3I 023 – TP 2

Configuration IP élémentaire sous NetKit

Semaine du 13 Mars 2017

---

1 Prise en main de Netkit

1.1 Présentation de Netkit

Netkit (`http://wiki.netkit.org`) est un émulateur de réseau. Il permet d'effectuer facilement des expérimentations réseau en déployant un ensemble de machines virtuelles munies de cartes réseau type Ethernet, reliées entre elles via des domaines de diffusion virtuels. La technologie utilisée est celle de user mode linux qui fonctionne sur la plupart des systèmes d'exploitation Linux modernes. En salle de TP, Netkit a été installé pour vous.

1.2 Les commandes en V

Les commandes ci-dessous vous permettront de manipuler les machines virtuelles qui constitueront votre réseau :

`vstart`	Permet de lancer une machine virtuelle équipée d'un certain nombre de cartes réseau reliées à des domaines de diffusion (e.g., `vstart --eth0=hubA pc1`).
`vlist`	Permet de prendre connaissance des machines virtuelles actuellement actives.
`vhalt`	Permet d'arrêter proprement une machine virtuelle (e.g., `vhalt -r pc1`).
`vcrash`	Permet d'arrêter brutalement une machine virtuelle (e.g., `vcrash -r pc1`).
`vclean`	Permet de faire le ménage en cas de problème.
`vconfig`	Permet d'ajouter à la volée une carte réseau à une machine virtuelle.

Pour obtenir plus d'informations sur l'utilisation de commandes NetKit, utilisez les manuels accessibles via la commande `man <commande>`.

Une machine virtuelle se lance à l'aide de la commande `vstart <nom_machine>`. Un terminal apparaît : c'est la console de votre machine.

Une fois une machine démarrée, vous êtes **root**. Testez quelques commandes (`uname -a`, `pwd`), vérifiez que vous accédez à votre répertoire utilisateur hôte via `/hosthome` et arrêtez la machine avec la commande `vhalt`. Pensez à supprimez les fichiers `*.disk` et `*.log` (par la suite, on arrêtera les machines avec la commande `vhalt -f` qui supprime automatiquement ces fichiers).

2 Première architecture minimale : deux hôtes

2.1 Configuration de l'architecture sur les machines virtuelles

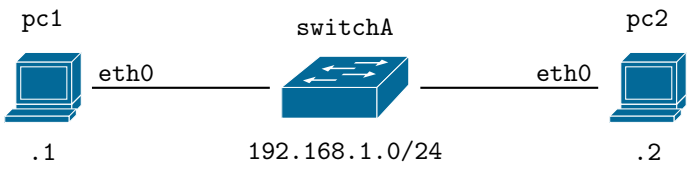

FIGURE 1 – Topologie réseau simple : deux hôtes

1. Grâce à la commande `vstart`, lancez deux machines virtuelles (`pc1` et `pc2`) connectées sur le même domaine de diffusion (nous appellerons ce domaine `switchA`).
2. Grâce à la commande `vlist`, vérifiez que ces deux machines sont correctement lancées.
3. Utilisez le manuel de la commande `ifconfig` qui vous est donné en annexe, afin de configurer (sur le terminal virtuel de chacune des deux machines) les adresses IP des machines virtuelles :
   (a) L'interface `eth0` de `pc1` avec l'adresse `192.168.1.1`.
   (b) L'interface `eth0` de `pc2` avec l'adresse `192.168.1.2`.
4. Vérifiez à l'aide de la commande `ping` `<adresse_IP_destination>` que les deux machines sont capables de s'atteindre mutuellement.
5. Pendant que la machine `pc1` ping `pc2`, observez le trafic ainsi généré sur l'interface `eth0` de la machine `pc2` à l'aide de la commande `tcpdump` (dont le manuel vous est aussi donné en annexe).
6. Ajoutez à la commande `tcpdump` l'option permettant d'observer le contenu des entêtes Ethernet.
7. Arrêtez les deux machines virtuelles à l'aide de la commande `vhalt`.

2.2 Configuration et Sauvegarde de l'architecture via un "Lab"

Vous allez maintenant reproduire l'architecture décrite ci-dessus grâce à un "Lab". Sous Netkit, un **Lab** est un ensemble de scripts permettant le lancement automatique de toutes les machines virtuelles constituant un réseau, et la configuration automatique de ces machines. Les deux principaux avantages par rapport à l'utilisation de commandes en V sont :

— Le démarrage et la configuration plus rapide de l'ensemble des machines virtuelles ;
— La possibilité de sauvegarder une architecture réseau et sa configuration.

Voici les différentes étapes pour créer un Lab Netkit :

1. Créez, dans un dossier `Lab_TP1_Hosts`, le fichier `lab.conf`.
2. Dans le fichier `lab.conf`, vous allez déclarer les machines virtuelles à lancer, ainsi que les interfaces réseaux de ces machines, et les domaines de diffusion auxquelles elles sont raccordées. Pour cela, pour chaque interface, ajoutez une ligne sous le format : `nom_machine[numero_interface]="nom_domaine_diffusion"`

   Par exemple, un fichier contenant les deux lignes ci-dessous créera deux machines `M1` et `M2` connectées au même domaine `D1` :
   ```
   M1[0]="D1"
   M2[0]="D1"
   ```
   Vous pouvez ajouter des descriptions à votre fichier `lab.conf` grâce aux balises suivantes :
   ```
   LAB_DESCRIPTION="Une description du Lab."
   LAB_VERSION=1.2.3.4
   LAB_AUTHOR="auteur 1, auteur 2, auteur 3"
   LAB_EMAIL=name@domaine.org
   LAB_WEB=http://www.unsiteweb.org/
   ```
3. Dans le dossier `Lab_TP1_Hosts`, créez un dossier `<nom_machine_virtuelle>` pour chaque machine virtuelle.
4. Dans le même dossier `Lab_TP1_Hosts`, créez un fichier `<nom_machine_virtuelle>.startup` pour chaque machine virtuelle.
5. Dans chaque fichier `*.startup`, re-écrivez les commandes `ifconfig eth0 ...` qui vous ont permis précédemment de configurer les machines virtuelles.
6. Vous pouvez maintenant lancer votre Lab grâce à la commande `lstart` (sans arguments), et l'arrêter grâce à la commande `lhalt` (sans arguments). D'autres commandes permettent de manipuler les Labs (e.g., `lclean`, `lcrash`, `lhalt`, `linfo`, `lrestart`, `lstart`, `ltest`), vous trouverez plus d'information grâce à la commande `man` `<nom_commande>`.

7. Après avoir lancé votre Lab, vérifiez à l'aides des commandes `ping` et `ifconfig` que les deux machines virtuelles sont bien configurées.

3 Routage statique

Le routage est le mécanisme par lequel des chemins sont sélectionnés dans un réseau pour acheminer les données d'un expéditeur jusqu'à un ou plusieurs destinataires. Chaque composant du réseau, que se soit les hôtes ou les routeurs, possède des informations sur son voisinage dans sa **table de routage**. Dans cette partie, nous allons étudier sur un exemple simple la mise en place et l'utilisation du **routage statique**.

3.1 Mise en place de la topologie réseau étudiée

Nous allons étudier le routage statique à partir d'une topologie réseau simple composée de deux postes client et de deux routeurs. Nous proposons de construire la topologie présenté sur la Figure 2 et de vérifier que la connectivité entre `pc1` et `pc2` est établie.

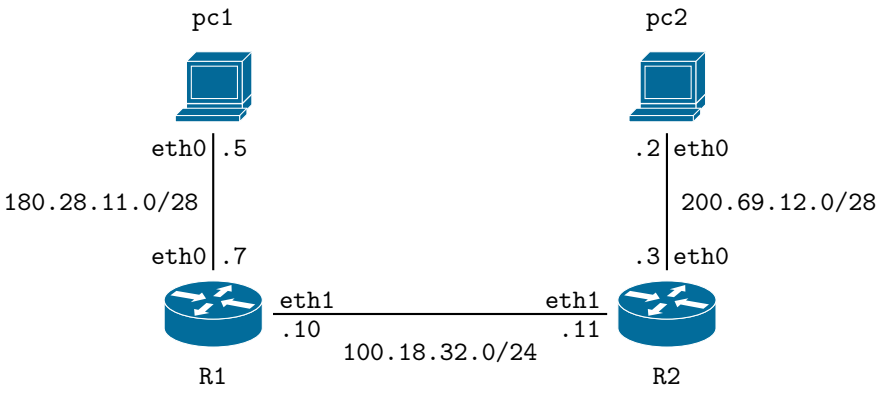

FIGURE 2 – Topologie réseau pour le routage statique

3.2 Création du Lab

1. Créez, dans un dossier `Lab_TP1_Statique`, le fichier `lab.conf`.
2. Dans le fichier `lab.conf`, déclarez les machines virtuelles à lancer, ainsi que les interfaces réseaux de ces machines, et les domaines de diffusion auxquelles elles sont raccordées. Pour cela, pour chaque interface, ajoutez une ligne `nom_machine[numero_interface]="nom_domaine_diffusion "` qui convient.
3. Dans le dossier `Lab_TP1_Statique`, créez un dossier `<nom_machine_virtuelle>` pour chaque machine virtuelle.
4. Dans le même dossier `Lab_TP1_Statique`, créez un fichier `<nom_machine_virtuelle>.startup` pour chaque machine virtuelle.
5. Dans chaque fichier `*.startup`, écrivez les commandes permettant de configurer les interfaces réseaux des machines virtuelles `ifconfig eth0 ...`.
6. Placez vous dans le dossier `Lab_TP1_Statique` et lancer votre Lab à l'aide de la commande `lstart`.

3.3 Diagnostic et configuration du réseau

1. Test de connectivité de `pc1` et `pc2`.
   (a) Connectez vous sur le terminal du `pc1` et lancez la commande `ping` vers les adresses `180.28.11.7` et `100.18.32.10`. Que constatez vous ?
   (b) Connectez vous sur le terminal du `pc2` et lancez la commande `ping` vers les adresses `200.69.12.3` et `100.18.32.11`. Que constatez vous ?
2. Analyse des tables de routage.
   (a) Utilisez la commande `route` sans arguments sur toutes les machines du Lab. Cette commande permet d'observer les tables de routage stockées par défaut sur les machines. Que constatez vous ?
3. Configuration des routes par défaut sur `pc1` et `pc2`.
   (a) Afin de régler le problème observé précédemment, utilisez la commande route avec les arguments nécessaire afin que les paquets générés à partir de `pc1` et `pc2` puissent être envoyés sur les autres réseaux formant le Lab.
   (b) A partir du terminal du `pc1`, vérifiez que l'adresse `100.18.32.10` répond au `ping`.
   (c) A partir du terminal du `pc2`, vérifiez que l'adresse `100.18.32.11` répond au `ping`.
4. Test de connectivité vers `R1` et `R2`.
   (a) A partir du terminal de `pc1`, lancez la commande `ping` vers l'adresse `100.18.32.11`. Que constatez vous ?
   (b) Afin de vérifier le comportement de `R2`, lancez une capture de paquet sur `R2` à l'aide de la commande `tcpdump` avec les arguments nécessaires pour capturer le trafic arrivant de `pc1`. Puis, lancez la commande `ping` sur `pc1` vers l'adresse `100.18.32.11`.
   (c) Regardez les informations capturées par `tcpdump`. Est ce que les paquets `echo request` envoyés par la commande `ping` arrivent sur l'interface `eth1` de `R2` ?
   (d) Arrêtez `tcpdump` sur `R2` en tapant `ctrl+c` et regardez, à l'aide de la commande route sans arguments, les routes présentes dans la table de routage de `R2`. Pouvez vous expliquer pourquoi la commande `ping` lancée sur `pc1` ne reçoit pas de réponses de `R2` ?
   (e) Effectuer les mêmes test à partir de `pc2` vers `R1`.
5. Configuration des routes statiques sur `R1` et `R2`.
   (a) A l'aide de la commande `route`, configurez les routes statiques permettant à `R2` de répondre au `ping` généré par `pc1`.
   (b) De même, à l'aide de la commande `route`, configurez les routes statiques permettant à `R1` de répondre au `ping` généré par `pc2`.
   (c) Vérifiez avec la commande `route` sans arguments le changement des tables de routage sur `R1` et `R2`.
   (d) Après avoir effectué ces changements, testez à l'aide de la commande `ping` que les adresses de `R1` et `R2` sont accessibles depuis `pc2` et `pc1`.
6. Vérification de la connectivité de bout en bout.
   (a) Vérifiez que l'adresse de `pc2` répond bien à la commande `ping` lancée à partir de `pc1`.
   (b) Vérifiez que l'adresse de `pc1` répond bien à la commande `ping` lancée à partir de `pc2`.
   (c) Utilisez la commande `traceroute 200.69.12.2` à partir de `pc1`. Quelle est l'utilité de cette commande ? Comment récupérer à l'aide de la commande `traceroute` toutes les adresses présentes sur les routeurs `R1` et `R2` ?
7. Sauvegarde des modifications de routage.

(a) Afin de réutiliser votre Lab avec les modifications que vous avez effectuées, ajoutez les commandes de configuration de routage que vous avez effectuées sur chaque machines dans leur fichiers respectif `<nom_machine_virtuelle>.startup`.

(b) Vous pouvez maintenant arrêter votre Lab à l'aide de la commande `lhalt -f`.

4 Résolution d'adresse avec ARP

Le protocole de résolution d'adresse (ARP) est un protocole effectuant la traduction d'une adresse de protocole de couche réseau (typiquement une adresse IPv4) en une adresse MAC (typiquement une adresse ethernet), ou même de tout matériel de la couche liaison. Il opère au dessous de la couche réseau et se situe à l'interface entre la couche réseau (couche 3 du modèle OSI) et la couche de liaison (couche 2 du modèle OSI). Il a été défini dans la RFC 826 : *An Ethernet Address Resolution Protocol*.

4.1 Mise en place de la topologie réseau étudiée

Le but de cette partie est de comprendre le mécanisme utilisé par ARP à partir de la topologie présenté sur la Figure 3.

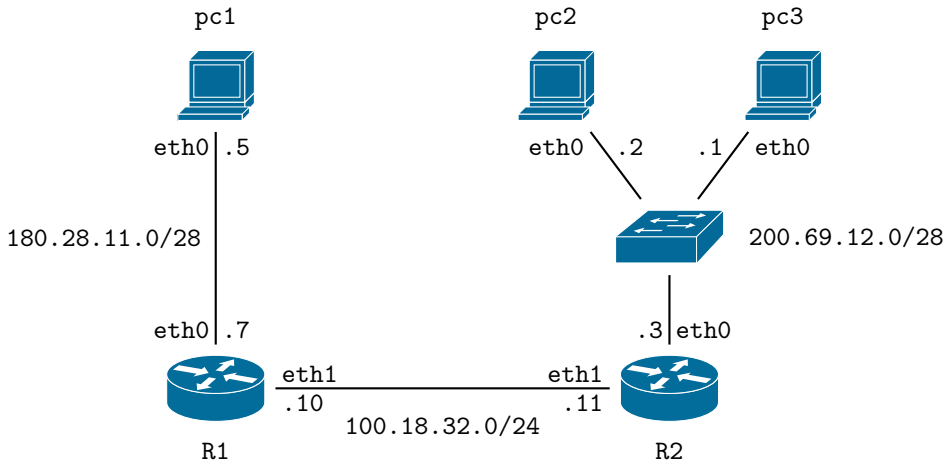

FIGURE 3 – Topologie réseau pour l'étude de ARP

1. Modifiez le Lab précédent `Lab_TP1_Statique` en ajoutant les informations suivantes :
   (a) Rajoutez les informations concernant `pc3` (interface et domaine de collision) dans le fichier `lab.conf`.
   (b) Créez un répertoire vide pour `pc3` et créez un fichier `pc3.startup` avec les informations de configuration nécessaires utilisant les commandes `ifconfig` et `route`.
2. Lancez le Lab avec la commande `lstart`.

4.2 Etude de la résolution d'adresse avec ARP

1. Inspection du cache ARP sur un même sous-réseau.
   (a) Lancez la commande `arp` sans arguments sur le terminal du `pc3` et du `pc2`. Vérifiez que le cache ARP est bien vide.
   (b) A partir du terminal de `pc3`, lancez la commande `ping` vers l'adresse de `pc2`. Ensuite vérifiez l'état du cache ARP sur `pc3` à l'aide de la commande `arp`. Que constatez vous ?

(c) De même, lancez la commande `arp` sur `pc2`. Que constatez vous ?

(d) A partir du terminal de `pc3`, lancez la commande `ping` vers l'adresse de `pc1`. Ensuite vérifiez l'état du cache ARP sur `pc3` à l'aide de la commande `arp`. Que constatez vous ?

(e) Inspectez le cache ARP sur `R1` et `R2`. Que constatez vous ?

(f) A partir des résultats observés précédemment, donnez une explication sur le mécanisme utilisé pour remplir le cache ARP.

2. Capture du trafic ARP.

(a) Arrêtez le Lab avec la commande `lcrash` et relancez le Lab avec la commande `lstart` afin d'effacer les cache ARP.

(b) Nous utiliserons la commande `tcpdump -e -t -i <interface>` pour capturer le trafique ARP. L'option `-e` permet de récupérer les informations au niveau de la couche liaison et l'option `-t` permet de supprimer les informations liées au temps (*timestamp*). Lancer cette commande avec les options citées sur les machines `R1`, `R2` et `pc1`.

(c) A partir du terminal de `pc2`, lancez la commande `ping -c 10` vers l'adresse `180.28.11.5`.

(d) Dès que la commande `ping` est terminée, arrêtez les captures sur `R1`, `R2` et `pc1`. Puis analysez ces captures. Pouvez vous expliquer le comportement du trafique ARP observé sur `R1`, `R2` et `pc1` ?

(e) Effectuez un diagramme permettant d'expliquer le trafic généré pour la résolution d'adresse par ARP et le trafic généré par la commande `ping`.

Commande `ifconfig`

Définition

La commande `ifconfig` est utilisée pour configurer (et maintenir ensuite) les interfaces réseau.

Synopsis

```
ifconfig [interface]
ifconfig interface [aftype] options | adresse ...
```

- Sans arguments, `ifconfig` affiche simplement l'état des interfaces actuellement définies.
- Si seul le paramètre `interface` est donné, il affiche seulement l'état de l'interface correspondante.
- Avec comme seul argument `-a`, cette commande affiche l'état de toues les interfaces, actives ou non.

Principaux arguments

- `interface` précise le nom de l'interface réseau. C'est généralement un nom de pilote suivi d'un numéro d'ordre comme `eth0` pour la première interface Ethernet, `eth1` pour la seconde, ...
- `address` spécifie une adresse IP pour cette interface.
- L'option `up` signifie que l'interface doit être activée (option implicitement positionnée).
- L'option `down` demande au driver de ne plus piloter l'interface précisée.
- L'option `netmask` `<addr>` positionne le masque de réseau pour cette interface. Ce *netmask* est positionné par défaut à une des valeurs habituelles pour réseaux de classes A, B ou C, mais il peut être positionné à n'importe quelle valeur.
- L'option `broadcast` `<addr>` positionne l'adresse de *broadcast* pour cette interface.

Exemples d'utilisation

```
ifconfig eth0 193.70.150.116 netmask 255.255.255.0 broadcast 193.70.150.255
ifconfig eth1 down
```

Commande `route`

Définition

La commande `route` affiche / manipule les tables de routage IP du noyau. Son utilisation première consiste à configurer des routes statiques vers des hôtes ou des réseaux via une interface, après sa configuration par le programme `ifconfig`.

Synopsis

```
route [-Cv]
route add|del [-net|-host] cible [netmask Nm] [gw Gw] [metric N] [dev If]
```

Principaux arguments

- `C` Travaille sur le cache de routage du noyau.
- `v` Active le mode verbeux.
- `del` Supprime une route.
- `add` Ajoute une route.
- `net` La cible est un réseau.
- `host` La cible est un hôte.
- `netmask Nm` Spécifie le masque de réseau de la route à ajouter.

— `metric M` Affecte la valeur `M` au champ métrique de la table de routage (utilisé par les démons de routage).
— `gw Gw` Route les paquets via une passerelle. NOTE : La passerelle concernée doit pouvoir être atteinte. Ceci signifie qu'une route statique vers cette passerelle doit préalablement exister.
— `dev If` Force la route à être associée au périphérique spécifié, sinon le noyau tente de le déterminer par lui-même.

Exemples d'utilisation

```
route add -net 127.0.0.0
route add -net 224.0.0.0 netmask 240.0.0.0 dev eth0
route add default gw mango-gw
route add -net 195.11.14.0 netmask 255.255.255.0 gw 100.0.0.9 dev eth1
```

Commande `tcpdump`

Définition

La commande `tcpdump` permet de faire un *dump* du trafic sur un réseau. A cet effet, `tcpdump` affiche une description du contenu des paquets sur une interface réseau. Il peut également être exécuté avec l'option `-w`, qui enregistre les données dans un fichier pour une analyse ultérieure, et/ou avec l'option `-r`, qui récupère les données à partir du fichier de sauvegarde.

Synopsis

```
tcpdump [ -Ae] [ -B buffer_size ] [ -c count ][ -F file ] [ -i interface ][ -r file ] [ -
    w file ]
```

Principaux arguments

— `A` Version imprimée de chaque paquet (moins son en-tête niveau de la liaison) en ASCII.
— `e` Imprime l'en-tête au niveau des liens sur chaque ligne.
— `B` Règle la taille de mémoire tampon de capture du système d'exploitation.
— `c` Affiche un nombre donné de paquets après la réception.
— `F` Utilise un fichier comme entrée pour les expressions de filtrage.
— `i` Ecoute sur l'interface spécifiée. S'il n'est pas spécifié, `tcpdump` recherche l'interface de numéro le plus bas.
— `r` lit les paquets à partir du fichier (qui a été créé avec l'option `-w`).

Exemples d'utilisation

```
tcpdump -i eth0
tcpdump -i eth0 -w /hosthome/capture.pcap
tcpdump e t i eth0
```