**TP 5 :**

**T\_Latence** = T\_Propagation + T\_Transmission + T\_ Attente :

**T\_Propagation** = Distance / Vitesse ( cuivre : Vitesse = 2.3×108 m/s verre : Vitesse =2×108 m/s )

**T\_Transmission** = Taille / Débit

**top** (pour afficher les process Unix)

**Calcul du temps de latence par calcul :**

**->** ping ensipcXXX -c 10 (prendre le plus petit temps **t** et le diviser par 2, 2\*T\_latence = **t**)

**->** ping ensipcXXX -s 1000 (prendre le plus petit temps **t’** et le diviser par 2, 2\*T\_latence = **t’**)

**->** On a **t = T\_p + 56/Débit + T\_att (1)**

**t’ = T\_p + 1000/Débit + T\_att (2)**

**-> (2) - (1) =>** On en déduit le débit

**Valeur théorique du temps de latence :**

**->** ethtool <inteface> (trouver avec ifconfig, ex : enp0s31f6)

Pour pas surcharger le réseau de l’Imag, configurez les interfaces de chaque

machines pour qu’elles limitent leur débit à 10Mbps :

**# ifconfig em0 media 10baseT**

**Générer du trafic UDP entre deux machines :**

Sur une machine, démarrer le serveur en attente de trafic UDP : **./udptarget**

Sur l’autre machine envoyez du trafic avec : **./udpmt ensipcXXX**

**->** On trouve la valeur du débit sur l’interface de celui qui envoie le trafic (à peu près 900 Mbit/s)

**Générer du trafic UDP de une machine vers trois autres :**

**->** lancer un **./udptarget** sur chacune des 3 machines réceptrices

**->** lancer dans trois terminals différents un **./udpmt ensipcXXX** qui correspond à chacune des trois machines

**=>** On trouve un débit trois fois plus petit (environ 300Mbit/s) car il est divisé par le nb de machine

**Générer du trafic UDP de trois machines vers une autre :**

**->** Sur la machine réceptrice faire sur trois terminaux différents :

- un : **./udptarget**

- un : **./udptarget -p 1301**

- un : **./udptarget -p 1302**

**->** Sur les trois émettrices faire :

- une : **./udpmt ensipcXXX** (celle de la machine réceptrice)

- une : **./udpmt ensipcXXX -p 1301**

- une : **./udpmt ensipcXXX -p 1302**

**=>** On trouve un débit d'environ 900Mbit/s car comme si c'était entre deux machines, 3 fois.

**Etude le la latence :**

**->** Si on ping différentes machines comme berkeley.edu, google.fr, pcserveur.ensimag.fr et localhost on obtient des temps de latence différents, beaucoup plus grand pour berkeley.edu que pour localhost, c’est du au temps de propagation qui est plus grand.

**[->** Choisir une machine du banc : petaouchnok, qui rentre la commande :

**(kldload dummynet**

**ipfw pipe 1 config delay 500ms**

**ipfw add pipe 1 ip from any to any)**

**(cette commande marche pas mais dans le nouveau sujet…)**

**ancien sujet: tc qdisc add dev eth0 root netem delay 1s (marche pas non plus mais dans le sujet)**

**->** Depuis cette machine pinguer les autres machines du banc

**=>** Le temps de latence augmente !

**->** Générez du trafic entre pétaouchnok et une autre machine à l’aide d’udpmt et d’udptarget

**=>** Le débit augmente de 1sec **] (on est pas sur, marche pas bien, changement de sujet)**

2) Etude d’un pare-feu:

se mettre sur une VM en root pour faire ça

chaîne : liste de règles à appliquer pour un certain type de paquet

iptables -L -n: pour afficher l’état actuel d’un pare-feu

(Changer la policy)

iptables -P INPUT DROP : bloque tous les paquets entrants, sauf ceux acceptés spécifiquement

iptables -P OUTPUT DROP : bloque tous les paquets sortants, sauf ceux acceptés spécifiquement

iptables -A INPUT -p icmp -j ACCEPT : autorise le protocole icmp en entrée

iptables -A OUTPUT -p icmp -j ACCEPT : autorise le protocole icmp en sortie

iptables -L -n -v : obtenir statistique sur le par-feu

rmq : Si on autorise pas le protocole udp , on ne peut pas faire de requête DNS donc on ne peut pas pinger un nom de domaine (ex : delos.imag.fr)

iptables -A OUTPUT -p udp --destination-port 53 -j ACCEPT : autorise protocole udp sur le port 53 en sortie

si on n’ajoute pas la règle pour aussi recevoir les paquets DNS, le ping vers un domaine ne fonctionne toujours pas car on ne peut pas recevoir la réponse du serveur.

règle à ajouter :

iptables -A INPUT -p udp --source-port 53 -j ACCEPT

16) Ca fonctionne car adresse IP donnée explicitement, donc pas besoin de DNS.

Fonctionne pas avec delos.imag.fr car il faut le DNS.

17) 5 paquets ICMP autorisés, 4 paquets autres bloqués.

18) Ping vers delos.imag.fr ne fonctionne toujours pas car il faut autoriser l’entrée de segments UDP venant du port source 53.

Ce sont justement ces segments UDP qui indiquent l’IP correspondant à delos.imag.fr

19) iptables -A INPUT -p udp --source-port 53 -j ACCEPT

20) Le champ « Differentiated Services Field » se situe dans l’en-tête IP. Il vaut 0x00 dans le cas du ping.

22) ping ne fonctionne plus. Il faut faire ping -Q 0xB8 delos.imag.fr

24) Le premier paquet de la connexion TCP (le paquet avec SYN) a utilisé la règle ajoutée. Comme celui-ci a été accepté, les paquets suivants ont été reliés au flot créé par le 1er. Ils sont donc passés par les règles autorisant les paquets appartenant à un flot existant. Ces règles sont dans les chaînes INPUT et OUTPUT.

27) Cette règle établit une limite de 3 segments UDP / minute vers le port 53 et les met dans les logs.

Si cette limite est dépassée, un message log noyau est généré et visible par dmesg.

28) La connexion ne fonctionne pas mais de l’activité réseau est visible sur la VM.

30) iptables -A INPUT -p tcp –dport 22 -m state –state NEW -j LOG

31) (pas sûr) limit n’est pas adapté car il est indépendant de l’IP. Donc si une IP essaie de se connecter et bloque l’accès, une autre IP sera bloquée aussi.

32) recent est adapté pour limiter le nombre de nouvelles connexions SSH. L’option –rcheck permet de vérifier qu’une IP est dans la liste des adresses à bloquer.

34) Après plusieurs connexions, on se fait bloquer.

36) Le module icmp ne permet pas de filtrer sur le numéro de séquence ICMP.

37) tcpdump affiche toutes les trames sur eth0, comme Wireshark.

38) tcpdump -n -i eth0 udp port 53 or dst host [129.88.47.4](https://l.facebook.com/l.php?u=http%3A%2F%2F129.88.47.4%2F&h=ATN3C7ivyy6UZfrk8XjDn6M8oueHr68IIrEYh-q_eiNOtgVLM2p9mxc-7AsdSUrdrypHW8PDa_vAMs5aJsl1hpyVTJsh15hiq-fUms_7T9c)