu de 105 si vous êtes en salles 410 !

Adressage :

Machines client:

1) Débrancher le câble physicue en1.

2) Reliez en0 du client à en0 de la passerelle.

2) Dans la VM, désactivez en1 avec la commande :

```
ip link set dev en1 down
```

4) Assigner une adresse à l'interface en0

Rappel : pour assigner une adresse (avec x =1 ou x=2)

```
ip addr add 192.168.x.1/24 dev en0
```

5) La mettre à up

```
ip link set dev en0 up
```

Pour la liaison passerelle-passerelle, on utilise le switch de la salle, c'est-à-dire la prise en1, qui est normalement déjà branchée.

Marchines passerelle :

1) Assigner une adresse à l'interface en0

Rappel : pour assigner une adresse (avec x =1 ou x=2)

```
ip addr add 192.168.x.254/24 dev en0
```

4) La mettre à up

```
ip link set dev en0 up
```

Transformation des passerelles en routeurs :

Afin que les passerelles puissent transmettre les paquets de chaque côté, il faut autoriser la recopie sur les interfaces en passant la commande :

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

Routage sur clients et passerelles :

Sur les clients il faut définir une route par défaut et sur les passerelles, une route vers le réseau distant.

- sur le client

```
ip route add default via 192.168.x.254
```

- Sur la passerelle de gauche ①

```
ip route add 192.168.2.0/24 via 10.4.105.1uv
```

- Sur la passerelle de droite ②

```
ip route add 192.168.1.0/24 via 10.4.105.1xy
```

Important : si on veut retirer une route, c'est la même commmande, mais avec `del` au lieu de `add` !!

Test :

Tester la communication entre les différentes machines avec la méthode suivante :

(i) lancez un ping sur une des machines client vers l'autre client

(ii) faites des tcpdump sur les interfaces successives, de proche en proche entre la source et la destination. Ex : tcpdump -i en0 -n 'icmp' (-n pour éviter les résolutions DNS).

II. Mise en place du pare-feu

La mise en œuvre du pare-feu va être progressive mais est souvent difficile à tester car en cas d'erreur, « ça ne passe plus » et il est difficile de voir pourquoi. Pour pouvoir passer facilement de l'état protégé au mode tout ouvert, nous allons utiliser des scripts qui mémoriseront toutes les commandes. Tout se passe ici sur les passerelles.

1. Organisation des opérations

Tout d'abord récupérer depuis la page Web du cours :

http://www.i3s.unice.fr/~urvoy/teaching/courses/m3104/

les 2 scripts « `metFW.sh` » et l'autre « `oteFW.sh` ».

Rendez les 2 fichiers exécutables avec la commande

chmod +x metFW.sh

chmod +x oteFW.sh

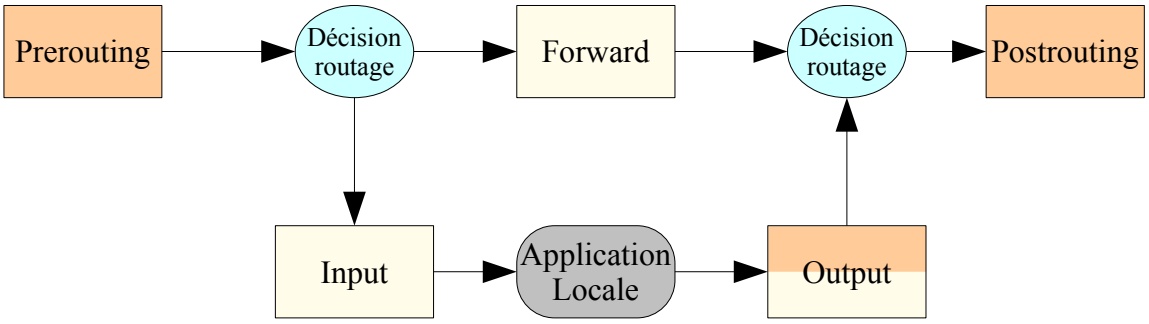
Adaptez les interfaces intérieure/extérieure dans les deux fichiers pour correspondre aux adresses de vos passerelles.

Nous allons ajouter des commandes dans le fichier « `metFW.sh` » pour construire notre pare-feu. Attention l'ordre des commandes `iptables` a son importance.

2. Présentation rapide de `netfilter`

On paramètre le noyau au moyen de la commande `iptables` qui utilise pour cela des critères (`matching`), des règles (suite de paramètres), des chaînes (ensemble de règles) des cibles (le détail dans `man iptables`), le tout dans plusieurs tables. Chaque table regroupe plusieurs fonctionnalités :

`filter` : filtre les paquets entrants, sortants ou transmis
`nat` : avant ou après les étapes de routage modifie les adresses et les ports d'origine ou de destination
`mangle` : à tous les niveaux, modifie n'importe quel champ des en-têtes IP.



Dessin 1 : Architecture Netfilter

En pratique, cela veut dire que chacune de ces tables contient des chaînes prédéfinies :
`filter` : Input, Output et Forward
`nat` : Prerouting, Postrouting et Output
`mangle` : toutes les chaînes. Input, Output, Forward, Prerouting, Postrouting

Important :
- **Input** est la chaine qui traite les données destinées aux services qui tournent sur la machine
- **Output** est la chaine qui traite les données générées par les services qui tournent sur la machine
- **Forward** est la chaine qui traite les données qui transitent sur la machine lorqu'elle joue le rôle du routeur.

La commande `iptables` ajoute donc des règles à une chaîne prédéfinie (ou définie par l'utilisateur). Ces règles sont constituées de critères de sélection, suivis d'un « -j cible» où cible est une action à effectuer : DROP pour jeter le paquet, REJECT pour le rejeter en prévenant la machine d'origine (RST), ACCEPT pour continuer le processus tout en finissant la scrutation de la chaîne, LOG pour conserver une trace (n'est pas final, la chaîne continue d'être scrutée), SNAT nat sur la source, DNAT nat sur la destination, MASQUARADE nat avec suivi de connexion, MARK marque sur le paquet …

3. Filtrage élémentaire

a) Filtrage local

Appliquer le fichier de commande « `metFW.sh` »

```
./metFW.sh
```

Tout le réseau doit être verrouillé.

```
ping 127.0.0.1
```

ne devrait pas fonctionner. Testez le. Idem pour ssh

```
ssh root@127.0.0.1
```

On va libérer les accès locaux : ajoutez dans `metFW.sh` les commandes suivantes au bon endroit dnas le fichier (là où il y a écrit "# Il faut remplir ici ") et on exécute le fichier.

```
$ipt -A INPUT -i lo -j ACCEPT
$ipt -A OUTPUT -o lo -j ACCEPT
```

Pour comprendre par quelle chaine les paquets entrants/sortant ou en transit son traitées, il suffit de demander à iptables qui fournit des statistiques en temps réel avec la commande iptables -L -v. Pour suivre en continue, on combine avec watch :

```
watch  iptables -L -v
```

Faites tourner ce watch dans un autre terminal pendant que vous vérifiez si tout fonctionne.

b) Filtrage intérieur

On note `$intif` l'interface intérieure et `$extif` l'extérieure.

On autorise le réseau interne à se connecter à la passerelle avec :

```
$ipt -A INPUT -i $intif -j ACCEPT
$ipt -A OUTPUT -o $intif -j ACCEPT
```

Est-ce que le client peut pinger la passerelle?

Sur la passerelle, peut-on contacter kheops ou l'autre passerelle ?

Pour être sûr que ce sont bien nos nouvelles règles qui ont modifié le fonctionnement, refaire les tests après avoir appliqué `oteFW.sh`.

c) Filtrage extérieur

On doit cependant pouvoir accéder au réseau extérieur malgré ou plutôt grâce à la passerelle. Il faut autoriser la traversée de la passerelle mais sous conditions. Actuellement la passerelle ne peut accéder à l'extérieur et en supposant qu'elle ait des réponses celles-ci ne peuvent rentrer.

Il existe dans le noyau une notion de connexion et donc de nouvelle connexion (NEW), de connexion établie (ESTABLISHED) ou associée (RELATED). **Cela permet par exemple de**

laisser entrer un paquet correspondant à une connexion déjà établie par le client local : c'est ce que fait tout firewall ou votre box à la maison (elle le fait avec netfiler d'ailleurs) !

Nous allons donc autoriser les paquets à sortir mais la réponse ne pourra rentrer que si elle est établie ou associée. Les nouvelles connexions ne pourront pas être ouvertes à partir de l'extérieur.

L'état de la connexion est testé par `-m state --state`

```
$ipt -A FORWARD -i $intif -o $extif -j ACCEPT
```

Et l'on autorise le paquet en réponse à rentrer

```
$ipt -A FORWARD -i $extif -o $intif -m state --state
      ESTABLISHED,RELATED -j ACCEPT
```

L'état des connexions peut être visualisé dans pseudo système de fichier `/proc/net/ip_conntrack`

d) Filtrage ICMP

Dans le fichier `metFW.sh` il y a un certain nombre de valeurs positionnées dans `/proc`. Certaines concernent ICMP, et empêchent le système de répondre. C'est la meilleure méthode car la passerelle n'est pas sensée avoir d'existence visible par rapport à l'extérieur et doit-être transparente pour l'intérieur.

Pour des raisons de test il peut être utile de laisser la passerelle répondre à certaines requêtes ICMP en particulier pour l'intérieur. Attention cela peut-être une source d'information sur le système et de **deni de service**.

On autorise l'ICMP uniquement pour indiquer que la passerelle est active, et on en limite le nombre de requêtes par seconde de l'extérieur.

```
echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_all
$ipt -A INPUT -i $extif -p icmp --icmp-type echo-request -m
      limit --limit 1/s -j ACCEPT
$ipt -A INPUT -p icmp --icmp-type echo-request -j DROP
$ipt -A OUTPUT -o $extif -m state --state
      RELATED,ESTABLISHED -j ACCEPT
```

Il n'y a donc pas de limite sur l'interface intérieure

Tester la nouvelle possibilité du ping de l'intérieur et de l'extérieur en mode « `flood` » (option -f)

de l'extérieur

```
ping -f 10.4.105.1uv
```

de l'intérieur

```
ping -f 192.168.z.254
```

III. Network Address Translation (NAT)

1. Un peu de mascarade

Actuellement si vous avez appliqué `metFW.sh` modifié, on peut accéder à l'autre passerelle à partir du client mais par contre on ne peut accéder à kheops. En effet, même si les paquets sortent sur le réseau 10.4.105.0/24, il n'y a pas de réponse du fait que le pare-feu de l'IUT les jettent catégoriquement puisqu'ils viennent d'une adresse privée (192.168.x.0/24).

Une solution est de masquer l'adresse privée du client en lui substituant l'adresse de la passerelle (10.4.105.xxx). Il peut s'agir de NAT ou de Mascarade.

a) Carnaval

Le plus simple est la mascarade : tout le trafic sortant est modifié, aux adresses d'origines sont substituées l'adresse de la passerelle. Aucune configuration individuelle des machines du réseau interne n'est nécessaire.

On masque toutes les adresses du sous-réseau qui sortent sur l'interface extérieure.

```
$ipt -t nat -A POSTROUTING -s 192.168.z.0/255.255.255.0 -o
      $extif -j MASQUERADE
```

La connexion à kheops devrait maintenant fonctionner à partir du client.

b) Redirection de port

On veut autoriser l'accès de l'extérieur, vers le client mais uniquement pour ssh. D'autre part, je rappelle que du fait de la mascarade le numéro IP du client vu de l'extérieur est le numéro de la passerelle. Nous allons donc contacter la passerelle en `ssh` et c'est le client qui doit répondre. Dans la réalité c'est une machine bastion qui est dédiée à ce genre de service.

On redirige le `ssh` de l'entrée vers la machine cliente sur le même port

```
$ipt -t nat -A PREROUTING -p tcp --dport 22 -d $FW_ext_IP -j
      DNAT --to-destination $Client_IP:22
```

On autorise le nouveau FORWARD

```
$ipt -A FORWARD -p tcp --dport 22 -d $Client_IP -m state --
      state NEW,ESTABLISHED,RELATED -j ACCEPT
```

Vérifiez l'adresse du serveur SSH une fois connecté. Vous devriez voir l'adresse du client et non du bastion (la passerelle).

imprimé le 25 mars 2024