NE323 – Travaux dirigés sur machine

Table des matières

1 Préparation

1.1 Synthèse de notre internet

Reprenez l'internet de quatre réseaux vu en cours, pour lequel on a alloué des adresses (sans classes) aux réseaux et à chaque machine.

Faites un schéma des 4 réseaux sur une feuille A4, indiquez toutes les adresses (on rappelle qu'on a numéroté les hôtes à partir du plus petit numéro possible et les routeurs à partir du plus grand, et traité les machines dans l'ordre alphabétique).

Notez également sur cette feuille les tables de routage de toutes les machines et routeurs.

1.2 Préparation de la VM et prise en main

Voir les deux vidéos TDM0 accessibles depuis chamilo (outils Liens).

En particulier vous devez télécharger depuis chamilo le projet marionnet tdmNE323.mar. Lancez marionnet et chargez le projet [1]. Vous retrouvez la topologie de l'internet du cours avec ses quatres réseaux (moins la machine C du réseau 1, inutile de surcharger). Certains de ces réseaux utilisent un hub, d'autres un switch.

2 Manipulations sur la couche réseau

2.1 Introduction

Les appareils sont présents et cablés et ont leurs adresses IP (le cas échéant) fixées mais n'ont pas des tables de routage complètes.

Initialement toutes les machines sont éteintes. Nous allons progressivement les démarrer, compléter la configuration, et expérimenter avec le réseau.

Dans la suite, quand on demande d'indiquer quel trafic on pourra observer, efforcez vous de décrire suffisament précisément les encapsulations. Donnez les adresses liaisons source et destination des trames (il sera plus simple d'utiliser les noms "symboliques" utilisés en cours comme La, Lr1 etc), et le contenu des trame. Si une trame contient un paquet IP, indiquez ses adresses source et destination ainsi que ce qu'il contient.

On pourra utiliser une notation de la forme

Lsrc Ldst type	infos des autres couches

ou

Lsrc Ldst IP	IPsrc IPdst	contenu

2.1.1 Quand cela "ne marche pas"

C'est souvent à cause d'une erreur "bête". Si une commande est refusée, examinez le message d'erreur affiché. Vérifiez la commande, est-elle cohérente ?

Si le trafic ne passe pas... Avez-vous allumé toutes les machines nécessaires ? Tapez vous la bonne adresse ? Vérifiez soigneusement les tables de routage, ne vous êtes vous pas trompé dans une adresse ?

2.2 Machine A

On commence par démarrer la machine A. Pour cela, cliquer sur le bouton "Machines" en haut sur la gauche de l'image du réseau, sélectionnez `Start` puis `A`. Un terminal représentant une console branchée sur A se lance. On peut se loguer sur A avec le compte `root` (l'administrateur dans le monde Unix) et le mot de passe `azerty` (mais si !)

On peut alors utiliser :

1. Lancez-le au premier plan dans un terminal, et ne touchez plus à ce terminal... marionnet est quelque peu chatouilleux.

- la commande `ifconfig` pour voir la configuration des interfaces
  - on retrouve associée à l'interface `eth0` son adresse liaison et son adresse IP.
- la commande `route` pour voir la la table de routage
  - on retrouve pour chaque route les colonnes vues en cours :
    - `Destination` pour le réseau,
    - `Genmask` pour le masque du réseau,
    - `Gateway` pour le prochain pas (une route directe est représentée par `*`),
    - `Iface` pour l'interface.

Quelles routes contient la table de routage (confrontez aux tables de routages vues en cours) ?

On démarre `hub1`, le *hub* du réseau 1 (avec le bouton représentant les hubs), et le "sniffer" qui va nous permettre de faire des captures de trafic dans le réseau simulé (avec le bouton *Real world access - Bridge*).

On tape sur A la commande :

```
A$ ping -c 1 192.168.0.2
```

Qu'est censée faire la commande `ping` ? Que va-t-il se passer dans ce cas précis ? Quel trafic devrait-on voir sur le réseau 1 ?

On peut vérifier en faisant une capture. Pour cela, on lance dans la machine virtuelle le logiciel `wireshark`. On peut par exemple sélectionner le workspace 2, y créer un terminal et taper la commande `wireshark &`. On peut alors sélectionner l'interface `br0` (pour *bridge*), elle est "branchée" sur l'appareil "sniffer" dans le réseau simulé) et démarrer une capture, avant de rentrer la commande `ping` sur A.

2.3 Réseau 1

Évidemment, on ne pouvait pas espérer `ping` uer B si elle est éteinte. On la démarre donc, et on réessaie la commande sur A.

Que va-t-il se passer ? Quel trafic devrait-on voir sur le réseau 1 ?

On pourra bien sur vérifier ensuite en observant le trafic grâce à notre *sniffer*.

On se logue sur B et on pingue A.

```
B$ ping -c 1 192.168.0.1
```

Quel trafic devrait-on voir sur le réseau 1 ?

2.4 Réseaux 1 et 2

La connectivité est assurée sur le réseau 1. Notre but est maintenant de rendre la communication possible entre les machines du réseau 1 et celles du réseau 2.

Si sur A on tape la commande

```
A$ ping -c 1 192.168.32.1
```

Que va-t-il se passer ?

Pour ajouter une route indirecte dans la table de routage d'une machine hôte (comme A) on utilise la commande `route` :

```
route add -net reseau/n gw prochainpas
```

On remarque qu'il n'est pas nécessaire de préciser l'interface, qui sera déterminée automatiquement (comment ?)

Pour une route directe on ferait :

```
route add -net reseau/n dev interface
```

Pour supprimer des routes (en cas d'erreur par exemple) utilisez la même commande en remplaçant `add` par `del`. Le plus simple est sans doute de ramener la commande initiale en cherchant dans l'historique (flèche vers le haut) et d'éditer la commande.

Complétez la table de routage de A.

Il est temps de vérifier la configuration du routeur R. On peut commencer par le démarrer, et on peut ensuite se logguer avec le login `quagga` et le mot de passe `quagga`. L'interface ligne de commande des routeurs est un peu différente de celle des hôtes.

On peut vérifier les adresses de R avec la commande `show interface`, ses routes avec `show ip route`. La présentation est un peu différente, prenez le temps de comprendre la table de routage et de retrouver les informations vues en cours (réseau, masque de réseau, prochain pas).

Vous remarquerez une interface `lo` (pour *loopback*) et une route associée vers 127.0.0.0/8 : le réseau *loopback*. Vous pourrez les ignorer superbement.

Confrontez ce que vous voyez avec les tables de routage vues en cours. Avec quels réseau R est-il capable de communiquer ? Est-il pour le moment nécessaire d'ajouter des routes ?

Démarrez maintenant le hub2, puis la machine D et complétez sa table de routage.

On pingue à nouveau D depuis A.

```
A$ ping -c 1 192.168.32.1
```

Que va-t-il se passer ? Quel trafic verra-t-on sur le réseau 1 ? sur le réseau 2 ?

On peut brancher le *sniffer* sur le hub2. Il suffit pour cela de déplacer le cable (bouton *Straight cable/-Modify/d12*).

2.5 Hôte inexistant

Depuis A on tape la commande

```
A$ ping -c 1 192.168.32.2
```

Que va-t-il se passer ? Quel trafic devrait-on observer sur le réseau 1 ? Sur le réseau 2 ?

2.6 Réseaux 1 et 3

Démarrez la machine G et le routeur S. Vérifiez la table de routage de S.

Ajoutez sur A la route permettant de joindre le réseau 3.

Sur A on pingue G

```
A$ ping -c 1 192.168.3.129
```

Quel trafic va-t-on observer sur le réseau 1 ? sur le réseau 2, le 3 ?

2.7 Réseaux 1 et 3 suite

Pour ajouter une route sur un routeur il faut passer en mode configuration avant d'entrer la commande :

```
R# configure terminal
R(config)# ip route reseau/n prochainpas
R(config)# exit
```

Comme vous le voyez la syntaxe est légèrement différente de celle à utiliser sur les hôtes. N'oubliez pas de vérifier l'effet de vos actions avec

```
R# show ip route
```

Vous pouvez aussi sauver la configuration avec `write`. Pour supprimer une route utilisez la même commande que pour l'ajout en ajoutant le mot `no` devant.

Ajoutez les routes nécessaires sur R, S et G.

Sur A on pingue à nouveau G

```
A$ ping -c 1 192.168.3.129
```

Quel trafic va-t-on observer sur le réseau 1 ? sur le réseau 2, le 3 ?

2.8 Traceroute

Sur A exécutez la commande `traceroute -I -q 1 -z 1 192.168.3.129`. Qu'est-ce qui est affiché ? Observez le trafic sur le réseau 1 pour comprendre comment cela fonctionne.

2.9 TTL

Ajoutons des routes pour le réseau imaginaire 1.2.3.0/24 :

- sur A avec R comme prochain pas,
- sur R avec S comme prochain pas,
- sur S avec R comme prochain pas.

Sur A on pingue une machine de ce réseau imaginaire :

```
A$ ping -c 1 1.2.3.4
```

Que va-t-il se passer ? Quel trafic verrait-on sur le réseau 1 ? le 2 ?

2.10 Fragmentation

Déterminez quelle taille minimale de données vous devez placer dans le message ICMP pour qu'un ping de A vers G engendre des paquets trop gros pour traverser le réseau 2. Vous pouvez le déterminer expérimentalement, mais soyez sur de justifier ensuite rigoureusement, par exemple en dessinant les différentes encapsulations de ces données. Voyez la page de manuel de `ping` et en particulier la description de l'option `-s`.

Attention, comme indiqué en cours, par défaut `wireshark` est trop malin et réassemble les paquets IP fragmentés avant de les afficher ! Désactivez ce comportement en décochant `Reassemble fragmented IPv4 datagrams` dans `Edit/ Preferences/ Protocols/ IPv4`.

3 Manipulations sur la couche transport

3.1 Préparation

Parcourez la page de manuel de l'outil réseau `netcat`, pour déterminer l'effet des commandes :

```
netcat 192.168.3.129 9999
netcat -u 192.168.3.129 9999
netcat -l -p 9999
netcat -u -l -p 9999
```

Vérifiez aussi l'option `-v`.

Attention il y a plusieurs versions, similaires mais différentes de `netcat`, le plus sur est de consulter la page de manuel directement sur l'image marionnet (netcat dit "traditionnel").

3.2 UDP

Utilisez `netcat` pour créer un "tchat" au dessus d'UDP entre deux machines de votre internet.

> Quel trafic pourra-t-on observer au lancement de `netcat` ? Puis quand on tapera des données dans le tchat ? Relevez-le précisément.
> Que se passe-t-il si vous tapez des données d'abord sur le "serveur" ? Pourquoi ?

Si vous terminez l'un des `netcat`, (par CTRL-c) du trafic sera-t-il généré ?

> Si vous tentez ensuite de continuer le tchat depuis le netcat restant, les données sont-elles envoyées ? Que se passe-t-il ?

3.3 Bases de TCP

Utilisez `netcat` pour créer un "tchat" au dessus de TCP entre deux machines de votre internet. Dessinez le diagramme d'échanges détaillé (avec les numéros de séquence et d'acquittement) d'une session de tchat complète : ouverture de connexion, échange de quelques messages, fermeture.

3.4 Contrôle de flux TCP

On sait que TCP met en œuvre un contrôle de flux utilisant la notion de "fenêtre".

Une implémentation moderne de TCP utilise un tas de mécanismes optionnels et des tampons dont la taille est gérée dynamiquement par le système. Nous pouvons cependant régler les paramètres de TCP sur une machine linux pour le dépouiller et en rendre ainsi plus visibles les principaux mécanismes.

Sous Linux, ces paramètres peuvent être consultés ou modifiés par l'intermédiaire de pseudo-fichiers dans le répertoire `/proc/sys/net/ipv4/`. Ces "fichiers" ne sont pas éditables mais on peut écrire dedans ou afficher leur contenu. Ici on tapera (en étant `root`, et sur les deux machines) les commandes suivantes :

```
# cd /proc/sys/net/ipv4
# echo 0 > tcp_window_scaling
# echo 0 > tcp_sack
# echo 0 > tcp_timestamps
# echo 1024 1024 1024 > tcp_rmem
```

Les trois premières commandes `echo` écrivent la valeur 0 dans le pseudo-fichier correpondant, ce qui désactive une option de TCP. La quatrième fixe la taille des tampons de réception à une petite valeur fixe, ce qui nous permettra d'observer plus facilement les mécanismes de contrôle de flux.

Pour vérifier la valeur du paramètre on peut utiliser la commande `cat` :

```
# cat tcp_window_scaling
0
```

Refaites le tchat TCP et relevez le trafic, en vous intéressant maintenant aux annonces de fenêtre.

Vous remarquerez que cette implémentation de TCP ne prend pas la peine de mettre à jour systématiquement la taille de la fenêtre après que l'application ait lu les données...

3.5 Pertes

L'onglet `Defects` de l'application marionnet permet de simuler l'existence de défauts dans le réseau. Configurez 20 % de pertes sur le port 1 du switch 3, en entrée et en sortie (*inward* et *outward*).

1. Quel taux de perte un `ping -i 0.1 -c 200` de A vers G indique-t-il ? Pourquoi ?

1. Reprenez votre tchat netcat sur UDP et observez l'effet des pertes de paquet sur l'"application" tchat.
2. Même chose en utilisant TCP. Observez les retransmissions avec wireshark.