INTRODUCTION A NETKIT

THE POOR MAN'S SYSTEM FOR EXPERIMENTING COMPUTER NETWORKING v3 (septembre 2013)

Ce document atteint la maturité avec cette version 3 qui intégre l'installation de NetKit et VisualNetKit sur plateformes Debian et Redhat.

Cette documentation est perfectible… Veuillez communiquer vos demandes de modifications à : Alain

alain.tournie@gmail.com

REFLEXIONS A PROPOS DES RESEAUX INFORMATIQUES

Les réseaux informatiques sont… complexes :

Différents matériels (ordinateurs, commutateurs, routeurs…) différentes interfaces, différents protocoles utilisés simultanément, topologies physiques variées et complexes.

EXPERIMENTATIONS SUR LES RESEAUX

Les expérimentations sur les réseaux réels sont souvent impossibles. les services offerts par les réseaux sont critiques pour l'entreprise et ne peuvent être perturbés. Les équipements réseaux sont coûteux et leurs achats ne peuvent être justifiés pour de simples tests. Ceci est également vrai pour les établissements de formation.

SIMULATION OU EMULATION ?

Un système de simulation s'attache à reproduire les performances d'un système réel : temps de latence, paquets perdus… Un système d'émulation tente de reproduire les fonctionnalités d'un système réel (configurations, architectures, protocoles) sans accorder d'importance aux performances.

NETKIT : UN SYSTEME D'EMULATION DE RESEAUX D'ORDINATEURS

Basé sur UML (User-Mode Linux) Chaque équipement réseau émulé est une machine virtuelle linux basée sur un UML KERNEL Chaque machine virtuelle Linux peut être configurée pour être un pont/commutateur ou un routeur. Plusieurs machines virtuelles peuvent être exécutées en même temps sur la même machine hôte. Chaque machine virtuelle possède : une console (terminal), une mémoire, un système de fichiers, une ou plusieurs interfaces réseaux. Chaque interface réseau peut être connectée à un domaine de collision virtuel. Chaque domaine de collision virtuel peut être connecté à plusieurs interfaces.

EMULER UN RESEAU INFORMATIQUE

chaque machine virtuelle sera créée dans une machine réelle. Plusieurs machines seront connectées à un domaine de collision et pourront communiquer avec chaque autre machine. Chaque machine virtuelle pourra être configurée comme station de travail, routeur, commutateur.

QU'EST-CE QUE NETKIT ?

un ensemble d'outils et de commandes qui peuvent être utilisés pour concevoir facilement un réseau informatique virtuel. Un système de fichiers standard qui peut être utilisé comme « modèle » pour créer le système de fichiers de chaque machine.

SYSTEME REQUIS SUR LA MACHINE HOTE

architecture i386 32 bits CPU 600 Mhz (ou supérieur) 10 Mo pour chaque machine virtuelle (dépend de la configuration de la MV). 600 Mo d'espace disque + 1-20 Mo pour chaque MV.

INSTALLATION DE NETKIT ET DE VISUALKIT

MODUS OPERANDI (validé sur Ubuntu 10.4)

INSTALLATION DE NETKIT

The poor man's system to experiment computer networking http://www.netkit.org/

ETAPE 1 : Téléchargez les 3 modules logiciels (latest release = x.y) : http://www.netkit.org/download.html

```
tar -xjSf netkit-x.y.tar.bz2
tar -xjSf netkit-filesystem-i386-Fx.y.tar.bz2
```

```
tar -xjSf netkit-kernel-i386-Kx.y.tar.bz2
```

Remplacer « x.y » par les numéros de version des fichiers.

ETAPE 2 : copier les 3 modules dans votre répertoire personnel (~)

ETAPE 3 : décompressez les trois modules :

```
$ tar -xjf netkit-x.y.tar.bz2
$ tar -xjf netkit-filesystem-Fx.y.tar.bz2
$ tar -xjf netkit-kernel-Kx.y.tar.bz2
```

ETAPE 4 : Adaptez et insérez les lignes suivantes dans le fichier /home/sio/.bashrc

```
# simulateur NETKIT
export NETKIT_HOME=/home/btssio/netkit
export MAN_PATH=:$NETKIT_HOME/man
export PATH=$NETKIT_HOME/bin:$PATH
export MANPATH=:/home/btssio/netkit/man
```

REMARQUE : Si vous installez NETKIT sur CENTOS 64 bits vous devez installer les paquets suivants :

```
# yum install compat-libstdc++-296.i686 compat-libstdc++-33.i686
```

Vous devez aussi installer :

```
# yum install xterm
```

ETAPE 5 : Relancez la session de l'utilisateur-installateur (sio)

ETAPE 6 : Vérifiez la configuration de NETKIT :

```
$  cd ~/netkit
$ ./check_configuration.sh
```

Si tout est OK, vous devez obtenir le message suivant :

```
>  Checking for availability of terminal emulator applications:
    xterm          : found
    konsole        : not found
    gnome-terminal : found
passed.

[ READY ] Congratulations! Your Netkit setup is now complete!
          Enjoy Netkit!
```

ETAPE 7 : Pour finir, vous pouvez tester NETKIT en gérant votre première machine virtuelle :

```
$ vstart ma_machine (lancement de la machine virtuelle « ma_machine »)
$ vlist (liste des machines virtuelles lancées)
$ vhalt -r ma_machine (arrêt de la machine « ma_machine »
```

INSTALLATION DE VISUALKIT

Pour effectuer toutes les étapes suivantes vous devez avoir les privilèges de l'utilisateur « root »

ETAPE 1 : Télécharger l'archive du logiciel VISUALKIT sur le site :

visualnetkit-1.4.tar.bz

ETAPE 2 : Déplacez l'archive téléchargée dans /home/sio

ETAPE 3 : Décompressez l'archive :

```
# tar -xvf visualnetkit-x.y.tar.gz
```

où « x.y » est le numéro de version. (version 1-4 en septembre 2013)

PRE-REQUIS COMPILATION SUR DEBIAN OU DERIVEE (validé sur Ubuntu 10.4)

Installez si nécessaire les paquets suivants :

```
# apt-get install   libqt4-dev g++
```

PRE-REQUIS COMPILATION SUR REDHAT OU DERIVEE (validé sur CENTOS 6-4)

installer si nécessaire les paquets suivants :

```
# yum install qt-devel gcc-c++
```

Modifier le contenu de la variable QMAKE dans le fichier .../visualnetkit-x.y/build.sh

```
QMAKE="/usr/lib/qt4/bin/qmake"
```

(CENTOS 6-4 32 bits)

ou

```
QMAKE="/usr/lib64/qt4/bin/qmake"
```

(CENTOS 6-4 64 bits)

ETAPE 4 :

```
#  cd /home/sio/visualnetkit
```

ETAPE 5 : Compilez en lançant le script :

```
#  ./build.sh
```

ETAPE 6 : Lancez VisualNetkit :

</code># /home/sio/visualnetkit/bin/visualnetkit.sh</code>

Vous pouvez bien entendu créer un « lanceur » sur le Bureau.

NETKIT PRINCIPALES COMMANDES

NETKIT embarque *2 jeux de commandes :*

commandes du type « **vcommands** » commandes du type « **lcommands** »

Les **vcommands** permettent de démarrer, d'arrêter et de configurer une machine virtuelle isolée.

Les **lcommands** permettent de monter un « laboratoire » constitué de plusieurs machines virtuelles en interaction dans un réseau informatique.

Le manuel en ligne d'une commande est obtenu en entrant le nom de la commande suivi de l'option « -h »

LES « V » COMMANDES

vstart : démarrer une nouvelle machine virtuelle. Les options de la commande permettent de préciser les caractéristiques de la machine (mémoire, cartes réseaux...)

syntaxe d'utilisation :

vstart [options] nom*de*la_machine

Exemple (création d'une machine nommée routeur possédant 2 cartes Ethernet):

vstart –eth0=lan1 –eth1=lan2 routeur

options les plus utilisées :

–ethN=nom*du*HUB connecte -M taille*de*la_memoire (en Mo)

vlist : liste des machines virtuelles démarrées, vconfig : permet d'attacher une carte réseau à une mv démarrée (très pratique) :

Exemple : vconfig eth2=SW2 ma_machine

- vhalt : permet d'arrêter « proprement » une machine virtuelle (on peut aussi utiliser la commande halt dans la machine virtuelle)
- vcrash : commande similaire à vhalt mais l'arrêt de la mv n'est pas « propre ». Cela permet de simuler un arrêt brutal de la mv tel que l'on pourrait en provoquer un sur une machine réelle par arrachage du cordon d'alimentation.
- vclean : commande permettant de faire « un grand nettoyage » : suppression des processus associés aux hubs virtuels, suppression des mv en fonctionnement, suppression des liaisons de type « tap ».

LES « L » COMMANDES

Un laboratoire (lab) est un ensemble de machines virtuelles en interconnexion constituant un réseau informatique pouvant être complexe.

```
                                                                                `
```

Un lab est en fait constitué d'un ensemble de répertoires (un par machine virtuelle à lancer), d'un fichier de configuration général lab.conf, et d'un fichier de configuration par machine (XX.startup).

Pour lancer un lab, il faut être positionné´ dans le répertoire contenant le fichier lab.conf.
Les commandes importantes permettant de gérer un lab sont :

```
* lstart : démarrer un lab netkit : exemple = lstart -d reseau -f
* lhalt : arrêt « propre » d'un laboratoire : exemple = lhalt -q (arrêt simultané de toutes les mv),
* lcrash : simulation d'un crash de toutes les mv du labo,
* linfo : permet d'obtenir des informations sur le labo sans démarrage de celui-çi,.
* ltest : permet de tester la configuration du labo et de vérifier que celui-çi a été monté proprement.
* lclean : supprime tous les fichiers temporaires d'un lab.
```

Vous pouvez monter un laboratoire en mode commande (voir documentation officielle) mais il est beaucoup plus « sympa » d'utiliser l'interface graphique VisualNetkit !!!

UTILISATION DU DVD LIVE « Netkit4TIC.iso V. 3.0)

1 – Démarrer le PC en « bootant » depuis le DVD.

2 – Au prompt, entrez la commande suivante afin de configurer le clavier en français:

knoppix lang=fr (tapez : lqng au lieu de lang)

3 – Après lancement du système, lancez une fenêtre « terminal »

4 – Passez en mode « superutilisateur » en entrant la commande :

```
# sudo -s
```

LANCEMENT ET ARRET D'UNE OU DE PLUSIEURS MACHINE(S) VIRTUELLE(S) AUTONOME(S)

1 – Vous pouvez lancer maintenant une ou plusieurs machine(s) virtuelle(s) Linux :

```
# vstart machine1
```

et… commencer à travailler !

2 – Vous pouvez, quand vous le désirez, stopper proprement votre machine virtuelle :

```
# vhalt machine1
```

SAUVEGARDE ET RESTAURATION D'UNE OU DE PLUSIEURS MACHINE(S) VIRTUELLE(S) AUTONOME(S)

1 – Vous pouvez enregistrer votre machine virtuelle sur une clé USB montée par une commande du type :

```
# cp machine1.disk   /media/sdb1
```

2– Après redémarrage de Netkit4TIC, vous pouvez recharger la ou les machine(s) sauvegardée(s) sur clé USB montée par une commande du type :

```
# cp /media/sdb1/machine1.disk    ~
```

Attention : typiquement, la taille d'un fichier « .disk » est de 1Go !!! Cette taille de fichier peut apparaître comme étant un problème. Voir point 4.4!!

LANCEMENT D'UN LABORATOIRE (PLUSIEURS MACHINES EN RESEAU)

Bien qu'il soit tout à fait possible de configurer un réseau de machines virtuelles en mode commande, il est beaucoup plus confortable d'utiliser l'interface graphique visualnetkit que vous lancerez par la commande :

```
#  visualnetkit.sh
```

SAUVEGARDE/RESTAURATION D'UN LABORATOIRE

Le laboratoire peut bien entendu être sauvegardé sur clé USB montée (ou autre support). Sauf nécessité et pour économiser de la place sur le support de sauvegarde, seuls les fichiers de configuration seront sauvegardés. Ces fichiers sont contenus dans le répertoire ayant le même nom que le laboratoire créé.

Au besoin, il est possible de détruire les systèmes de fichiers des différentes machines virtuelles (fichiers « .disk ») en exécutant la commande lclean après avoir arrêté le labo.

La sauvegarde/restauration des fichiers de configuration d'un laboratoire contenu dans un répertoire labo se fera par les commandes typiques suivantes :

```
# cp  -r  labo/   /media/sdb1
# cp  -r  /media/sdb1/labo/    repertoire_cible
```

ECHANGE DE FICHIERS ENTRE UNE MACHINE VIRTUELLE (GUEST) ET LA MACHINE HOTE (HOST)

Sur NETKIT, le répertoire /hosthome de chaque machine virtuelle est mappée sur le répertoire /home/user de la machine hôte (machine physique sauf installation virtuelle de NETKIT)

CAPTURE DE TRAMES

Les machines virtuelles lancées par NETKIT ne supportent pas une interface graphique. La capture de trames ne peut se faire par WIRESHARK. Nous utiliserons donc tcpdump.

La commande suivante permet de lancer une capture de trame :

```
#  tcpdump -i eth0 -U -w /hosthome/monfichier.cap
```

Sur la machine hôte et après la fin de la capture, vous pouvez lancer WIRESHARK pour lire le fichier /home/user/monfichier.cap

CAPTURE DE TRAMES AVEC uml_dump

uml_dump est un outil écrit par Julien Iguchi-Cartigny, Professeur associé au laboratoire XLIM (Université de Limoges). Ce logiciel permet de capturer et d'analyser en temps réel toutes les trames qui transitent sur un hub virtuel netkit :

http://kartoch.blogspot.fr/2010/06/new-package-of-umldump-for-netkit-27.html

Après installation, l'usage se résume à 2 commandes :

```
$ vdump switch_name
$ vdump A | wireshark -i - -k
```

SE CONNECTER A L'INTERNET ET/OU AU RESEAU DE LA MACHINE REELLE

Il faut rajouter une ligne de configuration dans le fichier « lab.conf »

```
internet[0]=tap,192.168.1.5,192.168.1.200
```

ou bien lancer la machine avec une interface tap :

```
vstart –eth0=tap,192.168.1.5,192.168.1.200  –eth1=...   internet
```

internet[0] est l'interface de la machine nommée « internet » accédant à l'internet (mais aussi au réseau local de la machine réelle...), 192.168.1.5 est l'adresse IP de l'interface réseau de la machine réelle (hôte), 192.168.1.200 est l'adresse IP de l'interface eth0 de la machine virtuelle nommée ici internet.

Attention : Ces 2 adresses doivent appartenir au même sous-réseau. Emphase (italique)

Remarque : si vous voulez récupérer des paquets de logiciels depuis l'Internet, il est fortement conseillé d'augmenter la mémoire virtuelle de la machine virtuelle en rajoutant la ligne suivante dans le fichier « lab.conf » :

```
internet[mem]=valeur
```

valeur est la taille de la mémoire exprimée en Mo.

Il convient ensuite de renseigner sur la machine virtuelle démarrée un ou plusieurs serveurs DNS dans le fichier /etc/resolv.conf

On peut par exemple utiliser des serveurs OpenDNS (208.67.222.222 et/ou 208.67.220.220)

Après démarrage du laboratoire, NETKIT effectue les opérations suivantes :

- ajout d'une règle de translation d'adresse,
- ajour d'une règle autorisant les flux émis par notre interface TAP,
- activation du routage,
- ajout d'une règle de routage sur la machine virtuelle indiquant que la passerelle est l'interface TAP du système hôte.

Si la machine internet ainsi configurée est reliée aux autres machines virtuelles du laboratoire via une deuxième interface eth1 et si l'on veut que toutes les machines du laboratoire accèdent à l'internet, il faut entrer la commande suivante sur la machine internet :

```
#  iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

Si le système de la machine hôte est CENTOS, il faut « libérer » le pare-feu :

```
[root@localhost btssio]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@localhost btssio]# iptables -D FORWARD 2
[root@localhost btssio]#
```

L'interface TAP de la machine réelle n'est pas désactivée à l'arrêt des machines virtuelles. Si vous voulez détruire cette interface, passez la commande :

```
# vclean -T
```

SE CONNECTER A UNE MACHINE VIRTUELLE QUI N'APPARTIENT PAS AU LABO NETKIT

Il peut être utile de connecter une machine virtuelle netkit à une machine virtuelle créée par un virtualisateur type Virtualbox ou Vmware... Cela permet par exemple de communiquer avec une machine virtuelle avec interface graphique ou bien encore de simuler un client Windows virtuel.

La technique consiste là-aussi à créer un pont via une interface tap. Il faut rajouter une ligne de configuration dans le fichier « lab.conf »

```
machine_netkit[0]=tap,10.0.2.15,10.0.2.200
```

ou bien lancer la machine avec une interface tap :

```
vstart –eth0=tap,192.168.1.5,192.168.1.200  –eth1=... machine_netkit
```

machine*netkit[0] est l'interface eth0 de la machine nommée « machine*netkit » accédant à la machine virtuelle Virtualbox, Vmware... ou autre.

10.0.2.15 est l'adresse IP de l'interface réseau de la machine virtuelle Virtualbox, Vmware... 10.0.2..200 est l'adresse IP de l'interface eth0 de la machine virtuelle netkit. Attention : Ces 2 adresses doivent appartenir au même sous-réseau.

REMPLACER LES HUBS VIRTUELS PAR DES SWITCHS

NETKIT est configuré par défaut pour utiliser des hubs. Ceux-ci, bien qu'obsolètes, ont l'avantage de permettre à tous les équipements connectés d'écouter (et donc éventuellement de capturer) l'intégralité du trafic du LAN. Si l'on veut substituer les hubs par des switchs (par exemple pour simuler des VLAN), il faut modifier le fichier :

```
bin/script_utils
```

et plus précisément remplacer les 2 lignes :

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

par les lignes (l'option « -hub » doit être omise) :

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

CONSTRUIRE UN LABO EN MODE COMMANDE : LA COMMANDE VSTART EN DETAIL

Chaque machine du laboratoire sera lancée par la commande vstart. Il est possible de consulter le manuel en ligne de la commande (man vstart). Bien entendu, il est possible d'écrire un script shell pour lancer un ensemble de machines virtuelles (donc un laboratoire).

Syntaxe de la commande vstart :

```
vstart [options] MACHINE-NAME
```

Citons ici quelques options intéressantes (la liste exhaustive des options est dans le « man ») :

-M MEMORY ou –mem=MEMORY : permet de spécifier la quantité de mémoire embarquée exprimée en Mo.

–ethN=DOMAIN : permet de définir une carte Ethernet de numéro N attachée à un domaine de collision (hub) ou de commutation (switch) appelé DOMAIN.

–ethN=tap,TAP*ADDRESS,GUEST*ADDRESS : permet de créer un pont (bridge) vers la machine réelle ou vers une machine virtuelle non créée par netkit. TAP*ADDRESS est l'adresse IP de la machine cible, GUEST*ADRESS est l'adresse IP de la machine invitée netkit (voir points 7 et 8).

–con0=MODE et/ou –con1=MODE : chaque machine virtuelle peut posséder au plus 2 consoles (terminaux de type tty).

Exemples de MODE :

this : attache la console aux terminaux standards stdin/stdout, **port:tcp_port :** attache la console à un port TCP. Très pratique pour accéder à la machine virtuelle à distance via telnet ou ssh par exemple.

D'autres options sont disponibles : consultez le man !!!

Exemple d'utilisation de la commande vstart :

```
 # vstart -M 40 --eth0=LAN --eth1=tap,192.168.1.3,192.168.1.11 --con0=this --con1=port:22 internet
```

Cette commande permet de lancer une machine virtuelle « internet » possédant 40 Mo de mémoire et 2 cartes Ethernet : eth0 attachée au réseau LAN, eth1 d'adresse 192.168.1.11 « bridgée » avec la carte de la machine hôte 192.168.1.3. La première console (con0) de la machine virtuelle sera la console de la machine réelle (mode this) et la machine virtuelle « internet » sera accessible à distance par ssh (port 22) via la console con1 depuis n'importe quelle autre machine virtuelle attachée à LAN.

ANATOMIE D'UN LABORATOIRE NETKIT

Arborescence typique d'un laboratoire à la mode netkit :

```
lecter@hannibal-desktop:~/ana-labo$ tree
 `-- ana2
    |-- R1
    |-- R1.disk
    |-- R1.log
    |-- R1.startup
    |-- R1.startup~
    |-- ST1
    |-- ST1.startup
    |-- ST1.startup~
    |-- ST2
    |-- ST2.disk
    |-- ST2.log
    |-- ST2.startup
    |-- ST2.startup~
    |-- ST3
    |-- ST3.disk
    |-- ST3.log
    |-- ST3.startup
    |-- ST3.startup~
    |-- lab.conf
    |-- lab.conf~
    |-- lab.dep
    |-- lab.xml
    `-- lab.xml~

5 directories, 19 files
lecter@hannibal-desktop:~/ana-labo$
```

LE FICHIER « lab.conf »

Ce fichier décrit la topologie du réseau :

```
###
# This file is createb by Visual Netkit 1.1 version
# http://www.netkit.org ~ http://code.google.com/p/visual-netkit/
#
LAB_DESCRIPTION="petit essai de Netkit !"
LAB_VERSION="1.0"
LAB_AUTHOR="Alain"
LAB_EMAIL="alain.tournie@gmail.com"
LAB_WEB=""
R1[0]="LAN1"
R1[1]="LAN2"

# La ligne suivante realise un pont entre la carte eth3 de R1 et la carte Ethernet de la machine hote
R1[3]="tap,192.168.1.2,192.168.1.200"

ST1[0]="LAN1"

# La ligne suivante indique la taille de la mémoire embarquee sur la MV ST1
ST1[mem]=512

ST2[0]="LAN1"
ST3[0]="LAN2"
```

LES SOUS-REPERTOIRES ASSOCIES AUX MACHINES VIRTUELLES

A chaque machine virtuelle déclarée dans le fichier « lab.conf » est associé un sous-répertoire.

Ainsi à la machine virtuelle R1 est associé le sous répertoire R1. Ce sous-répertoire est « mappé » avec le répertoire « /root » de la machine virtuelle R1 ce qui permet un échange de fichiers facile.

LE FICHIER OPTIONNEL « lab.dep »

Le fichier « lab.dep » décrit les relations de dépendance qui existent entre les ordres de démarrage des machines virtuelles.

```
lecter@hannibal-desktop:~/ana-labo/ana2$ cat lab.dep

R1: ST1 ST3
```

Ici R1 ne sera démarrée qu'après le lancement de ST1 et ST3

LES FICHIERS « .disk »

Le fichier R1.disk (par exemple) contient l'arborescence de la machine R1. Attention : typiquement , la taille de ce type de fichier est de 1Go environ.

La commande «lclean [options] [MACHINE-NAME...] » permet de détruire les fichiers « .ready », « .log », « .disk »

LES FICHIERS « STARTUP » et « SHUTDOWN »

Ces fichiers contiennent des scripts exécutés à l'intérieur des machines virtuelles au moment de leur démarrage (startup) ou de leur extinction (shutdown).

shared.startup et shared.shutdown vont affecter toutes les machines virtuelles. vm*name.startup et vm*name.shutdown ne concernent que la machine nommée vm_name.

Exemple du fichier R1.startup

```
###
# This file is createb by Visual Netkit 1.1 version
# http://www.netkit.org ~ http://code.google.com/p/visual-netkit/
#
/sbin/ifconfig eth0 192.168.1.1 netmask 255.255.255.0 broadcast  ## 'LAN1' collision domain ##
/sbin/ifconfig eth1  netmask  broadcast  ## 'LAN2' collision domain ##
/sbin/ifconfig eth0 192.168.1.1 netmask 255.255.255.0 up ## 'LAN1' collision domain ##
/sbin/ifconfig eth1 up ## 'LAN2' collision domain ##
/sbin/ifconfig eth3 up ## 'tap,192.168.1.2,192.168.1.200' collision domain ##
```