TP - Netkit

Netkit (http://wiki.netkit.org/index.php/Main_Page), un logiciel Open-Source, permet d'*émuler* les réseaux, avec une très grand partie de protocoles TCP/IP. Il est basé sur *uml* (user-mode linux), qui peut être exécuté comme un "user process" sur une *linux box* standard.

Pour une introduction détaillée sur Netkit, voir le document (disponible aussi sur la page web de ce cours) : http://wiki.netkit.org/netkit-labs/netkit_introduction/netkit-introduction.pdf

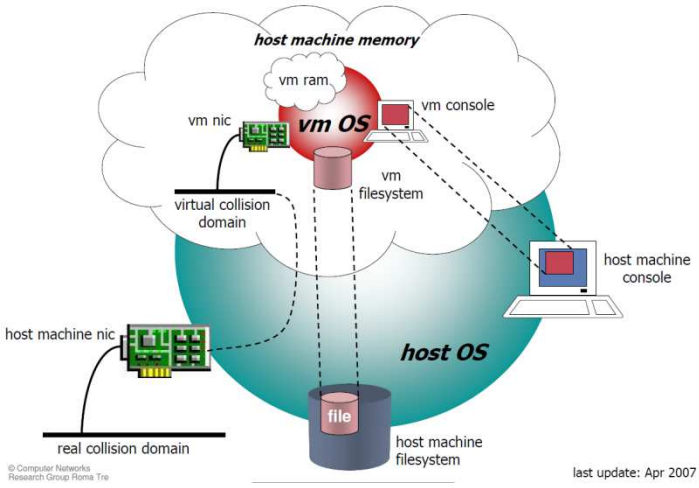
Netkit permet de créer des machines virtuelles, qui peuvent donc fonctionner comme des "feuilles" du réseaux (clients/servers), mais aussi comme bridge/switch ou routeurs. Plusieurs machines virtuelles peuvent être créées en même temps, pour émuler donc un réseaux avec nœuds/switch/routers ...

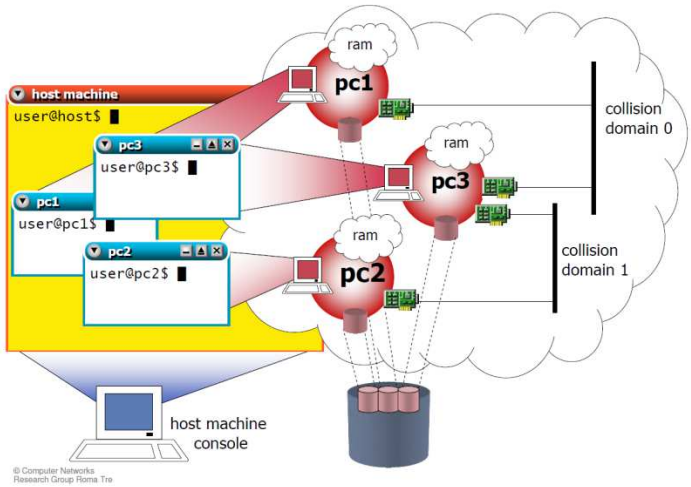
Chaque machine virtuelle possède :

- une console ("terminal window")
- de la mémoire
- un filesystem (qui est stocké dans un fichier dans le filesystem de la machine linux réelle)
- (une ou plusieurs) interfaces réseau

Chaque interface réseaux peut être connecté à un domaine de collision (virtuel).

Chaque domaine de collision peut être connecté à plusieurs interfaces réseau.





Commandes Netkit

Netkit met à disposition un ensemble d'outils et commandes pour créer facilement un réseau virtuel.

Ils existent deux types de commandes

- *v-prefixed commands* : pour créer/gérer une seule machine virtuelle
- *l-prefixed commands* : pour gérer un "laboratoire" (avec plusieurs machines virtuelles)

Voici la liste des commande *v-prefixed* :

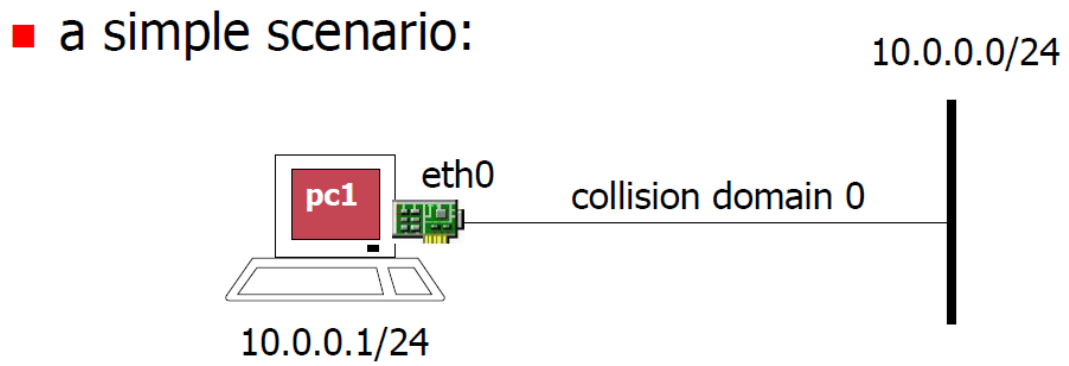
- **vstart**: starts a new virtual machine
- **vlist**: lists currently running virtual machines
- **vconfig**: attaches network interfaces to running vms
- **vhalt**: gracefully halts a virtual machine
- **vcrash**: causes a virtual machine to crash
- **vclean**: “panic command” to clean up all netkit

Et voici celle des commandes *l-prefixed* :

- **lstart**: starts a netkit lab
- **lhalt**: gracefully halts all vms of a lab
- **lcrash**: causes all the vms of a lab to crash
- **lclean**: removes temporary files from a lab directory
- **linfo**: provides information about a lab without starting it
- **ltest**: allows to run tests to check that the lab is working properly

1. Warm up Activity: Single Node

http://wiki.netkit.org/netkit-labs/netkit-labs_basic-topics/netkit-lab_single-host/netkit-lab_single-host.pdf



Sur votre machine Linux (réelle !) vous pouvez utiliser la commande "**vlist**" (liste des machines virtuelles actives) (contrôler le résultat de cette commande).

Ensuite on passe à la création de cette machine virtuelle avec la commande :

vstart pc1 --eth0=A

Cette commande va créer une machine virtuelle, dont le nom est pc1, qui a une interface réseaux eth0 connecté au domaine de collision appelé "A".

Un filesystem virtuel est automatiquement crée pour cette machine, et il est stocké sur le ficher (réel !) pc1.disk.

Utiliser à nouveau "**vlist**" 'machine réelle). Afficher la liste des fichiers .disk.

Vous êtes root sur la machine virtuelle pc1 (une nouvelle fenêtre s'est ouverte). Il faut donc configurer l'interface réseau avec l'adresse IP en figure, à l'aide de la commande ifconfig (avec "man ifconfig" vous pouvez voir la syntaxe de cette commande) :

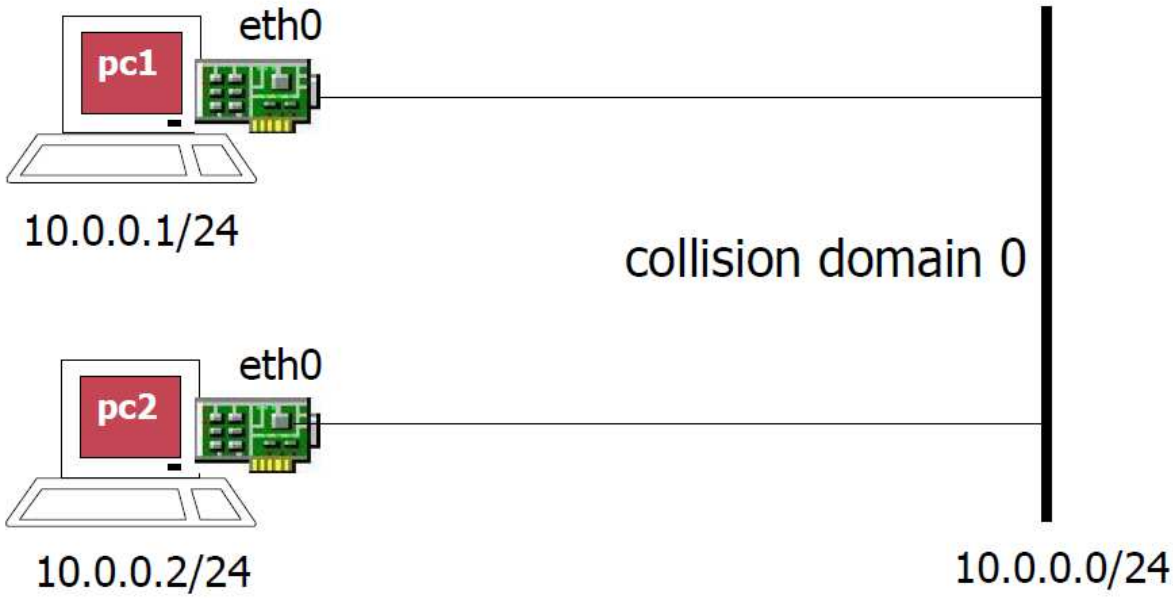
Q1.1 : ifconfig ...

Ensuite vérifiez la configuration avec la commande "**ifconfig eth0**" et aussi le tableau de routage : "**route**"

La commande "**vhalt pc1**" (machine réelle) ou "**halt**" dans pc1 (machine virtuelle) permet de fermer la machine virtuelle pc1.

2. Two hosts

http://wiki.netkit.org/netkit-labs/netkit-labs_basic-topics/netkit-lab_two-hosts/netkit-lab_two-hosts.pdf



Créer les 2 machines virtuelles illustrées dans la figure précédente (appeler le domaine de collision "A", comme dans l'exemple précédent)

Q2.1 : ...

...

Configurez les interfaces réseau de pc1 et pc2 :

Q2.2 : ...

...

Faire "ping" de pc1 vers pc2, et inversement de pc2 vers pc1 et tester si ça fonctionne :

Dans pc2 : **"ping 10.0.0.2"**

Utiliser tcpdump : sur la machine pc1 lancer un ping vers pc2 et le laisser tourner.

Entretemps, sur la machine pc2 lancer tcpdump : **tcpdump -i eth0**

Faire la même chose, mais cette fois avec la commande (toujours en pc2) suivante :

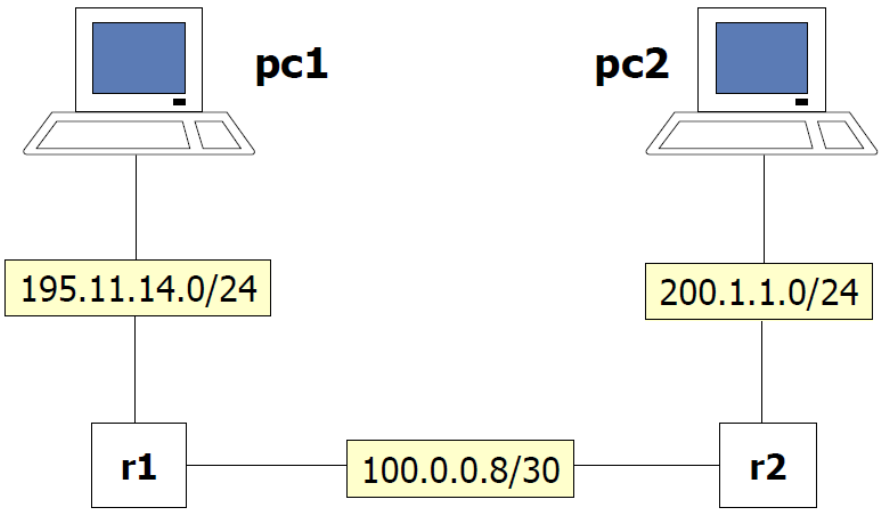
tcpdump -i eth0 -w /hosthome/capture.pcap

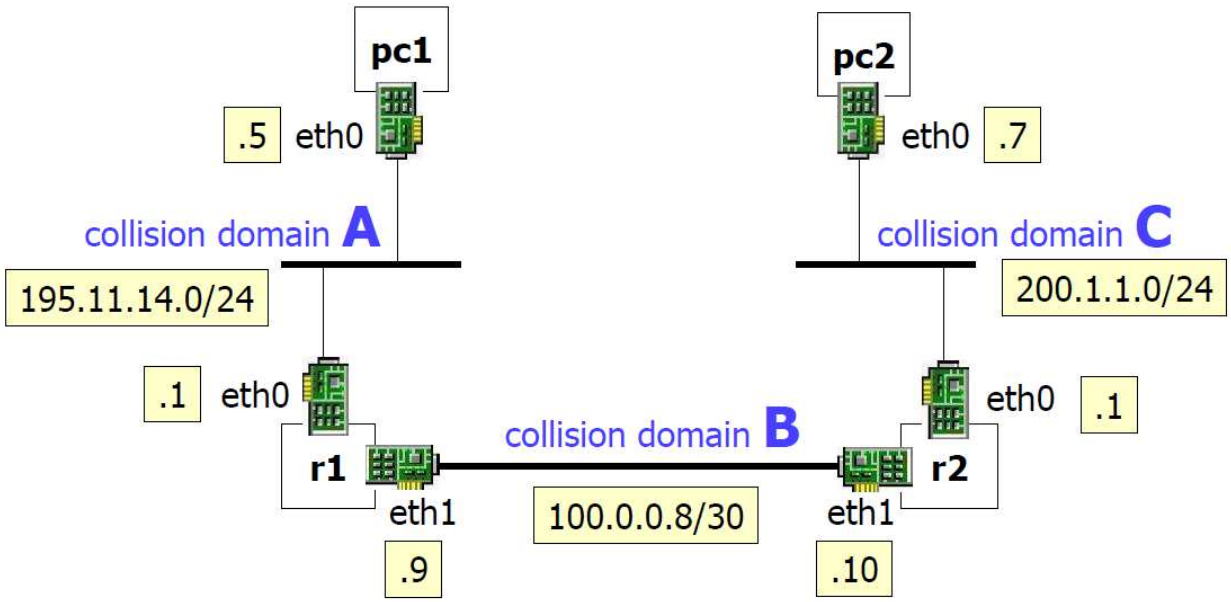
Ouvrir ensuite (sur la machine réelle) le ficher capture.pcap, avec Wireshark, et l'analyser.

3. Static Routing

http://www.netkit.org/netkit-labs/netkit-labs_basic-topics/netkit-lab_static-routing/netkit-lab_static-routing.pdf

Nous voulons créer le scénario suivant (la configuration détaillée des interfaces est dans la 2ème figure) :





Créer les 4 machines virtuelles (pc1, pc2, r1, r2), avec leurs interfaces et domaines de collision.

Q3.1 : **...**

...

...

...

Q3.2 : Configurer les interfaces réseau de ces 4 machines :

Dans pc1 : **...**

Dans pc2 : **...**

Dans r1 : **...**

...

Dans r2 : **...**

...

Q3.3 : Se mettre dans pc1 et faire "ping" vers l'interface eth0 de r1. Observer le comportement de la commande ping :

...

Q3.4 : Toujours en pc1, faire "ping" vers l'interface eth1 de r1, et observer le comportement de la commande ping :

...

Résultat : ...

Pourquoi on a ce résultat ? Regardez les tableaux de routage dans pc1 et dans r1 (commande "route").

Q3.5 : Il faut tout d'abord ajouter dans pc1 et pc2 le gateway (passerelle) par default, à l'aide de la commande "route" :

Configuration dans pc1 : **...**

Configuration dans pc2 : **...**

Q3.6 : Re-essayer, en pc1, à faire "ping" vers l'interface eth1 de r1, et observer le comportement de la commande :

...

Q3.7 : Toujours en pc1, faire "ping" vers l'interface eth1 de r2, et observer le comportement de la commande :

...

Q3.8 : Pourquoi l'interface de r2 est-elle "unreachable" ?

Est-ce que les messages "echo request" de "ping" arrivent en r2 ? Il suffit de vérifier en utilisant tcpdump sur l'interface eth1 de r2, tandis que la commande ping est toujours active en pc1 :

Dans r2 : **...**

...

L'adresse de pc1 est 195.11.14.5. Est-ce que r2 sait comment arriver vers pc1 ?

Regardez le tableau de routage de r2 pour répondre.

...

Q3.9 : Il faut donc "instruire" r2 sur comment rejoindre pc1. Nous allons configurer une route statique vers le réseau de pc1, à l'aide de la commande "route".

En r2 : **...**

Vérifier bien, après cette commande, le tableau de routage dans r2.

En même temps, configurer une route statique vers le réseau de pc2 en r1.

En r1 : **...**

Q3.10 : Maintenant, testez le fait que les deux pcs peuvent se connecter (commande "ping" de l'un à l'autre) :

pc1:# ...

Resultat du ping :

...

pc2:# ...

...

Q3.11 : nous avons vu que la route par default en pc1 peut être configurée avec la commande :

route add default gw 195.11.14.1 dev eth0

Pouvez-vous donner une commande qui a un résultat équivalent dans la forme suivante ? (à compléter) :

route add -net _____ netmask _____ gw _____ dev _____

Q3.12 : dans cette topologie de réseau, très simple, il est possible de configurer les routeurs r1 et r2 en utilisant exclusivement des routes par default. Proposez une solution en ce sens et testez-la.