

Travaux pratiques Réseau (Routage) Partie 1 – Ubuntu 16.04

Préambule

Lancez en préalable de toutes manipulations, les commandes suivantes :

Utilisez de préférence la commande sudo, plutôt que la commande sudo -i pour des commandes en mode privilégié.

Arrêt de NetworkManager : `systemctl stop NetworkManager`

Vous pouvez ensuite lancer la commande `nmcli dev status` pour vérifier

Suppression et visualisation des règles du pare-feu

`iptables -X`

`iptables -S`

`iptables -F`

`iptables -L`

Identifiez l'adresse IP de la carte réseau reliée au LAN de la salle. Elle se présente sous la forme 10.192.10.Y.

Demandez à l'enseignant le numéro X de votre binôme (associé à deux machines côte à côte).

Sur les machines deux cartes sont installées : quelles sont leur nom ? Identifiez-les.

Prenez un switch dans l'armoire. Prenez en soin, comme de tout le matériel.

Branchez un câble réseau entre les deux premières entrées puis branchez l'alimentation pour le réinitialiser.

Remarque : pour passer un hôte en mode routeur, privilégiez les commandes `sysctl -w net.ipv4.ip_forward = 1` ou `sysctl net.ipv4.conf.all.forwarding=1`

1.- Réalisation du montage représenté sur la figure 1. En binôme.

En vous aidant du switch, réalisez le montage réseau présenté sur la figure 1. Vous utiliserez les interfaces réseaux situées en bas des tours pour les relier au switch.

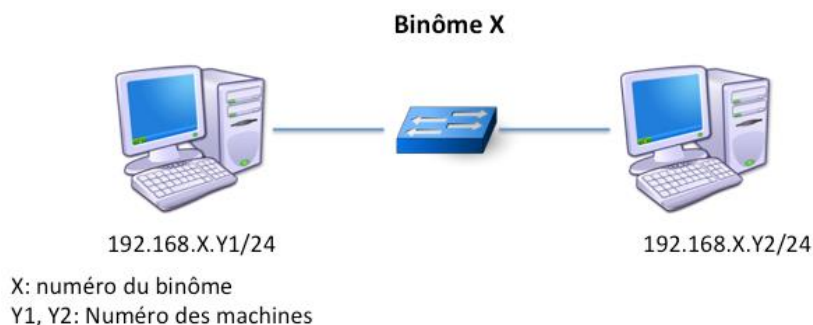


Figure 1. Deux PC connectés par l'intermédiaire d'un switch à leur carte réseau respective

La commande permettant de fixer les adresses IP aux interfaces réseaux est:
`ifconfig [Interface réseau] [Adresse de la carte réseau]/[masque]`

Vérifiez à l'aide de la commande ping la connexion entre les deux interfaces.

2.- Réalisation du montage représenté sur la figure 2. Deux binômes : (4 postes).

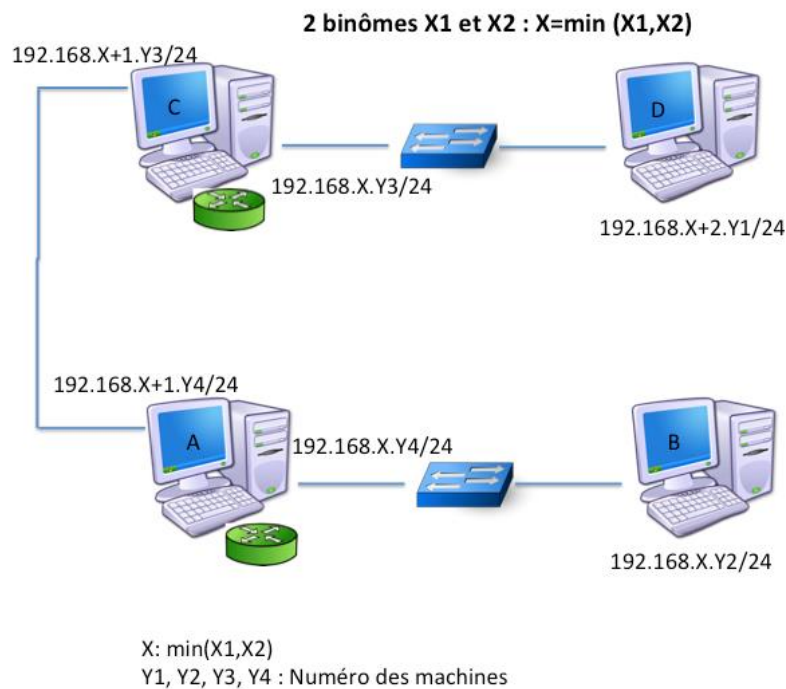


Figure 2. Montage de 4 PC en 3 sous-réseaux. Deux PC sont transformés en routeur.

Vous travaillerez en vous associant à vos collègues voisins autour de 4 machines connectées en 3 sous-réseaux comme il est précisé sur la figure 2. Deux machines seront transformées en routeur : elles seront connectées entre-elles deux via leur interface du haut. Les 4 machines seront déconnectées du réseau.

Construisez les routes de manière à ce que B communique avec D via A et C. Vous utiliserez les commandes

- `ifconfig [Interface réseau] [Adresse de la carte réseau]/[masque]`
- `route add -net [Adresse réseau]/[Masque réseau] gw [adresse IP passerelle]`
- `route -n`

Indication. Procédez par étapes successives. Vérifiez lors de la première étape, en utilisant la commande ping que B et A communiquent. A l'étape 2, que C et D communiquent, puis Etape 3 que A et C communiquent, (Etape 4) que A peut atteindre D, (Etape 5) C peut atteindre B, et enfin que D peut atteindre B.

Validation. Représentez sous la forme d'un schéma comme ceux présentés en cours et en TD les trois sous-réseaux créés. Montrez-le à l'enseignant, ainsi que les tableaux de route pour les postes et les deux routeurs. Demandez-lui de vérifier que votre réseau fonctionne. Aidez-vous du logiciel wireshark.

Dépannage : Déconnectez le câble réseau de C, et reconnectez-le. Quelles sont les lignes de commande que vous devez entrer pour que tout fonctionne à nouveau : D doit se connecter à B.

Demandez à un de vos collègues de débrancher un câble réseau et de supprimer une route (sans vous dire lesquels). Recherchez la panne et corrigez. Faites cela pour chacun d'entre-vous.

Une fois terminé, reconnectez les machines au réseau et rangez switch et câbles. Vérifiez que vos machines accèdent de nouveau au LAN.

Travaux pratiques Réseau (Routage) Partie 2

2.1 Présentation du logiciel Netkit et premier labo !

Netkit est un logiciel libre sous licence GPL. Il permet d'émuler très simplement sur une même machine divers équipements réseau tels que des routeurs, des switches, des serveurs. Chaque équipement réseau émulé possède une ou plusieurs cartes réseau virtuelles, ainsi que son propre espace disque et son propre espace mémoire directement liés au matériel de la machine hôte. L'utilisateur peut lancer ces machines une par une à la main avec les scripts fournis par NetKit, mais il peut également décrire sa propre topologie réseau, appelée « labs » dans le langage NetKit, dans un fichier de configuration. Ainsi l'utilisateur peut demander à NetKit de lire ce fichier afin qu'il crée les équipements virtuels du labs, ainsi que les liaisons entre ces derniers en une seule ligne de commande.

Nous vous conseillons de l'utiliser chez vous car il permet facilement de construire des réseaux complexes sans avoir besoin de ressources importantes (cf. www.netkit.org).

Placez-vous dans le répertoire `/usr/lib/netkit`, et vérifiez la configuration du système en exécutant la commande (en mode root) :

`./check_configuration.sh`

Ce script vérifie la configuration : vous devez obtenir un READY !

Un premier exemple de labo

Considérons le réseau suivant (Figure 3). Vous allez découvrir les deux manières de le configurer.

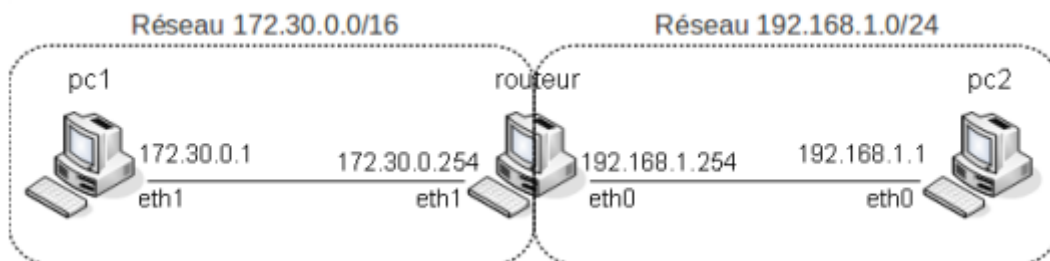


Figure 3. Un premier labo

Pour créer ce réseau sous Netkit, il faut déclarer deux hubs virtuels : le hub A, qui interconnecte pc1 (secret) et routeur, et le hub B qui interconnecte pc2 (public) et routeur. Le schéma est donc équivalent à celui de la Figure 4 :

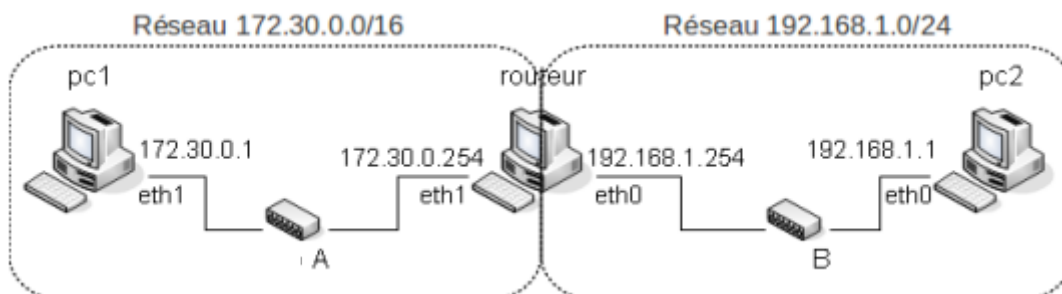


Figure 4. Schéma équivalent du labo précédent

1ère méthode : création directe depuis un terminal

Voici un exemple de commandes, pour lancer deux machines virtuelles dans un même LAN que nous appellerons « A » (remarque : vous pouvez bien-sûr taper les exemples de commande avant de réaliser le réseau précédent) :

```
$ vstart client --eth0=A
```

```
$ vstart pirate --eth0=A
```

Un terminal X est associé à chacune des machines virtuelles (client et pirate) lors de leur création. Chacune dispose d'une interface réseau (eth0) reliée au même hub virtuel du domaine de *broadcast* « A » (c'est-à-dire sur le même domaine de collision). Tous les paquets émis sur celui-ci seront transmis à toutes les interfaces des machines Netkit connectées à ce hub.

Nous disposons maintenant de deux machines virtuelles raccordées toutes les deux au hub virtuel « A ». Il ne reste plus qu'à configurer nos deux cartes réseau avec la commande `ifconfig`: `ifconfig eth0 192.168.1.10/24` par exemple pour pirate et `192.168.1.11/24` pour le client; modifiez le fichier `/etc/hosts` pour enregistrer la correspondance adresse IP-nom hôte et testez la connectivité entre les deux machines.

Pour créer une machine virtuelle directement depuis un terminal avec trois interfaces, la syntaxe est donc la suivante :

```
vstart nom_machine --ethx=A --ethy=B --ethz=C
```

Ici, on crée une machine nommée `nom_machine`, qui possède 3 interfaces : `ethx` est reliée au hub A, `ethy` au hub B, `ethz` au hub C.

Effectuez un `man vstart` et un `man vhalt` pour plus d'information. Arrêtez les deux machines créées à partir des exemples.

En utilisant ce qui précède, créez le laboratoire de la figure 4 (vous donnerez 512 Mo de mémoire à la machine routeur (utilisez l'option `--mem=512` ; la mémoire affectée est 16Mo par défaut)

Trois terminaux s'ouvrent (cf. figure 5) : ils s'appellent respectivement `secret` (pc1), `public` (pc2) et `routeur`. Ce sont les terminaux de chacune de vos machines virtuelles. Toutes les commandes que vous y taperez s'exécuteront sur vos machines virtuelles (et non sur votre machine hôte !). Vérifiez après avoir fait le nécessaire la connectivité entre les machines `public` et `secret`.

Pour arrêter une machine, il faut taper `vhalt nom_machine`. Attention ! Après cette commande, toutes les configurations effectuées sur la machine sont perdues. Il est donc déconseillé d'utiliser cette méthode dans le projet. **Arrêtez maintenant les machines.**

Remarque :

Il est également possible de remplacer les hubs virtuels par des *switchs* virtuels en modifiant les scripts de lancement de Netkit. **Pour ceci, il conviendra d'éditer le fichier `bin/script_utils` (dans `netkit`) et de remplacer la ligne suivante (faire une recherche de `-hub` ou `-tap`):**

```
HUB_COMMAND="$NETKIT_HOME/bin/uml_switch -tap $TAP_DEVICE -hub -unix $1 </dev/null 2>&1"
```

par

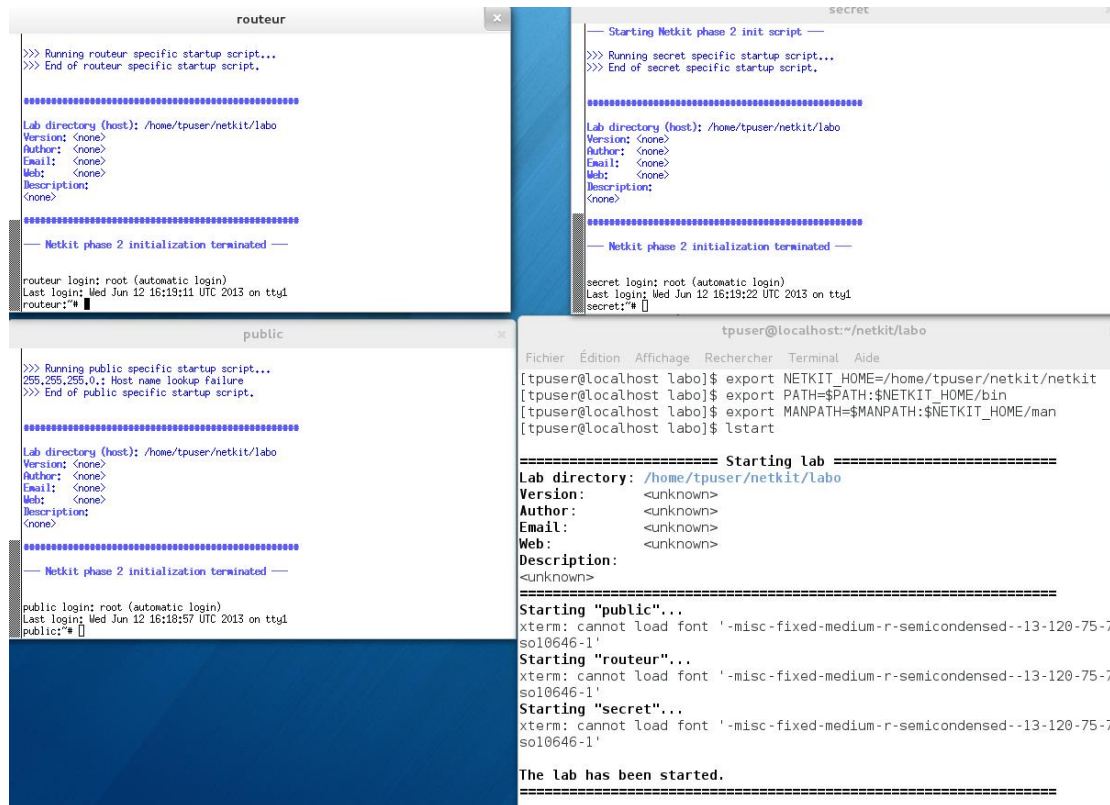
```
HUB_COMMAND="$NETKIT_HOME/bin/uml_switch -tap $TAP_DEVICE -unix $! </dev/null 2>&1"
```

et la ligne suivante

```
HUB_COMMAND="$NETKIT_HOME/bin/uml_switch -hub -unix $! </dev/null 2>&1"
```

par

```
HUB_COMMAND="$NETKIT_HOME/bin/uml_switch -unix $! </dev/null 2>&1"
```



```
tpuser@localhost:~/netkit/labo
[tpuser@localhost labo]$ export NETKIT_HOME=/home/tpuser/netkit/netkit
[tpuser@localhost labo]$ export PATH=$PATH:$NETKIT_HOME/bin
[tpuser@localhost labo]$ export MANPATH=$MANPATH:$NETKIT_HOME/man
[tpuser@localhost labo]$ lstart

===== Starting lab =====
Lab directory: /home/tpuser/netkit/labo
Version: <unknown>
Author: <unknown>
Email: <unknown>
Web: <unknown>
Description: <unknown>

Starting "public"...
xterm: cannot load font '-misc-fixed-medium-r-semicondensed--13-120-75-7
so10646-1'
Starting "routeur"...
xterm: cannot load font '-misc-fixed-medium-r-semicondensed--13-120-75-7
so10646-1'
Starting "secret"...
xterm: cannot load font '-misc-fixed-medium-r-semicondensed--13-120-75-7
so10646-1'

The lab has been started.
```

Figure 5. Création du labo de la figure 2

2ème méthode : utilisation d'un fichier lab.conf

Il est aussi possible de décrire la structure du réseau dans un fichier nommé lab.conf. C'est ce que vous ferez d'une manière générale dans vos montages. La description d'une machine dans le lab.conf se fait de la manière suivante :

Nom_machine[0]=A

Nom_machine[1]=B

Nom_machine[2]=C

Ces lignes créent une machine appelée nom_machine, qui comporte 3 interfaces, eth0, eth1 et eth2, qui sont respectivement connectées aux switches A, B et C.

Ecrivez le fichier lab.conf dans un dossier que vous nommerez labo dans votre répertoire de travail qui permet de simuler le réseau de la Figure 4 (pc1 sera appelé secret, et pc2 public comme précédemment).

Il est nécessaire de créer dans le répertoire contenant le fichier lab.conf des répertoires au nom des machines virtuelles. Ici, on doit donc créer 3 répertoires nommés : secret, public et routeur. Attention : la casse et l'orthographe des noms des répertoires doivent être identiques aux noms des machines décrites dans le fichier lab.conf.

Une fois les répertoires et le fichier lab.conf créés, on démarre les machines en tapant **lstart** dans le répertoire contenant le lab.conf.

Pour arrêter les machines, il est conseillé de lancer dans chacun des terminaux, **la commande shutdown -h now ou halt. Puis un lhalt dans le terminal de l'hôte.** Attention ! Hormis les répertoires au nom des machines, le fichier lab.conf et les fichiers *.startup (voir juste après), vous ne devez placer dans le répertoire de travail aucun fichier ni répertoire.

Il est possible d'exécuter des commandes au démarrage des machines. Pour cela, il suffit de créer un fichier nom_machine.startup dans le même répertoire que le fichier lab.conf et d'y inscrire les commandes à exécuter. Il est notamment intéressant d'y placer les commandes de configuration des paramètres IP et de la table de routage.

Par exemple, le contenu du fichier secret.startup serait :

```
ifconfig eth1 172.30.0.1 netmask 255.255.0.0 up
route add default gw 172.30.0.254
```

Il faut toujours terminer le fichier par un retour à la ligne pour que la dernière commande soit exécutée.

Remarques :

- Par défaut, vous êtes connecté en root sur la machine virtuelle. Le mot de passe du compte root est root.
- Au redémarrage des machines, les modifications réalisées dans les fichiers ne sont pas perdues.

Ecrivez les fichiers "nom_machine.startup" pour affecter les adresses IP aux interfaces réseaux et les routes des machines. Routeur est une machine de type routeur ! Démarrez le réseau. Vérifiez qu'il est bien nécessaire de configurer le mode routeur pour que la communication entre secret et public fonctionne.

Ecoute des paquets réseaux

Chaque machine virtuelle possède un répertoire /hosthome qui est lié à votre répertoire de votre machine hôte (ici /root). Ainsi, tous les fichiers placés dans /root sur la machine hôte sont accessibles depuis les machines virtuelles par le répertoire /hosthome, et inversement. Ceci vous permet de transférer simplement des fichiers entre l'hôte et les machines virtuelles.

En particulier, vous pouvez enregistrer les captures tcpdump dans un fichier que vous placerez dans /hosthome et que vous ouvrirez dans Wireshark lancé depuis votre machine hôte. La ligne de commande à taper sur la machine virtuelle est donc :

```
tcpdump -w /hosthome/capture.cap.
```

Votre répertoire /root sur la machine hôte contiendra le fichier capture.cap qu'il suffit d'ouvrir dans Wireshark. Vous bénéficiez ainsi de l'interface graphique pour l'analyse de vos captures.

Remarque

Pour analyser le trafic sur l'interface eth0 d'une machine, il suffit de taper : `tcpdump -i eth0`
Le trafic capturé est alors affiché directement dans le terminal.

Pour enregistrer le trafic d'une machine virtuelle dans un fichier lisible par Wireshark sur la machine hôte, il suffit de taper la commande `tcpdump -i eth0 -w /hosthome/capture.cap`

Le trafic est enregistré dans le fichier capture.cap que vous trouverez dans votre répertoire personnel sur la machine hôte. Vous pouvez alors l'ouvrir avec `tcpdump -r /hosthome/capture.cap` ou avec Wireshark sur l'hôte..

Pour pouvoir écrire des commandes dans le terminal tout en capturant le trafic, ajoutez un & à la fin de la commande tcpdump :

```
tcpdump -i eth0 -w /home/capture.cap &
```

Avant d'ouvrir le fichier dans Wireshark, il faut arrêter le processus tcpdump. Pour cela, exécutez la commande : `killall tcpdump`

P.S. Si Wireshark n'est pas installé : `yum search wireshark` (et installez ensuite l'intégration bureau GNOME correspondant à votre hôte).

Capturez, via le routeur, les paquets transmis par la commande ping de secret vers public. Visualisez ces paquets à l'aide de Wireshark (voir annexe).

Arrêtez le réseau et fermez les fenêtres des machines virtuelles.

2.2 Connexion vers l'extérieur

Pour accéder à l'extérieur, nous créons un routeur disposant d'une interface `eth1` sur le switch virtuel « A », d'une interface `eth0` sur le switch virtuel « B », et d'une interface `eth2` (interface TAP du système hôte), afin de faire communiquer l'ensemble des machines virtuelles vers l'extérieur et donc de pouvoir installer de nouveaux paquets.

Remarque : Interfaces TAP

La création d'une machine virtuelle disposant d'une interface TAP nécessite que l'utilisateur dispose des droits root sur le système hôte.

Exemple (ne pas réaliser) :

La commande suivante `vstart routeur --eth1=tap,192.168.0.1,192.168.0.2 --eth0=A`

permet de créer :

- sur l'hôte une interface `nk_tap_root` avec l'adresse IP 192.168.0.1,
- une interface `eth1` avec l'IP 192.168.0.2 sur la machine virtuelle routeur qui pourrait maintenant joindre l'extérieur.

Netkit a en fait effectué les opérations suivantes sur le système hôte:

- ajout d'une règle de translation d'adresse;
- ajout d'une règle autorisant les flux émis par notre interface TAP;
- activation du routage;
- une règle de routage sur le routeur précisant que la passerelle par défaut est l'interface TAP du système hôte.

Il ne reste plus qu'à finir de configurer le DNS sur le routeur (fichier `/etc/resolv.conf`), son adresse interne (disons 172.30.0.250) et de s'assurer que le routage est activé (il l'est par défaut sur toutes les machines Netkit).

Cela ne suffit pas pour que les machines autres que le routeur puissent joindre l'extérieur. Le système hôte sera en effet incapable de router les paquets retour. Il n'a en effet aucune connaissance du plan d'adressage Internet des machines Netkit (dans le cas de l'exemple 172.30.0.0/16). La solution est de réaliser une seconde translation d'adresse au niveau du routeur pour que les paquets qu'il émet vers l'interface TAP le soient avec l'adresse IP 192.168.0.2.

La commande est la suivante : `iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE`

Modifiez la configuration de lab.conf pour que les sous réseaux A et B puissent accéder à l'extérieur (cf.figure 6)

(syntaxe: `routeur[2]=tap,adresse_IP(hote),adresse_IP(routeur)`)).

avec `adresse_IP(hôte)=192.168.0.1` et `adresse_IP(routeur)=192.168.0.2` (2 est le numéro de l'interface de connexion vers l'extérieur)

Ajoutez sur l'hôte la commandes :

`iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -j MASQUERADE`

Ajoutez sur le routeur (tout ce qui sort est masqué)

`iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE`

Configurez le fichier `/etc/resolv.conf` sur toutes les machines.

Lancez le laboratoire et vérifiez que pour l'ensemble des machines virtuelles ainsi que pour l'hôte les interfaces sont correctes. Vérifiez l'accès à l'extérieur à partir de `secret` et `public`.

Remarques : vous devrez lancer le laboratoire en tant que `root`. N'oubliez pas de configurer le fichier `.bashrc` de `root` avec les commandes `export` si vous relancez un terminal. Le firewall doit être désactivé avant la création du labo.

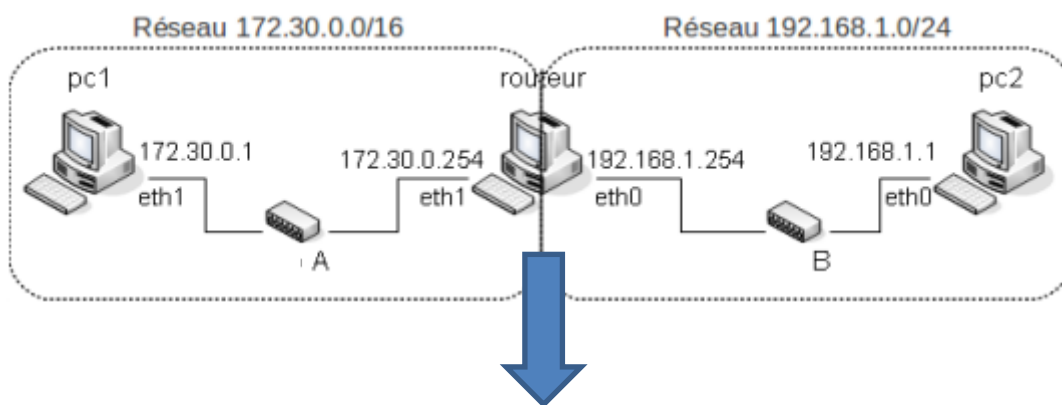


Figure 6. Connexion des sous réseaux vers l'extérieur

2.3 Netkit pour revoir le cours et le TD

Soit le réseau présenté sur la figure 7. Le dossier de configuration (tpl3netE) de ce réseau doit être téléchargé du serveur Moodle.

- Consultez les différents fichiers de configuration, `lab.conf`, et `*.startup`. Puis lancez le laboratoire avec la commande `lstart`.
- Complétez les tables de routage directement sur les terminaux des machines virtuelles de telle façon que les routeurs R2 et R3 puissent communiquer avec R5 et R6. Indication: aidez-vous des commandes `ping` et `traceroute` pour diagnostiquer le problème; vous avez 5 routes à écrire sur deux routeurs. Choisissez les bien !
- Vérifiez avec la commande `traceroute` que pc65 communique maintenant avec pc130 et pc161.

- Modifiez les fichiers *.startup pour prendre en compte ces nouvelles routes. Dans un terminal sur l'hôte, lancez la commande killall netkit-kernel, puis relancez le labo. Vérifiez que tout fonctionne.
- Tapez les deux commandes nécessaires pour que votre réseau puisse voir la salle. La première commande se lancera sur l'hôte, la seconde sur R1. Vérifiez que pc130 peut sortir sur le réseau de la salle. De pc130, connectez-vous à une machine de la salle en ssh.

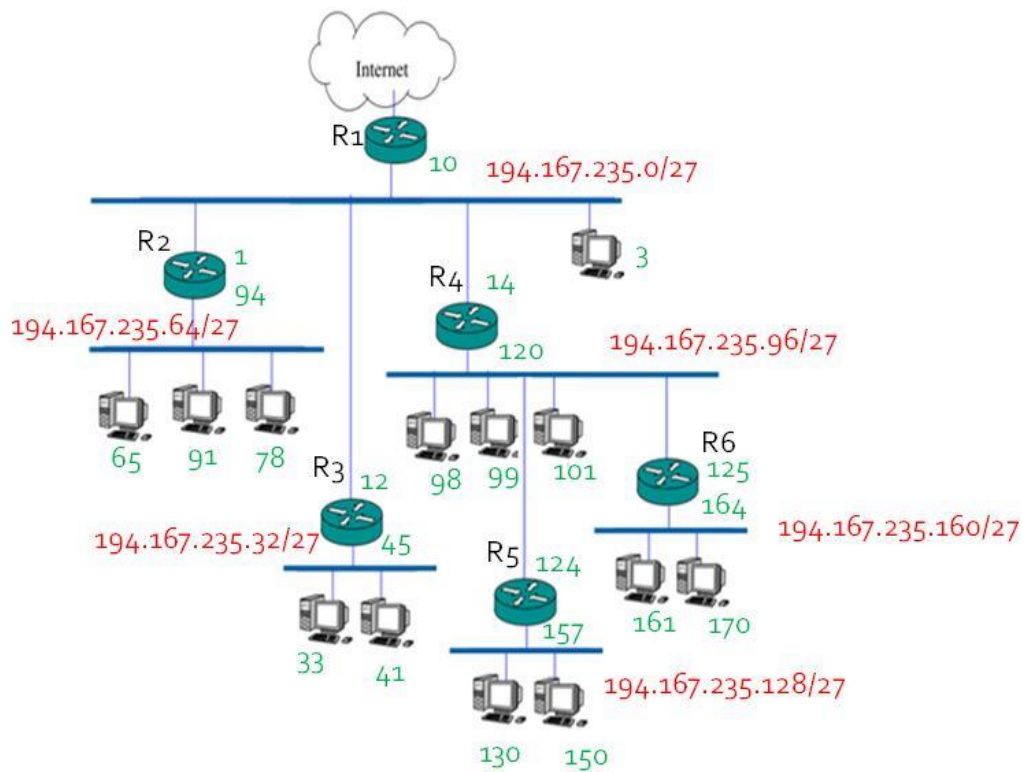


Figure 7. réseau constitué de 6 routeurs et de 13 postes. Dans le fichier de configuration donné, en plus des 6 routeurs, seuls les postes pc65, pc3, pc98, pc161 pc33 et pc130 seront lancés

ANNEXE

Pour analyser le fonctionnement des protocoles, vous allez utiliser le logiciel Wireshark (site www.wireshark.org). Wireshark (appelé autrefois Ethereal) est un logiciel gratuit et open-source, utilisable sur les systèmes Unix et Windows.

Wireshark est un analyseur de trames, ou encore analyseur de protocoles, ou sniffer. Il permet de lire toutes les trames passant sur l'interface réseau de votre machine dans le but d'analyser les échanges sur le réseau (surveillance du trafic pour la maintenance ou la sécurité). Wireshark analyse le trafic passant sur les interfaces Ethernet et WiFi. Wireshark repose sur la commande unix tcpdump, qui permet de sniffer le trafic en mode commandes ; le logiciel fournit juste une interface graphique à tcpdump pour faciliter la lecture. Il dispose en plus de nombreux outils permettant de simplifier et affiner l'analyse du trafic.

L'interface graphique présente trois fenêtres (Figure 10) :

- Fenêtre du haut : liste des paquets capturés avec résumé de leurs caractéristiques. En cliquant sur un paquet de cette fenêtre, vous modifiez le contenu des deux autres ;
- Fenêtre du milieu : contenu du paquet, en-tête par en-tête (des couches basses vers les couches hautes) ;
- Fenêtre du bas : représentation hexadécimale du paquet sélectionné. Les champs sélectionnés dans la fenêtre du milieu y sont affichés en caractères gras.

Pour observer le contenu de chaque champ d'un en-tête, il suffit de cliquer sur l'icône en triangle en face de l'en-tête (Figure 11).

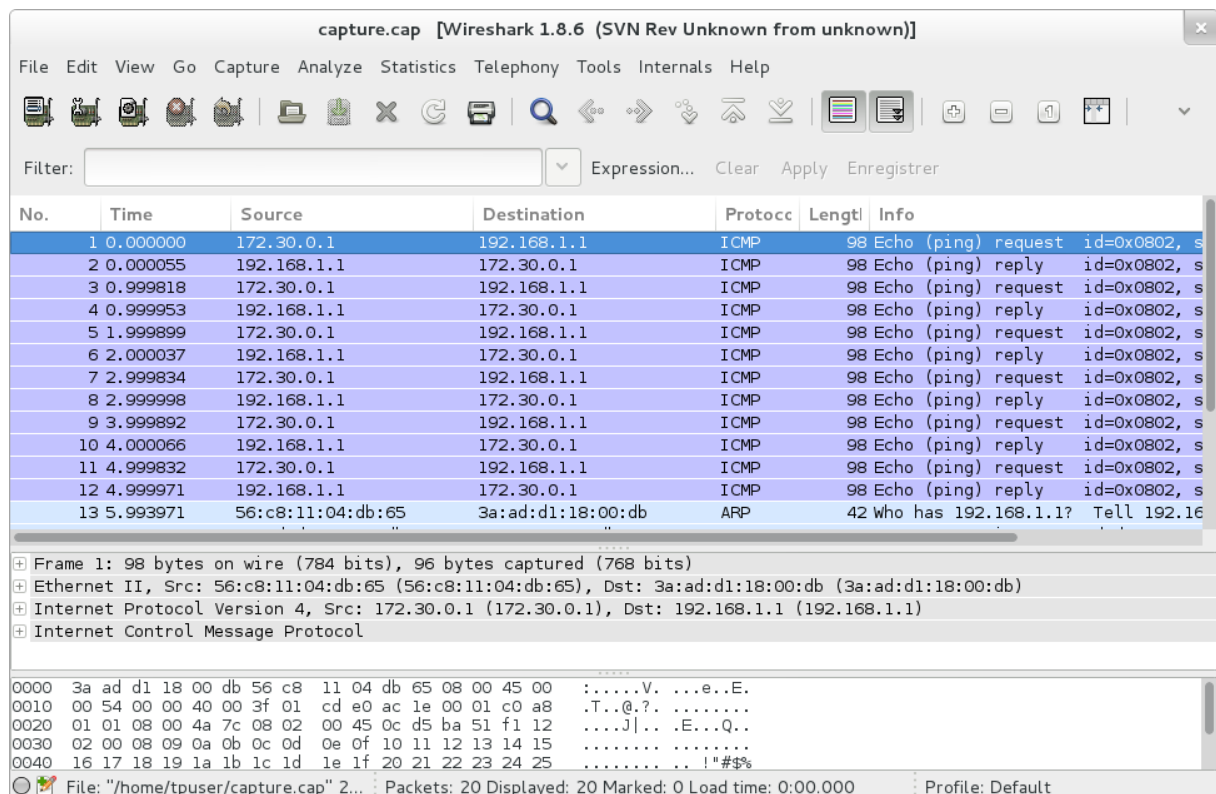


Figure 9. Capture par routeur

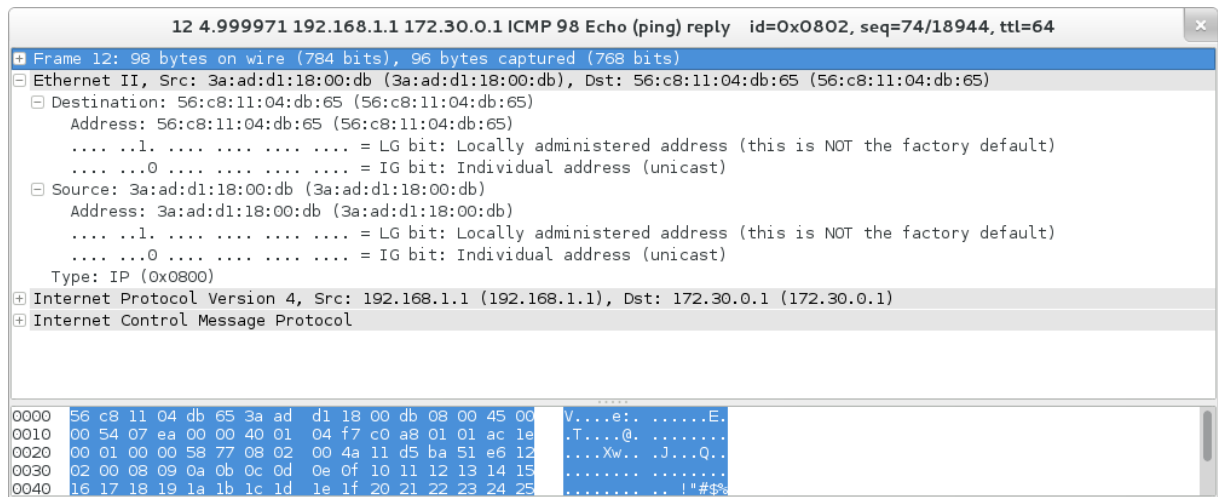


Figure 10. Contenu d'un