Ensimag 1ère année

TP n°5 – Performance & pare-feu

Il est recommandé de prendre des notes. Se référer à la version numérique de ce document pour les liens (sur Chamilo).
Une question sur l'effet ou l'utilisation d'une commande ?
man nom_de_la_commande !

1 Performances

Démarrage de la séance : Au démarrage de l'ordinateur, sélectionnez *FreeBSD restore*, qui réinstallera une version réinitialisée de FreeBSD. L'ordinateur va redémarrer, sélectionnez maintenant *FreeBSD*. Suivant les salles dans lesquels vous êtes, on vous demandera un login : entrer `root`, éventuellement mot de passe : `root./`

Nous allons ici nous intéresser à l'évaluation des performances sur un réseau IP. Deux grandeurs vont particulièrement nous intéresser : le débit et la latence. Pour quelques questions vous serez amené à utiliser 4 machines, pensez donc à vous coordonner avec vos voisins.

1.1 Étude du débit

Pour tester le débit des liens entre vos machines, nous allons utiliser des utilitaires nommés `udptarget` et `udpmt`. Le premier sert de serveur : il ouvre un socket UDP sur le port 13000, attend des données du client, et affiche la quantité de trafic reçu. Le second sert de client : il ouvre un socket UDP vers sa cible sur son port 13000, envoie un maximum de trafic, et affiche quelques statistiques.

1.1.1 Étude du débit théorique

On essaye d'estimer le débit des liens ethernet de la salle de TP dans laquelle vous êtes, à l'aide de la formule détaillant la latence :

$$T_{latence} = T_{propagation} + T_{mission} + T_{attente}$$

Pinguez la machine voisine. Notez le temps de latence (vous prendrez la valeur minimale sur une dizaine d'échantillons : c'est le meilleur cas que nous voulons observer). Recommencez avec un gros paquet (quelques dizaines de kilo-octets).

Q 1 — *À l'aide de ces deux valeurs, calculez le débit du lien. Comparez cette valeur à la configuration actuelle de votre interface en utilisant la commande* `ethtool <interface>`.

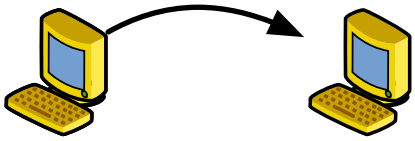

FIGURE 1 – Envoi de messages d'une machine vers une autre.

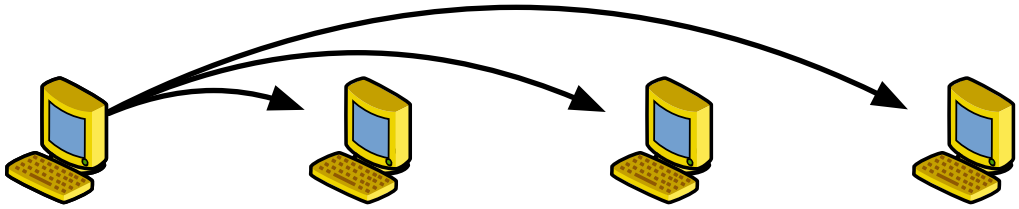

FIGURE 2 – Envoi de messages d'une machine vers trois autres.

Afin de ne pas surcharger le réseau de l'Ensimag, configurez les interfaces de chaque machines afin qu'elles limitent leur débit à 10Mbps :

```
# ifconfig em0 media 10baseT
```

Vous veillerez par ailleurs à générer du trafic UNIQUEMENT vers les machines testées et à ne pas le faire sortir de votre salle. Essayez de garder vos mesures courtes et d'arrêter la génération de trafic après celles-ci.

1.1.2 Cas simple

Vous allez générer du trafic UDP entre deux machines à l'aide d'`udpmt` et d'`udptarget` comme sur la topologie 1.

Sur une des machines, démarrer le serveur en attente de trafic UDP à l'aide de la commande `udptarget`. Sur l'autre machine envoyez du trafic avec `udpmt <dst@>`.

Q 2 — *Quel débit observez vous ? Comparez cette valeur au résultat précédent.*

1.1.3 1 vers 3

Nous allons maintenant utiliser quatre machines. Vous allez envoyer du trafic d'une machine vers les trois autres comme proposé sur la figure 2.

Q 3 — *Quels débits vous attendez-vous à observer sur chacune des machines ?*

Lancez `udptarget` sur chacune des machines allant recevoir du trafic. Sur la machine émettrice lancez 3 `udpmt` en direction de chacune des cibles.

Q 4 — *Quels débits observez vous sur chacune des machines ? Pourquoi ?*

1.1.4 3 vers 1

Nous allons inverser la topologie précédente et envoyer du trafic depuis trois machines vers une seule cible comme indiqué sur la figure 3.

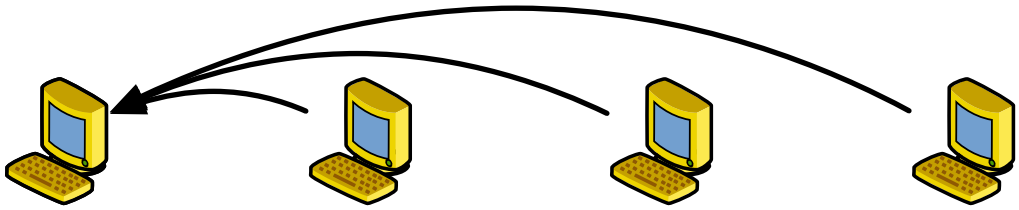

FIGURE 3 – Envoi de messages depuis trois machine vers une seule.

Q 5 — *Quels débits vous attendez-vous à observer sur chacune des machines ?*

Lancez trois `udptarget` sur la machine cible. Attention, vous allez devoir indiquer 3 ports d'écoute différents en utilisant l'option `-p`. Lancez maintenant un `udpmt` sur chacune des trois machine. De la même manière que pour la cible vous allez devoir préciser le port que vous souhaitez utiliser.

Q 6 — *Quels débits observez vous sur chacune des machines ? Pourquoi ?*

1.2 Étude de la latence

1.2.1 Première approche

Pinguez, l'une après l'autre, les machines `berkeley.edu`, `google.fr`, `pcserveur.ensimag.fr` et `localhost`. Notez les différents RTT.

Q 7 — *Y a-t-il une différence entre ces 4 valeurs ? Pourquoi ?*

Nous allons augmenter virtuellement la latence pour atteindre une des machines. Sélectionnez l'une d'entre elle parmi celle sur votre banc (que nous appellerons dans le sujet, pour simplifier, la machine `pétaouchnok` ; *c.f.* le wiktionnaire pour ceux qui ne connaîtrait pas ce mot), et entrez la commande suivante :

```
kldload dummynet
ipfw pipe 1 config delay 500ms
ipfw add pipe 1 ip from any to any
```

Q 8 — *Tentez, depuis* `pétaouchnok`*, de pinguer les autres machines du banc. Y a-t-il une différence par rapport à la valeur précédente ?*

1.2.2 Latence... quelle influence sur le débit

Q 9 — *Supposons que vous voulez télécharger un gros fichier (*e.g. *la distribution Debian, le dernier Star Wars en full-HD), mais que la machine qui possède ce fichier se trouve « très loin », c'est-à-dire que la communication avec elle est soumise à une forte latence. Est-ce que cela devrait changer le temps de téléchargement ?*

Q 10 — *Générez du trafic entre* `pétaouchnok` *et une autre machine à l'aide d'*`udpmt` *et d'*`udptarget`*. Notez-vous une différence de débit par rapport à celui constaté dans la partie 1.1.2 ? Pourquoi ?*

1.2.3 Latence... quelle influence sur l'interactivité ?

Q 11 — *Supposons que vous voulez regarder les jeux olympiques d'hiver de 2018 depuis la France (le débit est supposé suffisant pour la qualité requise). Est-ce que la grande latence engendrée par la distance qui vous sépare de Pyeongchang vous gênera ? Pourquoi ?*

Q 12 — *Supposons que vous êtes en vidéoconférence avec un potentiel futur employeur de Rio de Janerio (le débit est supposé suffisant pour la qualité requise). Est-ce qu'une grande latence engendrée par la distance qui vous sépare du Brésil vous gênera ? Pourquoi ?*

Depuis une machine du banc, connectez-vous à `pétaouchnok` en SSH (*c.f.* `man ssh` si vous ne savez pas comment expliciter le port à utiliser).

Q 13 — *Essayez d'interagir avec la ligne de commande. Que remarquez-vous ? Comment pouvez-vous l'expliquer ?*

2 Etude d'un pare-feu sous Linux

Pour cette partie vous allez devoir utiliser une machine virtuelle accessible depuis CentOS. Vous devez donc redémarrer votre machine sous CentOS.

Manipulations en machine virtuelle

Dans la suite, il est nécessaire de manipuler en `root`. Vous allez donc manipuler dans une machine virtuelle (VM).

Sauf indication contraire dans la suite du sujet de TP, toutes les manipulations doivent être effectuées dans cette machine virtuelle.

Pour lancer la VM, la commande à utiliser est :

```
lance-vm-3MMIRC5.sh
```

Puis sélectionner « use a temporary machine folder » (option 1). La machine virtuelle démarre alors sur une interface graphique similaire à celle des PC Ensimag (Gnome 3). Aucun mot de passe n'est demandé, et vous êtes automatiquement connecté à l'interface graphique avec l'utilisateur `root`. Il ne vous reste donc plus qu'à ouvrir un terminal pour pouvoir manipuler en `root`.

Si jamais vous avez besoin du mot de passe de la VM, par exemple pour vous connecter en SSH, le mot de passe associé au compte `root` est « `root./` ».

L'objectif de cette partie de TP est de comprendre le fonctionnement d'un pare-feu (*firewall*), à travers l'usage d'`iptables` sous Linux.

Note : il faut être `root` pour configurer le pare-feu. Continuez donc de travailler dans le VM.

2.1 Exploration

Le but d'un pare-feu est de classifier les paquets qui entrent et sortent d'une machine, puis d'appliquer des *actions* sur chaque paquet, selon des règles de politique pré-établies. Généralement, les actions se limitent à laisser passer ou bloquer le paquet, mais il est possible de mettre en place des actions plus complexes.

Dans ce TP, nous nous intéresserons uniquement aux paquets générés ou reçus par la machine locale. Un routeur pourrait également agir comme pare-feu sur les paquets qu'il route, mais ce n'est pas le but du TP.

Q 14 — *Affichez l'état actuel du pare-feu grâce à `iptables -L -n`. A quoi correspond une* chaîne *? Combien y a-t-il de chaînes ? Repérez la politique par défaut pour chaque chaîne (`POLICY`).*

Une bonne pratique de sécurité consiste à bloquer par défaut tous les paquets, et n'autoriser que les types de paquets souhaités.

Q 15 — *Changez la politique par défaut des chaînes `INPUT` et `OUTPUT` de façon à bloquer par défaut les paquets entrants et sortants :*

```
# iptables -P INPUT DROP
# iptables -P OUTPUT DROP
```

2.2 Premières politiques de pare-feu

Nous allons commencer par autoriser le protocole ICMP, puisqu'il est indispensable pour gérer les cas d'erreurs et pour pouvoir tester le fonctionnement d'un réseau. Pour cela, nous allons utiliser l'option `-A` d'`iptables` pour rajouter (*Append*) des règles :

```
# iptables -A INPUT -p icmp -j ACCEPT
# iptables -A OUTPUT -p icmp -j ACCEPT
```

Q 16 — *Ajoutez ces règles et testez un `ping` vers `129.88.47.4`. Cela fonctionne-t-il ? Et un ping vers `delos.imag.fr` ? Expliquez la différence de comportement.*

Q 17 — *Obtenez des statistiques sur le pare-feu grâce à `iptables -L -n -v`. Vérifiez que plusieurs paquets ICMP ont utilisé les règles rajoutées précédemment, et que plusieurs paquets ont été bloqués. Dans quel(s) cas la politique par défaut est-elle utilisée ?*

Nous aimerions également autoriser l'utilisation de DNS, qui fonctionne avec le protocole UDP sur le port 53. Pour cela, ajoutez la règle suivante :

```
# iptables -A OUTPUT -p udp --destination-port 53 -j ACCEPT
```

Remarquez la nouvelle option `--destination-port`, qui est spécifique à UDP. Les options de base d'`iptables` sont décrites dans `man iptables`, tandis que les options spécifiques à chaque protocole (par exemple `--destination-port` ici) sont décrites dans `man iptables-extensions`.

Q 18 — *Est-ce que le ping vers `delos.imag.fr` fonctionne mieux ? Si non, quelle pourrait être la cause ? Utilisez Wireshark et `iptables -L -n -v` pour essayer de comprendre d'où vient le problème.*

Q 19 — *Ajoutez la règle manquante pour résoudre le problème.* ***Indice :*** *il est nécessaire d'utiliser une nouvelle option spécifique au protocole UDP.*

2.3 Utilisation du champ ToS

Les règles utilisées jusqu'ici examinaient le protocole (UDP, ICMP), et également le port de destination dans le cas d'UDP. Il s'agit du cas d'usage le plus classique, mais il est également possible de prendre des décisions en fonction de n'importe quel champ de l'en-tête d'un paquet.

Q 20 — *Lancez un `ping` et observez les paquets correspondants dans Wireshark. Repérez notamment le champ "Differentiated Services Field" : dans quel en-tête se situe ce champ ? (Ethernet, IP, ICMP...) Quelle est la valeur de ce champ dans le cas de `ping` ?*

Ce champ, anciennement appelé "ToS" (Type of Service), est normalement utilisé pour différencier des classes de paquets. Il permet de prioriser certains types de paquets par rapport à d'autres (principe de QoS, pour "Quality of Service"). Ici, nous allons simplement nous servir de ce champ pour autoriser ou bloquer certaines classes de paquets.

Q 21 — *Remplacez la règle de sortie ICMP comme suit, la nouvelle règle n'acceptant que la valeur ToS `0xB8` :*

```
iptables -D OUTPUT -p icmp -j ACCEPT
iptables -A OUTPUT -p icmp -m tos --tos 0xB8 -j ACCEPT
```

Q 22 — *Est-ce que* `ping` *fonctionne toujours ? Trouvez l'option de* `ping` *permettant de choisir le champ ToS, et vérifiez que cette option permet à* `ping` *de passer la règle de pare-feu. Vérifiez avec Wireshark que le champ "Differentiated Services" est bien mis à la bonne valeur.*

Remettre le pare-feu dans son état d'origine :

```
iptables -F OUTPUT
iptables -A OUTPUT -p icmp -j ACCEPT
iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
```

2.4 Pare-feu à état (*stateful*)

Jusqu'ici, les règles de pare-feu manipulées étaient *sans état* (stateless), c'est à dire qu'elles se basent uniquement sur l'en-tête des paquets. Il est souvent utile d'introduire un *état*, c'est à dire que le traitement d'un paquet dépendra d'un événement passé.

Dans un premier temps, nous allons utiliser un module qui essaie d'identifier des *flots* de paquets, en se basant sur les IP source et destination, le protocole, les ports source et destination, etc. Lorsqu'un nouveau paquet arrive, le pare-feu essaie de le relier à un flot existant.

Q 23 — *Commencez par ajouter une règle autorisant les paquets appartenant à un flot existant :*

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Ces règles permettent de simplifier grandement la gestion d'un pare-feu, comme nous allons le voir.

Q 24 — *Autorisez les connexions SSH sortantes en ne rajoutant qu'une seule règle supplémentaire. Testez en vous connectant à* `pcserveur.ensimag.fr` *en SSH, avec votre utilisateur Ensimag, puis déconnectez-vous. En vous appuyant sur* `iptables -L -n -v`*, combien de paquets ont utilisés la règle que vous venez d'ajouter ? Selon vous, de quel(s) paquet(s) de votre connection TCP s'agit-il ? Quelle(s) règle(s) sont appliquée(s) aux paquets suivants de votre connection TCP, et dans quelle(s) chaîne(s) ?*

Q 25 — *Observez la sortie de* `conntrack -L`*. Pourquoi parle-t-on de pare-feu à "état" ?*

Nous allons maintenant aborder un autre module qui utilise également un état.

Q 26 — *Ajoutez la règle suivante :*

```
iptables -I OUTPUT 1 -p udp --dport 53 -m limit --limit 3/minute --limit-burst 3 -j LOG
```

Q 27 — *Que fait cette règle ? Vous pouvez observer la sortie de* `dmesg` *après avoir généré du trafic réseau.*

2.5 Sécuriser un serveur

Dans cette partie, on souhaite fournir certains services réseau au monde extérieur, sans pour autant autoriser toutes les connexions entrantes.

Rappel : sur votre PC Ensimag, une redirection de port a été mise en place : le trafic envoyé sur l'hôte sur le port 2222 est redirigé vers la VM sur le port 22.

Q 28 — *Depuis votre PC Ensimag, essayez de vous connecter en SSH sur* `localhost` *sur le port* 2222. *Quel est le résultat ? Sur la VM, voyez-vous de l'activité réseau ?*

Q 29 — *Ajoutez une règle* `iptables` *autorisant les connexions SSH en entrée, et testez qu'elle fonctionne. Vous pouvez profiter des règles* stateful *ajoutées dans la partie précédente. N'oubliez pas de vous déconnecter de la session SSH, et enlevez la règle que vous venez d'ajouter.*

On souhaite maintenant appliquer du "rate-limiting", c'est à dire qu'on ne souhaite autoriser qu'un certain nombre de nouvelles connexions SSH dans un intervalle de temps donné. Le but est de prévenir les attaques par force brute qui cherchent à deviner le mot de passe, ce qui est une attaque très courante lorsqu'on expose un serveur SSH sur Internet.

Q 30 — *Quel type de paquet doit-on prendre en compte pour ne filtrer que les ouvertures de connexion SSH ? Mettre en place une règle qui loggue les ouvertures de connexion SSH entrantes (sans les accepter !), et testez son fonctionnement. Votre règle ne doit pas logguer l'intégralité des paquets SSH.*

Q 31 — *Est-ce que le module* `limit` *pourrait être utilisé ici pour limiter le nombre de nouvelles connexions SSH ? Que se passerait-il si deux clients (avec des IP sources différentes) essayaient de se connecter ?*

On souhaite appliquer une limite indépendante pour chaque IP source qui se connecte.

Q 32 — *Consultez la documentation du module* `recent` *dans* `man iptables-extensions`. *Est-ce que ce module semble adapté pour limiter le nombre de nouvelles connexions SSH ? Quelle est l'option permettant de vérifier qu'une adresse IP est dans la liste des adresses à bloquer ?*

Q 33 — *Ajoutez les règles suivantes :*

```
iptables -I INPUT 1 -p tcp --dport 22 -m recent --name SSH --rcheck --seconds 30 --hitcount 3 -j
iptables -I INPUT 2 -p tcp --dport 22 --syn -m recent --name SSH --set -j ACCEPT
```

Q 34 — *Essayez de vous connecter et déconnecter plusieurs fois en SSH sur la VM, depuis votre PC Ensimag. Quelles règles sont appliquées ? Que constatez-vous après plusieurs connexions ? Quel est le contenu de* `/proc/net/xt_recent/DEFAULT` *?*

2.6 Utilisation de règles `pcap`

On souhaite maintenant filtrer les paquets ICMP sur la base du numéro de séquence, utilisé notamment par `ping`. Ce champ est initialement mis à 1 et est incrémenté pour chaque paquet envoyé par `ping`.

Q 35 — *À l'aide d'une capture Wireshark, repérez le numéro de séquence dans l'en-tête ICMP.*

Q 36 — *Trouvez le module* `icmp` *dans* `man iptables-extensions`. *Ce module permet-il de filtrer sur le numéro de séquence ICMP ?*

Nous allons dans ce cas devoir utiliser un mécanisme plus général : les filtres `pcap`. Il s'agit d'un langage généraliste permettant de décrire quels paquets doivent être pris en compte par un programme. Ce langage est notamment utilisé par `tcpdump` pour filtrer l'affichage.

Q 37 — *Lancez tcpdump avec la commande* `tcpdump -n -i eth0` *et lancez un* `ping` *dans un autre terminal de la VM. Que voyez-vous ?*

Pour filtrer la capture et l'affichage, il est possible de passer une expression `pcap` comme dernier argument. Par exemple, pour filtrer par protocole : `tcpdump -n -i eth0 icmp`, ou bien `tcpdump -n -i eth0 tcp port 80`. Il est aussi possible de filtrer par hôte, avec `host <IP>`. Enfin, les expressions peuvent être combinées avec `and` et `or`, et on peut exprimer une négation avec `not`.

Q 38 — *Filtrez l'affichage de `tcpdump` pour voir tous les paquets DNS ou tous les paquets à destination de `129.88.47.4`. Vérifiez le bon fonctionnement du filtre en pingant `www.renater.fr` puis `delos.imag.fr`.*

Pour les curieux, la syntaxe des filtres `pcap` est décrite dans `man pcap-filter`.

Nous allons maintenant pouvoir construire un filtre concernant le numéro de séquence ICMP.

Q 39 — *En vous aidant de Wireshark, quel est l'offset (en octets) du champ qui contient le numéro de séquence, par rapport au début de l'en-tête ICMP ? Notez aussi la taille (en octets) de ce champ.*

La syntaxe `pcap` pour filtrer sur un champ arbitraire d'un protocole donné (ici `icmp`) est la suivante :

```
icmp[offset:size] = valeur
icmp[offset:size] < valeur
# etc
```

Cela consiste à extraire un nombre entier du paquet à la position `offset`, et à le comparer à une valeur fixée. Il est également possible d'effectuer des opérations arithmétiques (addition, division, modulo...).

Q 40 — *Construisez un filtre `pcap` permettant de filtrer les paquets ICMP dont le numéro de séquence est supérieur à 3, et testez son bon fonctionnement avec `tcpdump`.*

`iptables` ne sait pas travailler directement avec des filtres `pcap`, mais accepte des programmes BPF (Berkeley Packet Filter). BPF est une sorte d'assembleur bas niveau pour les filtres de paquets. Nous allons donc compiler nos filtres `pcap` en BPF.

Q 41 — *Que fait le filtre `pcap` suivant ?*

```
icmp[6:2] % 2 = 0
```

Q 42 — *Compilez ce filtre en utilisant* `nfbpf_compile RAW '<filtre pcap>'`*, et ajoutez le programe BPF obtenu à `iptables` en utilisant le module `bpf` :*

```
iptables -I OUTPUT 1 -p icmp -m bpf --bytecode "<code bpf>" -j DROP
```

Q 43 — *Testez le fonctionnement de cette règle en utilisant `ping`.*

Pour les vraiment curieux qui seraient intrigués par les opcodes BPF obtenus, il est possible d'obtenir un programme BPF plus lisible grâce à `tcpdump -d <filtre pcap>`. Pour une introduction à la syntaxe de BPF, voir `man bpfc` et `http://tylerfisher.org/bpf/`.