

Netkit est un logiciel libre sous licence GPL qui est composé de différents scripts permettant le lancement et l'arrêt de machines virtuelles et l'utilisation des outils réseau. Il est téléchargeable sur www.netkit.org.

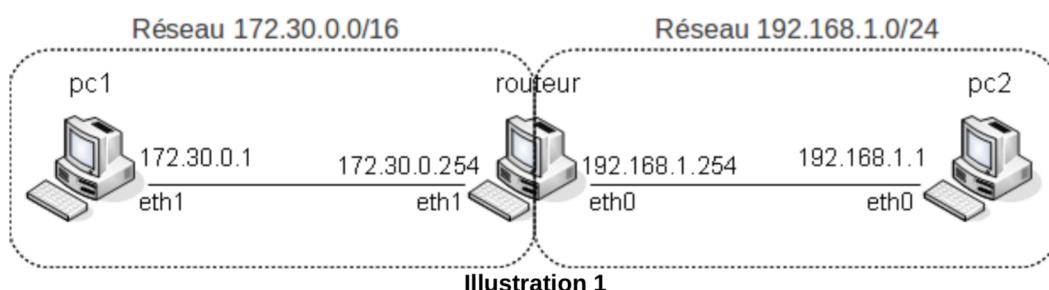
1. Installation du logiciel

Pour l'installer sur votre machine personnelle :

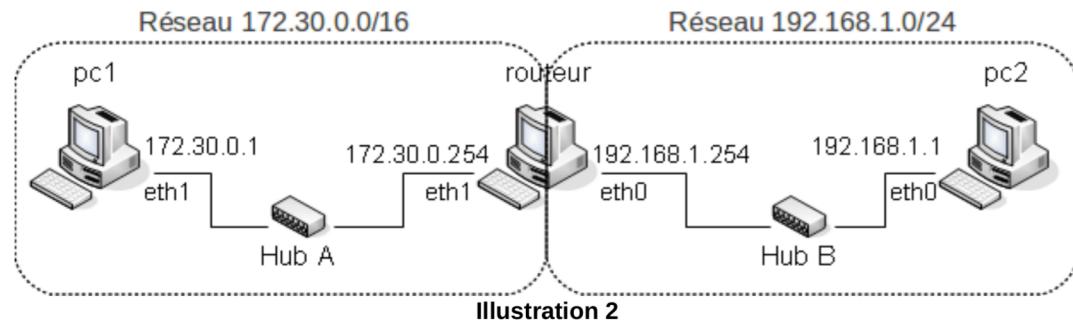
- Assurez-vous que le **paquetage uml-utilities** est installé sur votre machine. Si ce n'est pas le cas : `aptitude install uml-utilities`
- Créez un répertoire pour l'installation du logiciel, par exemple `/opt/netkit_install`
- Téléchargez depuis la page http://wiki.netkit.org/index.php/Download_Official les fichiers `netkit-x.ytar.bz2`, `netkit-filesystem-Fx.y.tar.bz2` et `netkit-kernel-Kx.y.tar.bz2` et placez-les dans le répertoire `/opt/netkit_install`
- Décomptez les fichiers en exécutant successivement les commandes :
`tar xjf netkit-x.ytar.bz2`
`tar xjf netkit-filesystem-Fx.y.tar.bz2`
`tar xjf netkit-kernel-Kx.y.tar.bz2`
- Rajoutez dans le fichier `~/.bashrc` les lignes suivantes qui définissent des variables d'environnement :
`export NETKIT_HOME=/opt/netkit_install/netkit`
`export PATH=$PATH:$NETKIT_HOME/bin`
`export MANPATH=$MANPATH:$NETKIT_HOME/man`
- Depuis un nouveau terminal (pour relancer le shell), placez-vous dans le répertoire `$NETKIT_HOME` et vérifiez la configuration du système hôte en exécutant la commande `./check_configuration.sh`

2. Création de machines virtuelles

Voici un exemple qui vous permettra de comprendre le fonctionnement de Netkit. Considérons le réseau suivant (Illustration 1). Vous allez découvrir les deux manières de le configurer.



Pour créer ce réseau sous Netkit, il faut déclarer deux hubs virtuels : le hub A, qui interconnecte pc1 et routeur, et le hub B qui interconnecte routeur et pc2. Le schéma est donc équivalent à celui de l'illustration 2 :



➤ 1ère méthode : création directe depuis un terminal

Pour créer une machine virtuelle directement depuis un terminal, la syntaxe est 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.

La syntaxe correspondant à la création du réseau précédent est :

```
vstart pc1 --eth1=A  
vstart pc2 --eth0=B  
vstart routeur --eth1=A --eth0=B --mem=512
```

Trois terminaux s'ouvrent : ils s'appellent respectivement pc1, 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 !). Le paramètre --mem indique la la machine routeur possède 512 Mo de mémoire (au lieu de 16 Mo par défaut).

Pour arrêter une machine, il faut exécuter dans le terminal où la commande vstart a été exécutée :

```
vhalt nom_machine
```

Il est aussi possible d'exécuter directement dans le terminal virtuel la commande linux : halt

Pour redémarrer une machine virtuelle, il suffit d'exécuter la commande linux : reboot

Attention ! Après cette commande, **toutes les configurations effectuées sur la machine (ifconfig, route) sont perdues. Les modifications de fichier, quant à elles, sont conservées.**

Remarque : si vous avez fermé accidentellement le terminal de la machine virtuelle, il est possible qu'elle soit toujours active même si l'interface graphique a disparu ! Vous devez alors tuer les processus correspondants (commande ps ax pour afficher les processus et relever leur numéro (PID), puis kill numéro_processus).

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

Il est aussi possible de décrire la structure du réseau dans un fichier nommé lab.conf. 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 hubs A, B et C.

Dans l'exemple précédent, le fichier lab.conf est donc :

```
pc1[1]=A  
pc2[0]=B  
routeur[1]=A  
routeur[0]=B  
routeur[mem]=512
```

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 : pc1, pc2 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 toutes les machines en une fois, il suffit de taper `lhalt`. On peut aussi les arrêter manuellement une à une par la commande `vhalt`.

Il est possible de préciser l'ordre de démarrage des machiens. Pour cela, il suffit d'ajouter dans le fichier lab.conf la ligne : `machines="routeur pc1 pc2"`

Dans cet exemple, la première machine démarrée est routeur, puis pc1 et enfin pc2.

Attention ! Hormis les répertoires au nom des machines, le fichier lab.conf et les fichiers .startup (voir plus loin), 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 pc1.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.

➤ Communication entre la machine hôte et la machine virtuelle

Chaque machine virtuelle possède un répertoire **/hosthome** qui est lié à votre répertoire personnel sur votre machine hôte. Ainsi, tous les fichiers placés dans votre répertoire personnel 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**.

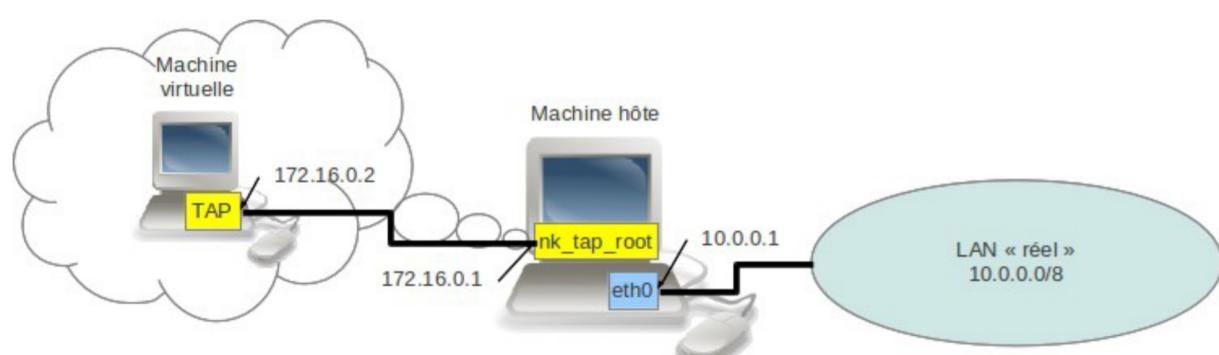
Votre répertoire personnel sur la machine hôte contiendra le fichier **capture.cap** qu'il suffit d'ouvrir dans Wireshark. Vous bénéficiez ainsi de l'interface graphique pour l'analyse de vos captures.

Si vous souhaitez exécuter d'autres commandes dans le terminal après avoir lancé votre capture, ajoutez un & à la fin de la commande : **tcpdump -w /hosthome/capture.cap &**

Pour arrêter la capture, il suffit d'exécuter la commande : **killall tcpdump**

3. Création d'une interface TAP

L'interface TAP permet à votre machine virtuelle de communiquer avec le réseau « réel » auquel est connectée votre machine hôte. Une interface virtuelle, nommée **nk_tap_root**, est aussi créée sur la machine hôte lors du démarrage de la machine virtuelle. Il faut définir une adresse pour chacune de ces deux interfaces. Les adresses doivent appartenir au même réseau et ne pas être utilisées sur le réseau « réel »... Un exemple est fourni sur l'Illustration 3.



Lors de la création de la machine virtuelle, il faut rajouter les paramètres en gras :

- si vous créez les machines directement dans un terminal :

```
vstart pc1 --eth0=A --eth1=tap,172.16.0.1,172.16.0.2
```

- si vous utilisez un fichier lab.conf :

```
pc1[eth1]=tap,172.16.0.1,172.16.0.2
```

Attention à la notation : on indique bien [eth1] et non [1] dans la ligne de commande du lab.conf

On définit ici une interface eth1 qui connecte la machine virtuelle à la machine réelle. La première adresse IP mentionnée est celle de l'interface côté machine hôte, la deuxième celle de l'interface côté machine virtuelle.

Il faut ensuite définir une ligne de routage par défaut dont la passerelle est l'interface nk_tap_root de la machine hôte : `route add default gw 172.16.0.1`

Il faut permettre à la machine hôte de servir de routeur. Elle doit en effet relayer les paquets sortant de la machine virtuelle et ceux qui lui sont destinés. Pour cela, le contenu du fichier /proc/sys/net/ipv4/ip_forward doit valoir 1. Vous pouvez exécuter simplement la commande :

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Enfin, si vous souhaitez utiliser les noms DNS des machines réelles, vous devez indiquer l'adresse IP de serveurs DNS dans le fichier /etc/resolv.conf. Il suffit par exemple de recopier les adresses des serveurs DNS enregistrés dans le fichier /etc/resolv.conf de la machine hôte.

Désormais vous pouvez, depuis votre machine virtuelle, communiquer avec votre machine hôte, le réseau local et Internet. En particulier, vous pouvez installer des paquetages en utilisant la commande aptitude install (ou apt-get install).