TP5 : Filtrage

Dans ce dernier TP d'Internet, nous allons mettre en place un réseau avec des contraintes de sécurité, mettre en place des règles de filtrage et les tester.

Objectifs :

- Comprendre et utiliser la notion de flux de communication
- Savoir sécuriser un réseau à l'aide du filtrage (netfilter)
- Redirection de trafic
- Découverte et utilisation d'iperf (mode client/serveur)

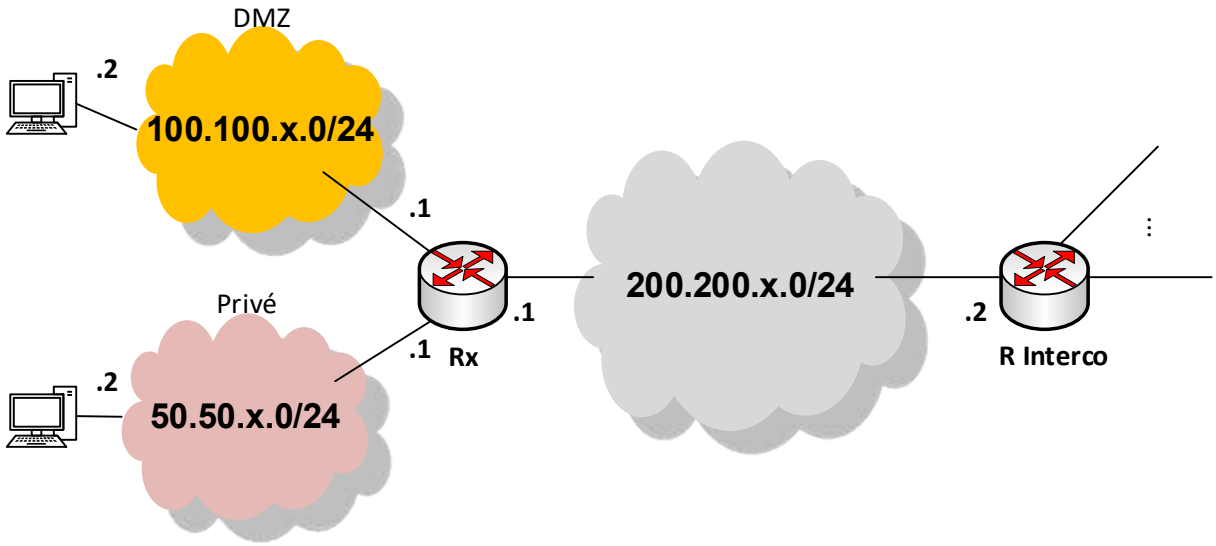

Figure 1 : Topologie

Partie I : Topologie d'un réseau d'entreprise

La figure 1 représente le réseau d'une entreprise et son interconnexion au reste du monde (ici les autres réseaux d'entreprise) via le routeur R Interco.

Pour la suite du TP, vous allez travailler en **binôme** sur trois PCs. L'un sera le routeur de l'entreprise et les deux autres seront un serveur dans la DMZ et une machine dans le réseau privé. Toute la configuration sera faite statiquement. L'**enseignant** sera lui en charge de **R Interco**.

Le site de l'entreprise est composé de deux réseaux : un réseau privé pour les utilisateurs et les services propres au réseau interne de l'entreprise, et un réseau DMZ (DeMilitarized Zone) dont le rôle est d'avoir un espace contrôlé mais ouvert vers l'extérieur, et ainsi permettre à l'entreprise de proposer des services comme un serveur web, mail, VPN, etc…

Exercice 1 :

Mettez en place l'architecture et vérifier la bonne communication entre tout l'espace publique. Quels éléments peuvent contacter le réseau privé ? Pourquoi ?

Partie II : Génération de trafic : iPerf3

Pour générer du trafic, le seul outil que nous utilisons pour le moment est ping qui repose sur le protocole icmp. Ce protocole est très pratique pour tester la connectivité dans un réseau, mais ne permet pas de tester les accès à des services précis (utilisation de ports udp/tcp) ou encore de tester le débit disponible sur un réseau.

Ici, nous proposons d'utiliser un autre outil, iperf (et plus précisément sa version 3). Iperf est une application client-serveur qui fonctionne aussi bien en tcp qu'en udp. Son utilisation la plus commune est de tester le débit d'une connexion Internet.

Exercice 2 :

Vérifier qu'iperf3 est bien installé sur vos différentes machines en utilisant la commande suivante qui vous fournira l'aide de la commande :

```
iperf3 -h
```

Si la commande ne fonctionne pas, pas de panique il faudra juste avoir accès à Internet sur votre machine et utiliser la commande suivante :

```
sudo apt-get install iperf3
```

La communication iperf3 par défaut est faite en tcp, et les données sont envoyées du client vers le serveur sur un port par défaut.

Commande du côté serveur pour lancer le serveur en mode par défaut :

```
iperf3 -s
```

Commande du côté client pour lancer le client en mode par défaut :

```
iperf3 -c <adresse IP du serveur>
```

Exercice 3 :

Mettez en place le serveur iperf3 sur la machine de la DMZ. Quel port utilise t'il par défaut ?
Ouvrir wireshark sur la machine du réseau privé et du réseau DMZ et lancer le client sur la machine du réseau privé. Quelles sont les valeurs par défaut utilisées par le client ? Quel débit atteint iperf3 ? Qu'observe t'on sur wireshark ?

Voici quelques options que l'on peut utiliser pour iperf3 :

- Côté serveur :
  - –u pour udp ;
  - –p pour le port d'écoute.
- Côté client :
  - –p pour le port destination ;
  - –u pour faire de l'udp ;
  - –t pour la durée du test ;

- –n pour le nombre d'octets à envoyer (ex : -n 10240M), cette option ne doit pas être utilisée avec l'option de temps ;
- –b pour définir le débit pour udp ;
- –R inverse le sens de la communication (c'est le serveur qui envoie les données) ;
- --get-server-output permet d'obtenir sur le client les résultats du serveur.

Pour plus d'options se référer à https://iperf.fr/fr/iperf-doc.php et/ou à la commande man.

> ***Exercice 4 :***
> Mettez en place avec iperf, l'équivalent d'un transfert http entre un serveur dans la DMZ et un client dans la partie privée de 10M du serveur.

Partie III : Principe du filtrage

L'utilisation d'une connexion directe présente l'avantage de faciliter l'accès à l'information et la publication de l'information. Elle présente cependant également un gros inconvénient, celui de la sécurité. En effet, s'il est possible d'accéder à l'information et plus généralement aux machines de façon "amicale", il est probablement tout aussi simple d'y accéder à des fins moins avouables! Une connexion directe à Internet implique donc traditionnellement la mise en place de systèmes de sécurité et de veille. Un de ces systèmes est le pare-feu. Il doit être positionné dans le réseau pour pouvoir contrôler les flux. Ici le pare-feu sera positionné sur le routeur Rx.

Le principe d'un pare-feu est de contrôler les flux de communications. Deux politiques sont possibles, soit prévoir tout ce qui ne doit pas passer et autoriser tout le reste, soit lister tout ce qui doit passer et refuser tout le reste. Il s'agit là d'une politique par défaut, et la première est trop difficile à mettre en place, c'est donc la seconde qui est mise en place.

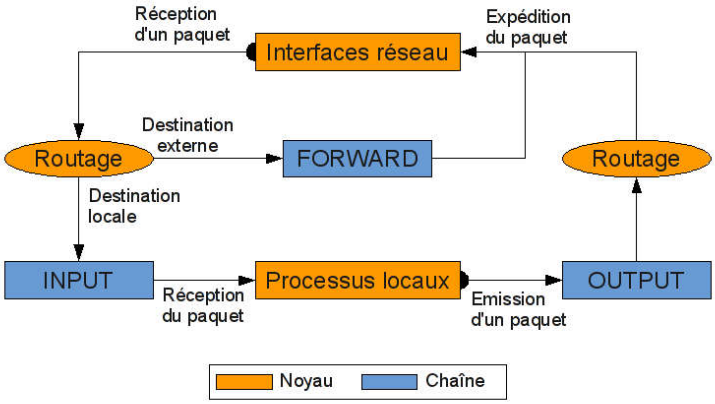

Figure 1 : Les trois chaînes de filtrage réseau dans netfilter

Plusieurs logiciels et types de pare-feu existent. Ici nous allons utiliser une sous-partie de netfilter sous linux en utilisant la commande iptables. Les trois chaînes qui permettent de faire du filtrage (Fig2 ou fascicule réseau) sont :

- OUPUT – tout ce que votre machine envoie ; c'est-à-dire du trafic généré par l'équipement lui-même ;
- INPUT – tout ce qui est à destination de votre machine ; c'est-à-dire le trafic rentrant l'équipement et traité à un niveau au-dessus d'IP.
- FORWARD – tout ce qui traverse votre machine ; à savoir un trafic qui rentre sur une interface, est routé pour enfin sortir sur une autre interface.

Même s'il peut être intéressant de protéger le pare-feu lui-même, ici c'est le FORWARD qui nous intéresse.

> ***Exercice 5 :***
> Changer la politique par défaut du pare-feu, en utilisant la commande iptables -P. Qu'observe t'on à présent concernant vos différents flux ? Voit-on des messages sur le réseau prévenant les utilisateurs ?

Partie IV : Règles de filtrage

Bon c'est malin tout ça, mais si nous avons un réseau avec de l'IP c'est pour pouvoir communiquer. Il faut à présent mettre en place des règles pour cela. L'objectif est de lister tout ce que l'on souhaite pouvoir faire sur son réseau.

Ici nous allons nous limiter à trois types de flux :

- Autoriser le protocole ICMP partout ;
- Permettre au réseau DMZ de se connecter sur un serveur web du réseau privé ;
- Permettre au réseau DMZ de faire du DNS en UDP avec tous les réseaux publics.

Le principe d'iptables reste le même que dans le TP précédent :

```
iptables -t <type> -A/I/L/D/R/F <CHAINE> <Conditions> -j <ACTION>
```

Le –t est optionnel car par défaut c'est filter. L'action sera ACCEPT vu que la politique par défaut est REJECT ou DROP.

> ***Exercice 6 : Règle ICMP***
> Ajouter votre première règle dans le FORWARD, pour que protocole ICMP soit utilisable entre tous les réseaux et vérifier cela grâce à la commande ping.

> ***Exercice 7 : Règle HTTP***
> Ajouter les règles dans le FORWARD, pour la règle HTTP. Pourquoi une seule règle n'est-elle pas suffisante contrairement à la règle ICMP ? Observer que votre règle fonctionne grâce à iperf3.

> ***Exercice 8 : Règle DNS***
> Ajouter les règles dans le FORWARD, pour la règle DNS. De combien de règles a-t-on besoin ici ? Observer que votre règle fonctionne grâce à iperf3.

Pour aller plus loin :
Dans cette partie optionnelle, nous allons considérer un serveur web sur le réseau privé. Malheureusement les adresses du réseau privé sont devenues privées et non routable sur le routeur d'inteco. (Il n'est pas besoin de changer les adresses, il suffit de supprimer les routes vers les réseaux 50.50.X.0/24 sur le routeur d'interco). De ce fait, on utilise comme adresse publique du serveur web, une adresse non utilisée du réseau publique de la DMZ

> ***Exercice 9 : DNAT***
> En utilisant le DNAT, rediriger le trafic web vers le bon serveur. Faites très attention à la spécification de la règle. Est-ce suffisant ? Pourquoi ? Changer donc les règles du pare-feu si besoin.