

VIRTUALISATION DE RESEAU AVEC KVM

Serge Bordères

Centre d'Etudes Nucléaires de Bordeaux-Gradignan

9 avril 2009

Contexte

■ Transport : 27 Commutateurs HP Procurve

- Spanning-tree
- VLAN : 802.1Q
- 25 VLAN

■ Intelligence du réseau : sous Linux

- Routage et filtrage (Netfilter)
- Fonctions DNS/DHCP
- Serveurs Radius (Freeradius) .Filaire et Wifi. (authentification mac-based et 802.1X)
- Serveurs VPN (OpenVpn)

Contexte

■ Tous les systèmes Linux participants au réseau sont bootés sur CDRROM

- Générés par RAMUX/CDNUX
- Fedora 9
- Pas de disque
- Mémoire flash (clé USB)

■ Dispositif de continuité des services réseau

- Deux routeurs Linux redondants. Utilisation de **VRRP** (actif/passif)
- Trois serveurs DNS/DHCP/RADIUS (machines physiques)
- Un (puis deux) serveur(s) Openvpn (actif/actif)
- Capacité de booter les CD sur « n'importe qu'elle » machine

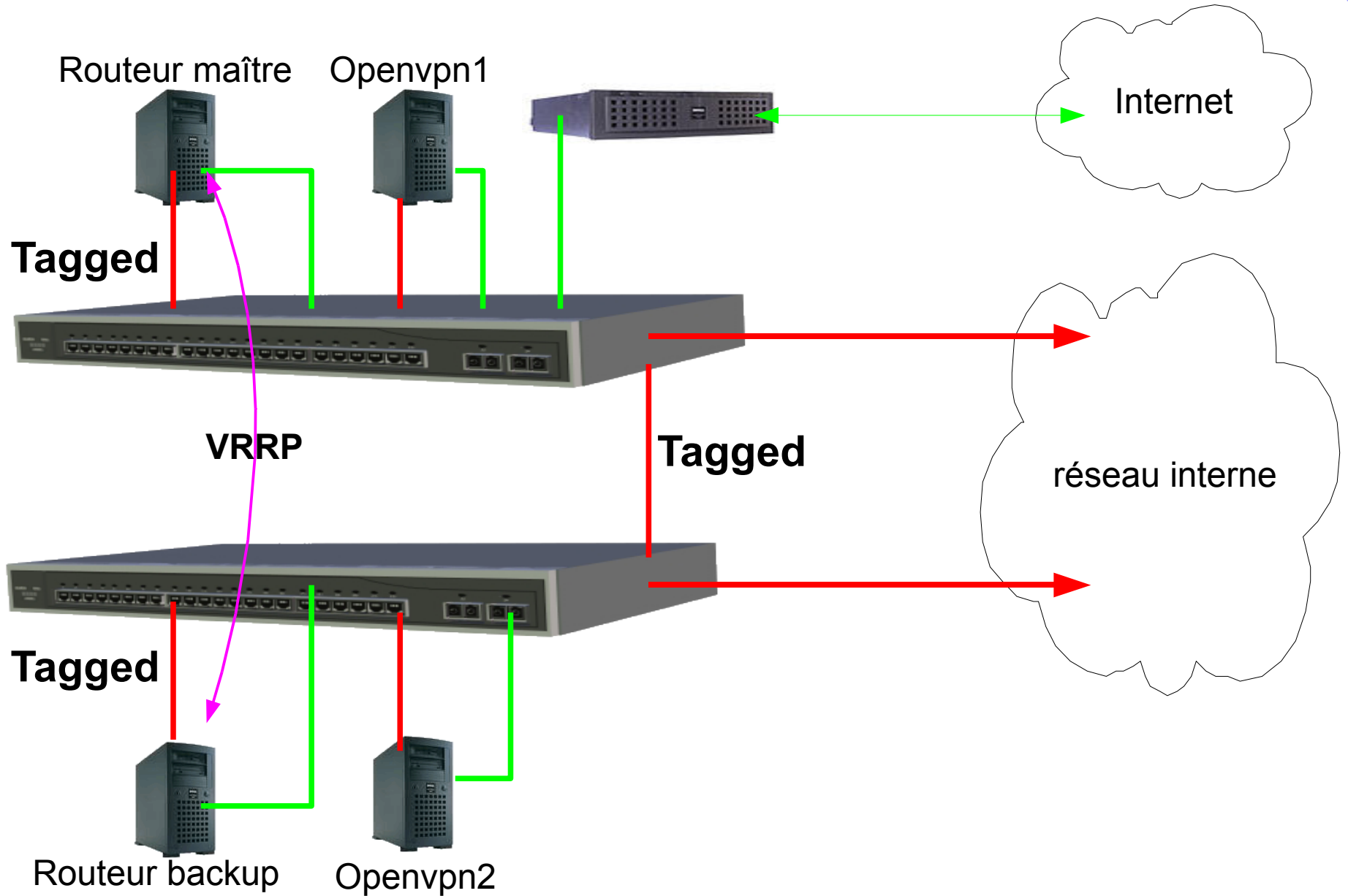
■ Architecture réseau

- Liaisons redondantes entre les commutateurs (spanning tree)
- Rôle redondant des commutateurs

■ Configurations réseau et filtrage

- Générées par **LabWall** (Netfilter)

Contexte



Questions de départ

- Est-il possible de mettre en place une plateforme virtuelle de tests des configurations réseau, des filtrages et des relations entre les différents éléments du réseau (routeurs, serveurs...) ?
 - ✦ Il est très difficile de faire des tests sur la configuration réelle parce que le fonctionnement du réseau est hyper-sensible pour l'ensemble des applications.
 - ✦ Pour se familiariser avec les protocoles mis en oeuvre (provoquer des situations, analyser les comportements, définir des stratégies...)
- Est-il envisageable d'utiliser la plateforme virtuelle comme solution de secours ?
- Est-il envisageable que le réseau en exploitation soit toujours virtuel ?
- Comment techniquement ?

Différents usages de la virtualisation

■ Simuler le réseau réel

- Reproduire le comportement du réseau réel

■ Mettre à jour/modifier le réseau

- Tests des updates système
- Tests de nouvelles configurations/fonctionnalités
- Test des procédures

■ Exploiter

- Passer le réseau virtuel au niveau réel après update
- Utiliser le réseau virtuel en remplacement total ou partiel du réseau réel

De quoi a-t'on besoin ?

- D'un système de virtualisation
- D'un commutateur virtuel supportant 802.1Q
- Le système de virtualisation doit pouvoir booter sur les images ISO des CD
 - Transposition « tels que » des systèmes physiques au virtuel
 - On boot la même chose au niveau physique ou au niveau virtuel
- De faire communiquer le commutateur virtuel avec le réseau réel
- D'assurer le transport des VLAN (tag) entre le monde virtuel et le monde réel

|

KVM: pourquoi ?

- Souhait d'une solution de virtualisation aussi « intime » que possible avec l'environnement Linux => KVM est inclus dans le noyau Linux (depuis 2.6.20)
- Souhait d'une solution de virtualisation rapidement mise en oeuvre
L'objectif c'est d'abord de travailler sur l'application et pas de passer trop de temps sur la mise en oeuvre du système de virtualisation. Avec KVM ½ journée a suffit pour apprendre et mettre en place l'environnement initial.
- Nécessité de boot sur image ISO : KVM sait faire naturellement.
- Existence d'un commutateur virtuel (**VDE**), supportant 802.1Q
- Possibilité de créer plusieurs commutateurs virtuels reliés entre-eux (éventuellement sur des machine physiques différentes)
- Curiosité pour KVM

KVM

- Hyperviseur de virtualisation inclus dans le noyau Linux (depuis 2.6.20)
- Dérive du projet QEMU
- Solution de full virtualisation
- Possibilité de paravirtualisation possible (notamment sur l'interface réseau) (driver VIRTIO dans le kernel)
- Utilise les extensions VT des processeurs Intel ou AMD-V pour AMD

VDE : Virtual Distributed Ethernet

VDE est un commutateur virtuel lié à KVM
Il tourne dans un processus classique dans la machine hôte

- Support des VLAN 802.1Q
- Support du spanning tree (pas testé)
- Gérable par ligne de commande et fichiers de configuration
- Commandes permettant de :
 - + Créer des ports
 - + Créer des VLAN
 - + Associer des VLAN aux ports
 - + Déclarer des port Tagged
 - + Etc
- Possibilité de créer plusieurs commutateurs virtuels, éventuellement sur des machines hôtes distinctes (notion de câble croisés)
- Existence d'un outil de test (**wirefilter**) capable de simuler des problèmes de types physiques tels que des paquets perdus, corrompus etc. (pas testé)

Création du monde...virtuel

- Connecter une machine virtuelle sur un VLAN
- La faire communiquer avec le réseau réel dans son VLAN et avec les autres VLAN

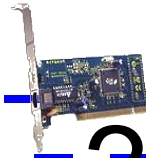
Création du monde...virtuel



MV1



Comment relier le commutateur virtuel et le commutateur réel ?



Machine hôte

Réseau physique



Routeur réel
(RROUTE)



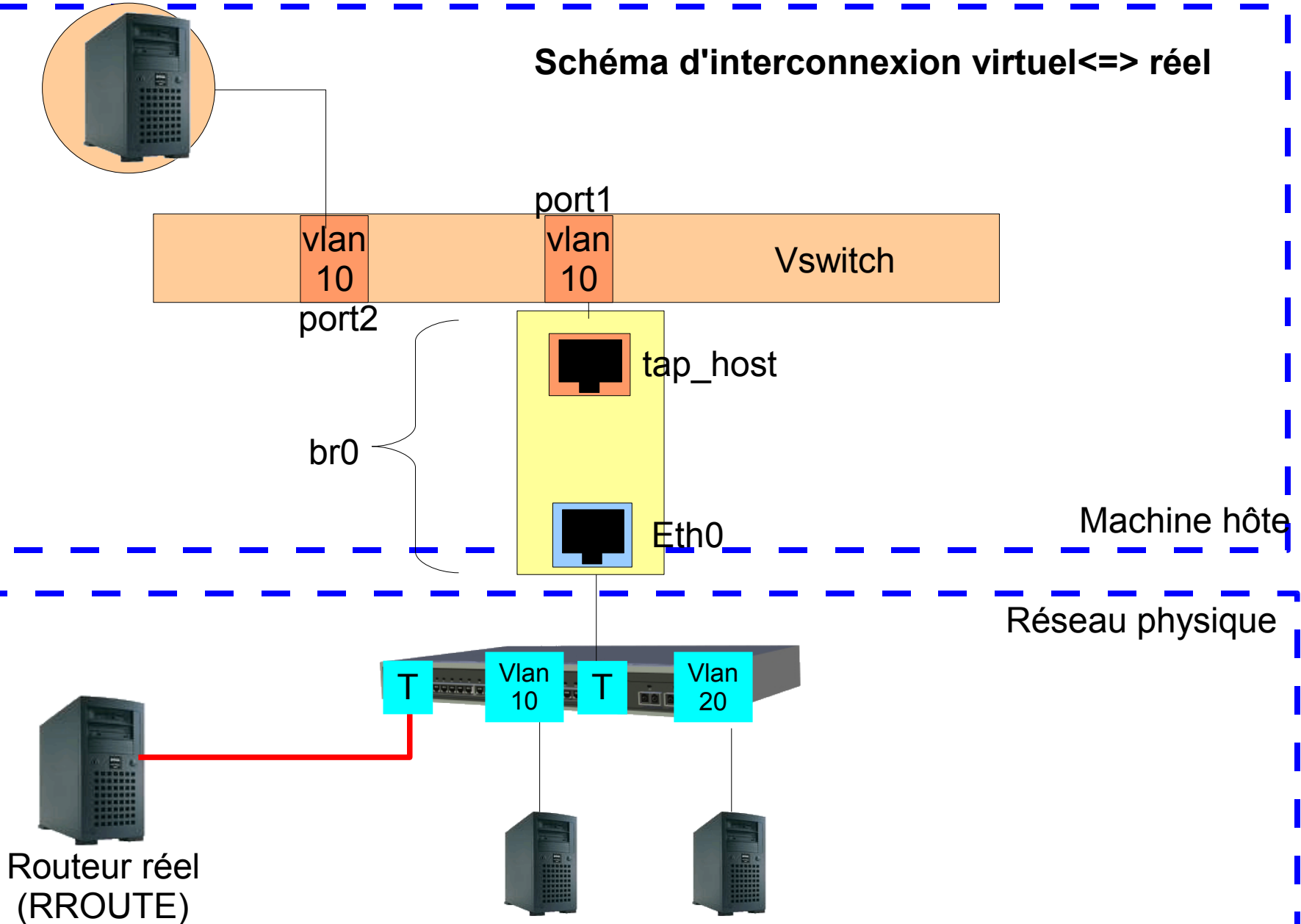
Création du monde...virtuel

■ Interconnexion du réseau virtuel et du réseau physique

- Créer une interface virtuelle TAP dans la machine hôte
- Créer le commutateur virtuel
- Connecter l'interface TAP sur le commutateur
- Connecter l'interface physique (eth0) sur un port tagged du commutateur physique
- Relier l'interface TAP à l'interface physique eth0 => pont

Création du monde...virtuel

Schéma d'interconnexion virtuel<=> réel



Création du monde...virtuel

Configurations


■ Créer l'interface tap

```
tunctl -t tap_host
```

■ Créer le commutateur virtuel

```
vde_switch -s /var/run/vde.ctl -tap tap_host
```

L'interface tap_host est automatiquement branchée sur le port 1 du commutateur virtuel



■ Faire un pont entre le eth0 et tap_host

```
brctl addbr br0  
brctl addif br0 eth0  
brctl addif br0 tap_host
```

```
ifconfig br0 up
```

Création du monde...virtuel

Configurations

■ Créer les ports sur le commutateur virtuel

`vde$ port/create 2` => Création du port 2 sur Vswitch

■ Créer les VLAN

`vde$ vlan/create 10` => Création du vlan 10 sur vswitch

`vde$ vlan/create 20` => Création du vlan 20 sur Vswitch

■ Affecter les VLAN aux ports

`vde$ port/setvlan 2 10` => Affectation statique du VLAN 10 sur le port 2

`vde$ port/addport 10 1` => Tag du VLAN 10 sur le port 1

`vde$ port/addport 20 1` => Tag du VLAN 20 sur le port 1

Création du monde...virtuel

Lancement de la machine virtuelle et connexion sur le commutateur virtuel

vdekvm

-cdrom ramux.MV1.iso

-m 500 ← *mémoire*

-net vde,vlan=0,sock=**/var/run/vde.ctl**,port=**2**

-net nic,model=**virtio**,macaddr=**00:16:3e:00:00:15**,vlan=0

-name MV1 &

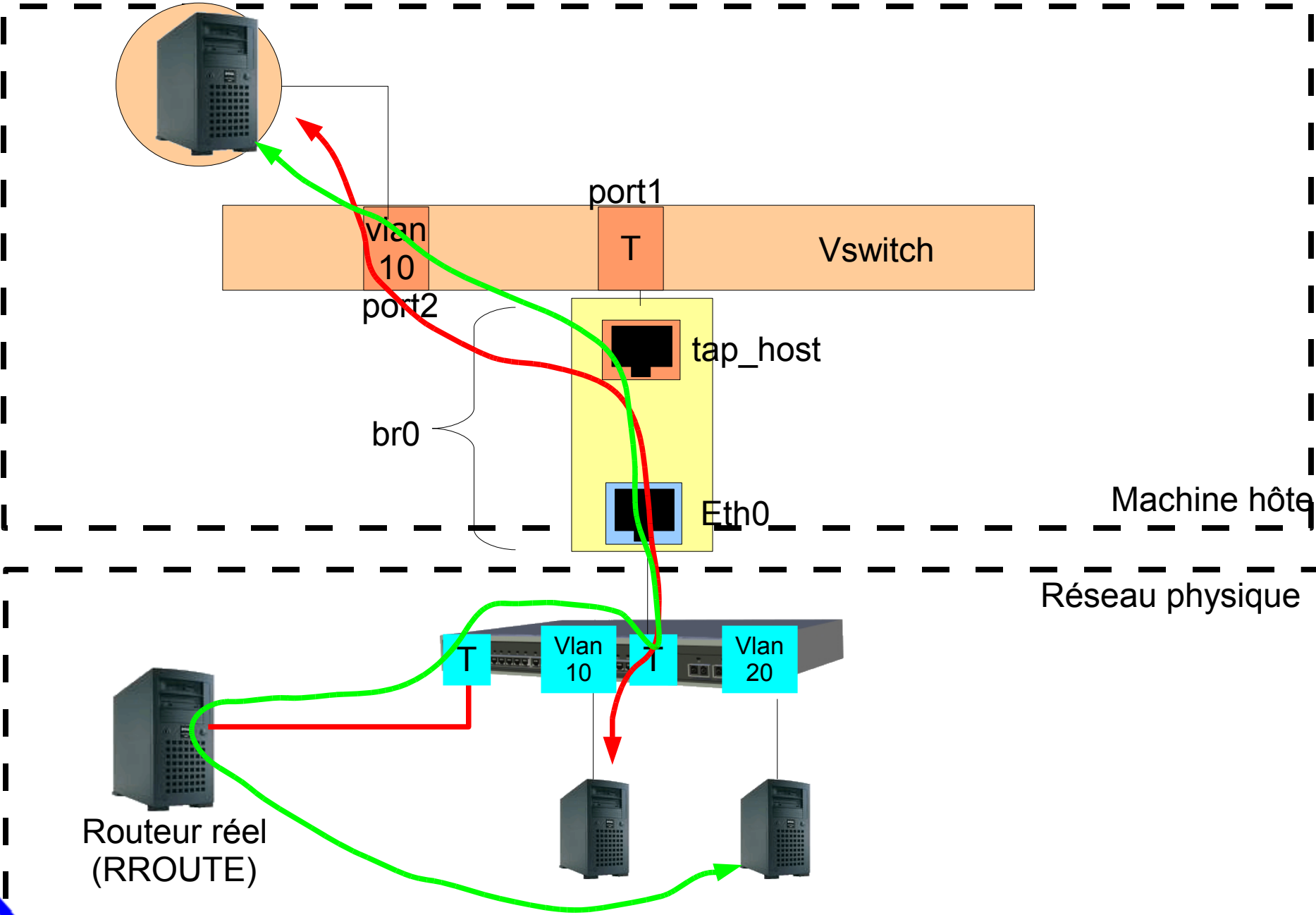
driver

Adresse MAC

Socket du commutateur virtuel

*Port de connexion
sur le commutateur virtuel*

Création du monde...virtuel



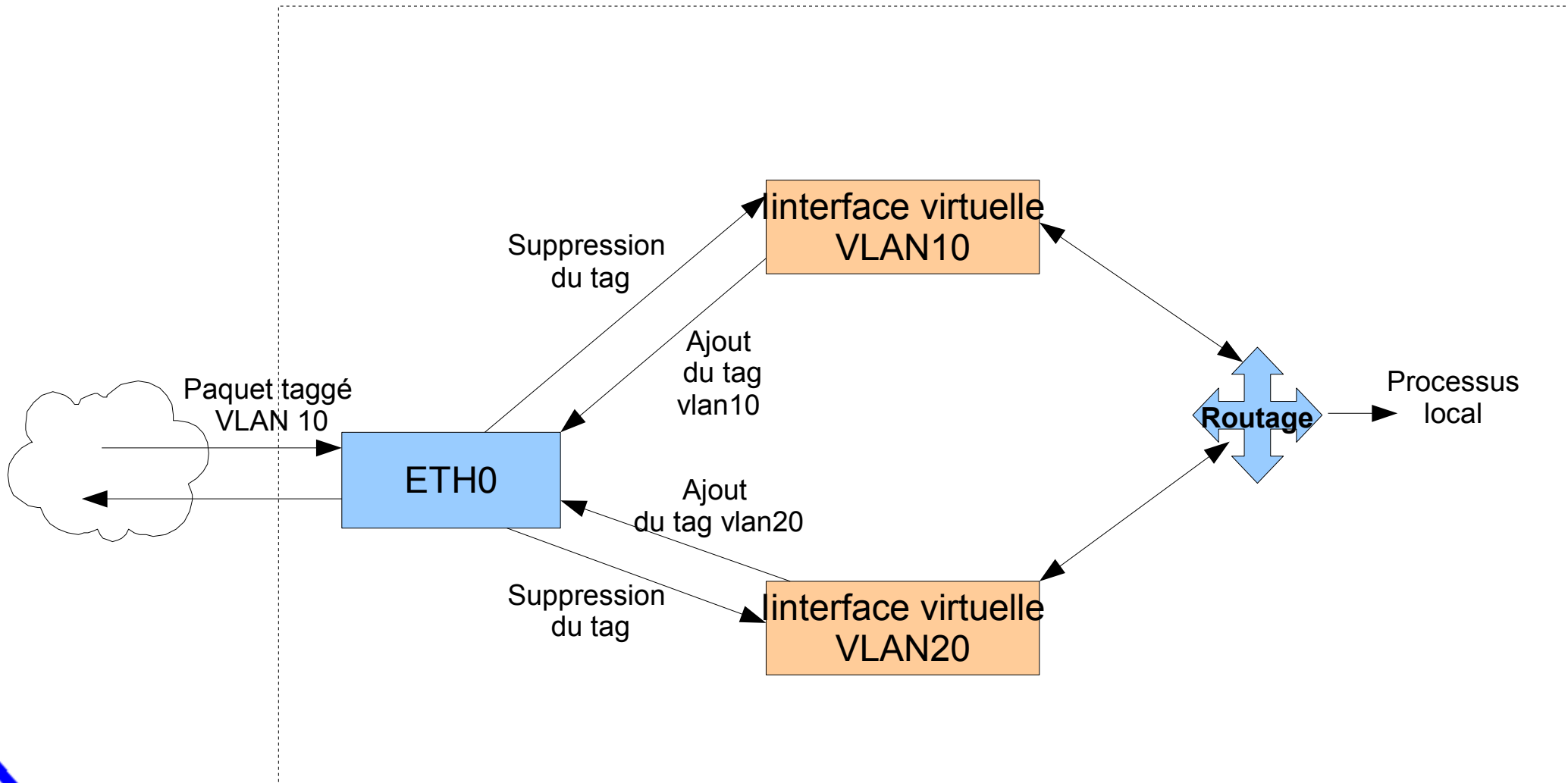
Création du monde...virtuel

Connexion de la machine hôte sur le réseau (réel)

- Dans le schéma précédent, la machine hôte n'est pas vue sur le réseau. Elle sert uniquement de pont entre commutateurs virtuels et réels.
- Pour connecter la machine hôte il faut lui créer une interface virtuelle pour le VLAN auquel elle doit appartenir et lui donner une adresse IP sur ce VLAN.

```
vconfig add eth0 vlan10
```

Création du monde...virtuel



Précautions à prendre

■ **DANGER : Si la machine hôte est sur plusieurs VLAN**

- Si une machine virtuelle définit la machine hôte comme gateway par défaut alors elle pourra communiquer avec un autre sous-réseau sans passer par le filtrage du routeur et la politique de sécurité risque d'être contournée.

■ Remèdes :

- Ne pas mettre la machine hôte sur plusieurs VLAN
- Ne pas lui permettre le routage
- La placer seule sur VLAN et filtrer ses communications vers le reste du réseau

Le monde virtuel est créé

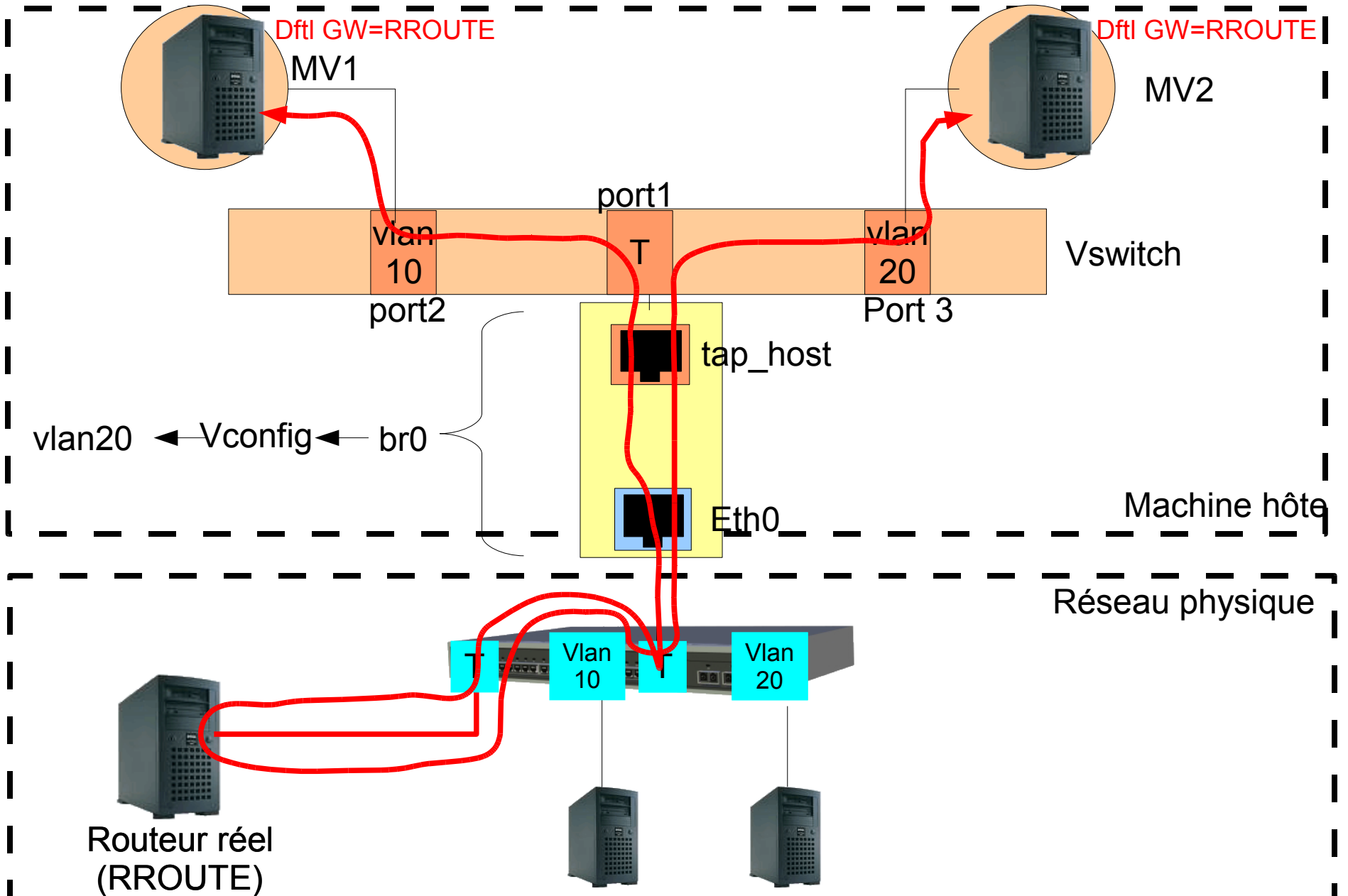
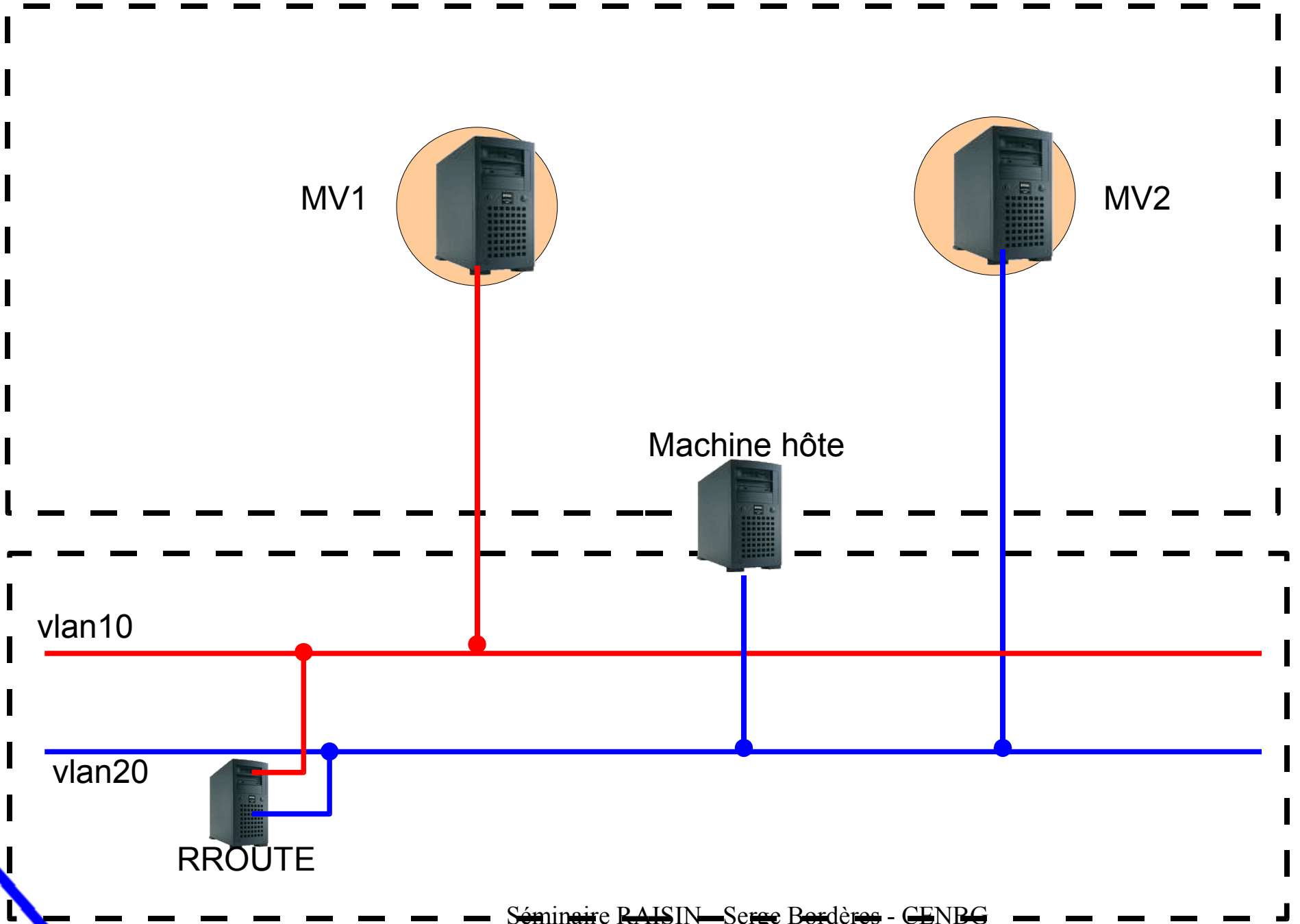
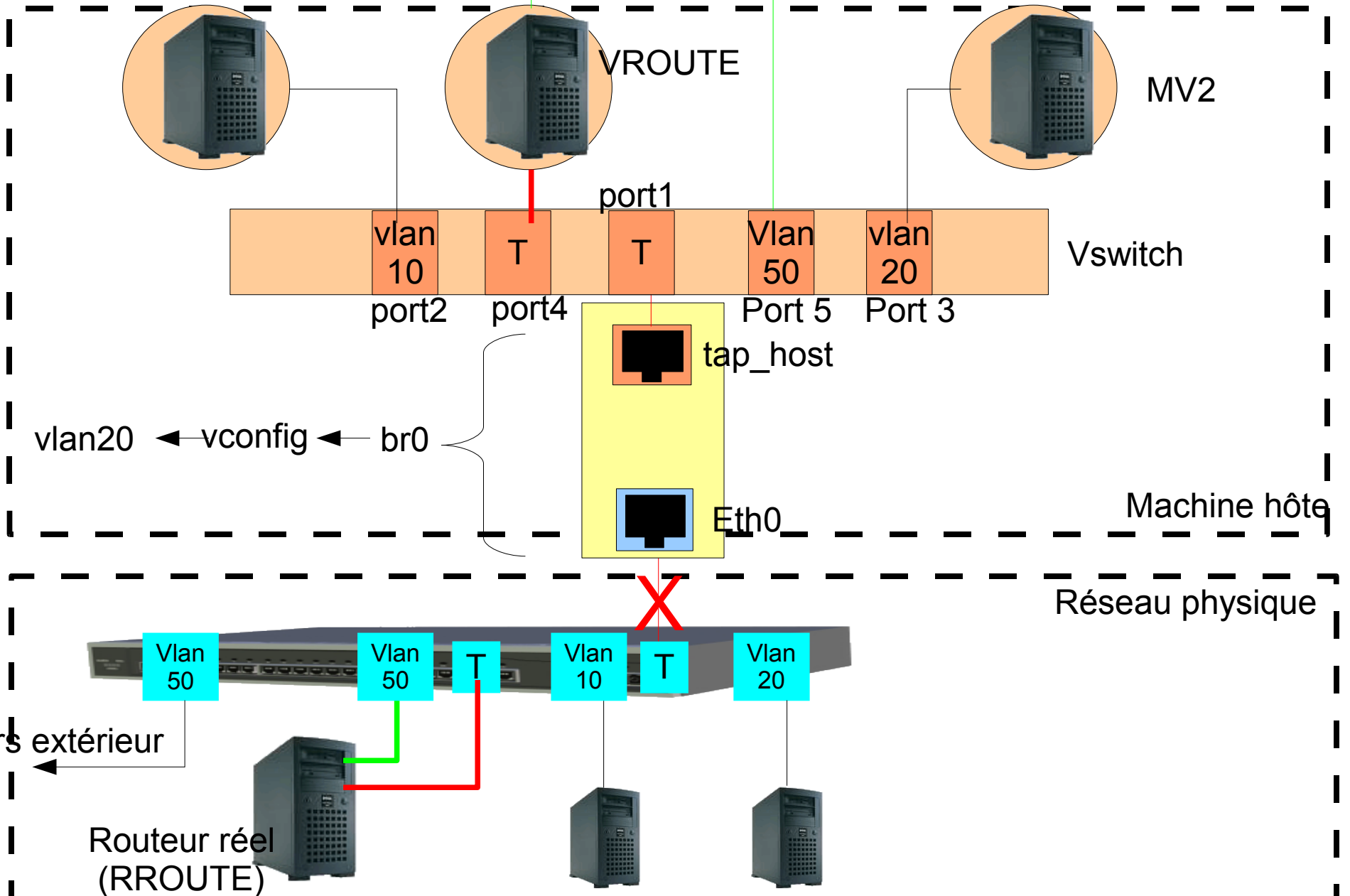


Schéma logique d'inter-connexion

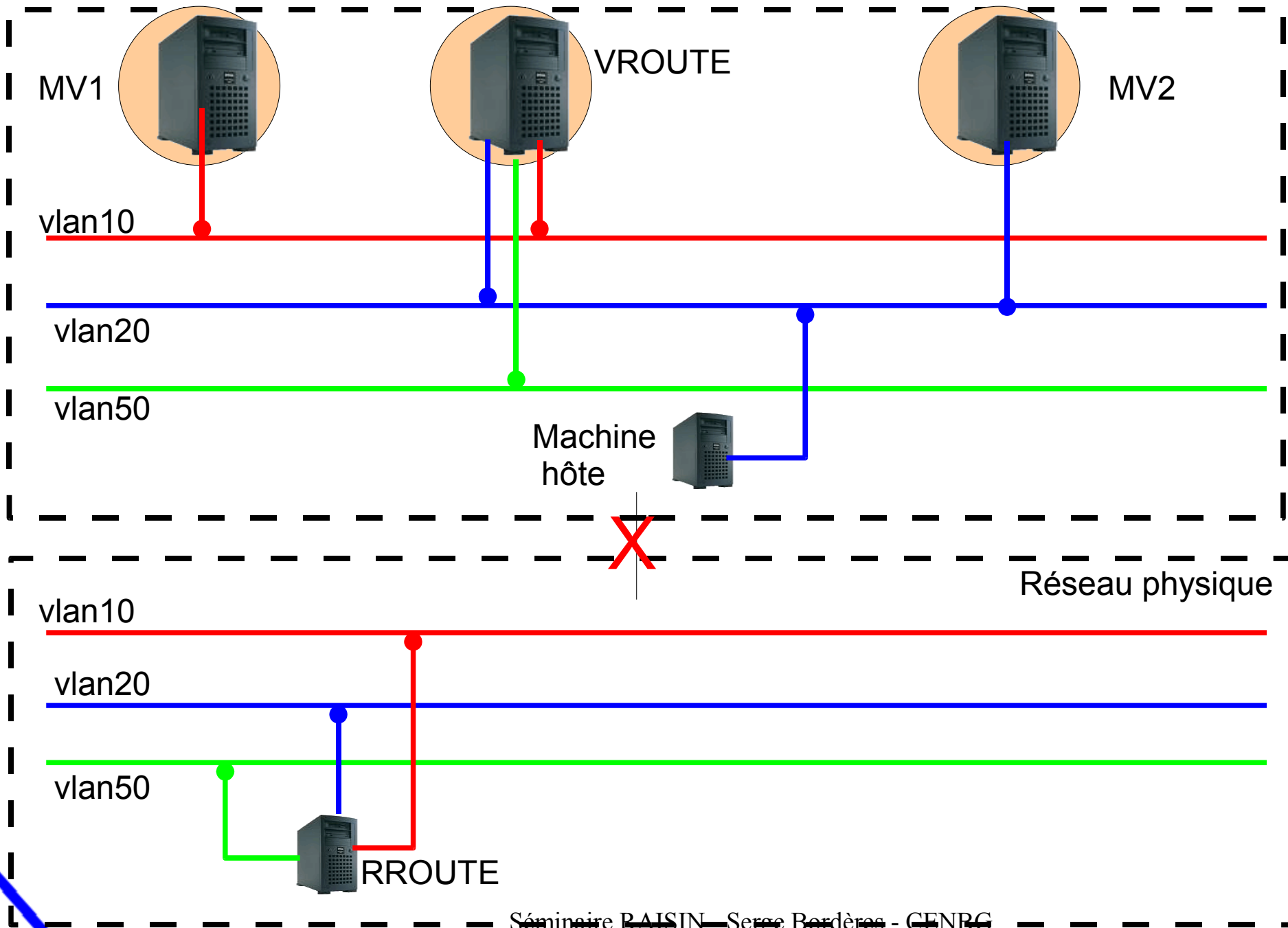


VIRTUALISATION DU ROUTEUR

Virtualisation du routeur



Virtualisation du routeur- schéma logique



Virtualisation du routeur

Lancement de la machine virtuelle du routeur et connexion sur le commutateur virtuel

vdekvm

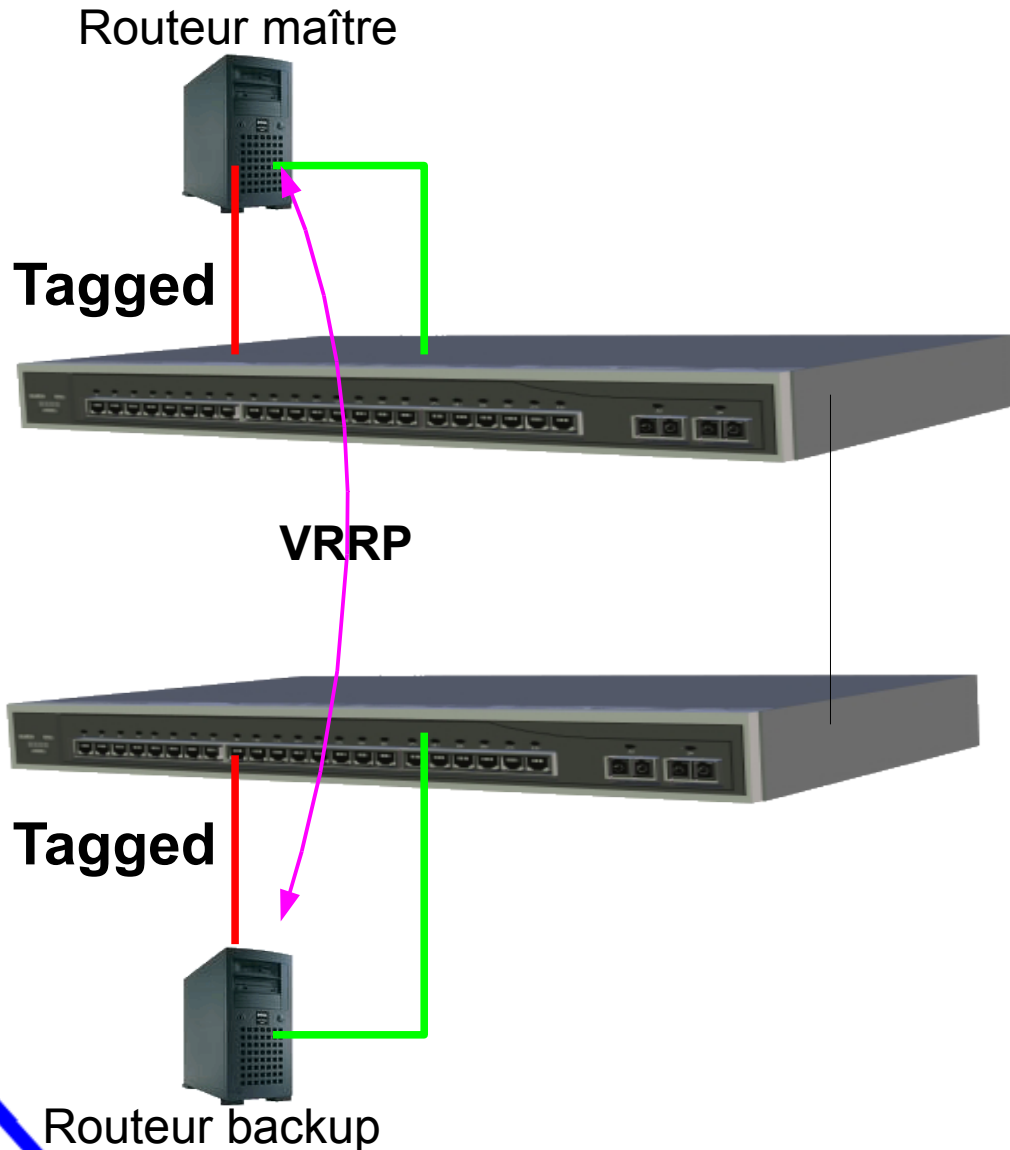
```
-cdrom ramux.RT1.iso  
-m 500  
-net nic,model=virtio,macaddr=00:16:3e:00:00:02,vlan=0  
-net vde,sock=/var/run/vdectl,port=4  
  
-net nic,model=virtio,macaddr=00:16:3e:00:00:03,vlan=1  
-net vde,vlan=1,sock=/var/run/vdectl,port=5  
-name RT1 i&
```

La 1ère interface Ethernet du routeur est placée sur le port 4

La 2ème interface Ethernet du routeur est placée sur le port 5

REDONDANCE DYNAMIQUE DE ROUTEURS

Principe de la redondance dynamique



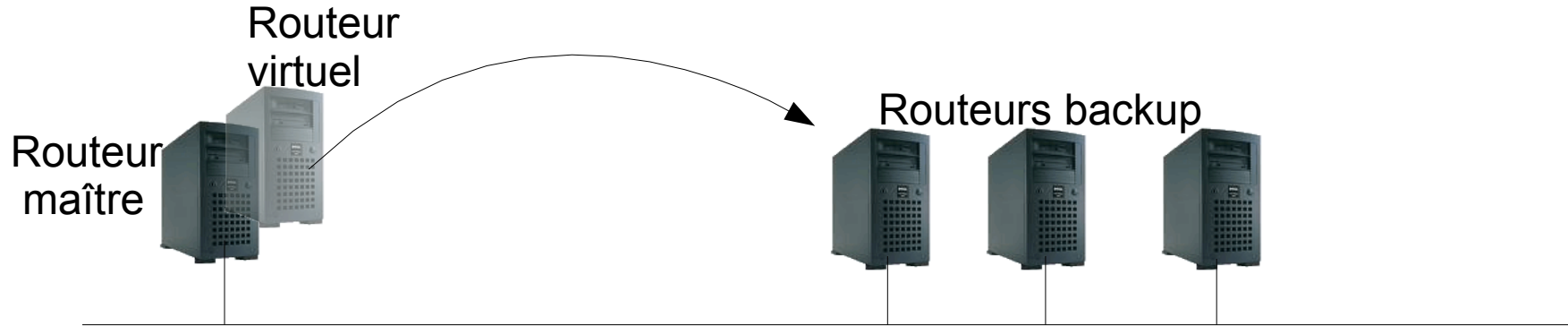
- Le routeur maître est gateway par défaut pour l'ensemble du réseau
- Le routeur backup est passif
- En cas de défaillance sur le maître, ou sur le commutateur du maître, le routeur backup devient maître et l'annonce au réseau
- Le protocole qui gère « l' élection d'un nouveau maître est VRRP

Le protocole VRRP

■ **VRRP**= Virtual Router Redondancy protocol (décrit dans RFC3768)

C'est un protocole de redondance de routeurs c'est-à-dire qui permet à un routeur de secours de prendre automatiquement la place d'un routeur principale en cas de défaillance

Le protocole VRRP



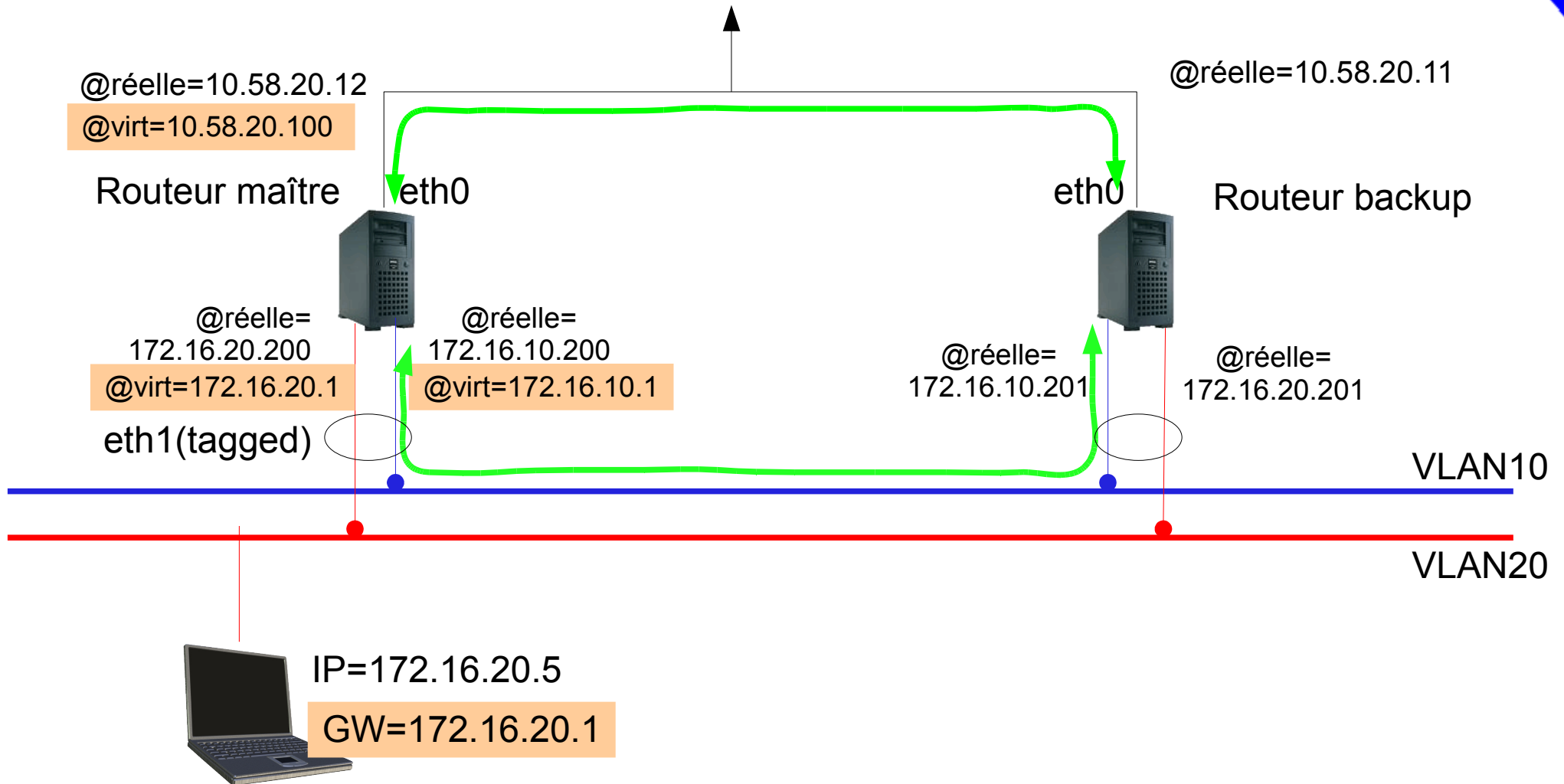
■ VRRP défini :

- Un routeur master initial (celui qui a la responsabilité du routage en temps normaux). Il a la priorité la plus grande.
 - Un ou plusieurs routeur(s) de backup
 - Un routeur virtuel placé sur le routeur master courant.
- Les clients du réseau ne voient que le routeur virtuel comme gateway par défaut
- En cas de défaillance du maître, il y a élection d'un nouveau master et le routeur virtuel se déplace sur cette machine.

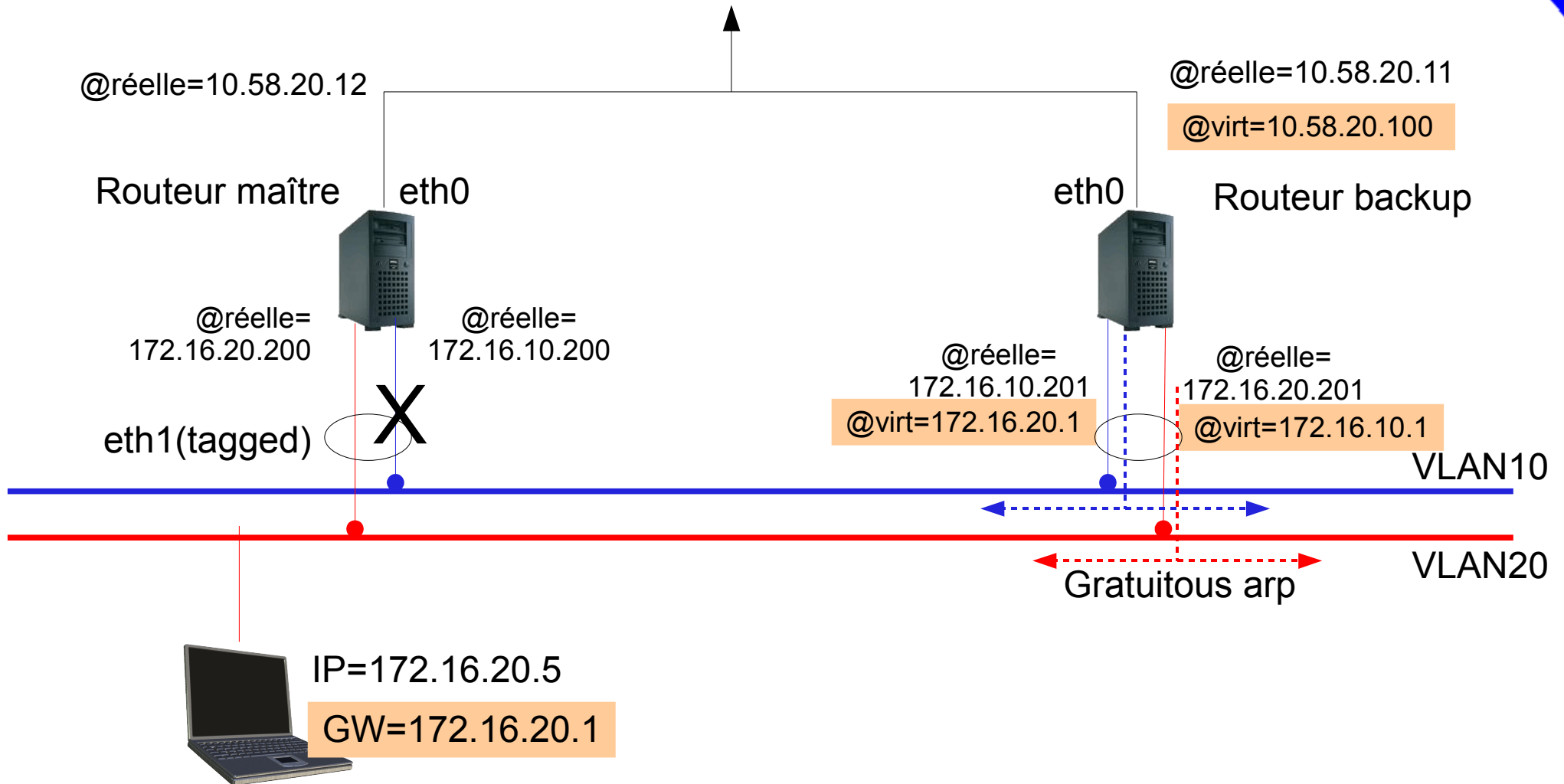
Le protocole VRRP

- Chaque interface (physique ou VLAN) du routeur master possède une adresse réelle
- Chaque interface de routeurs backup possède une adresse réelle
- Le routeur virtuel = une adresse virtuelle sur chaque interface du routeur dans l'état master.
- Le protocole prévoit également une adresse MAC virtuelle par interface mais dans Linux on ne peut pas avoir deux adresses MAC par interface.
- L'annonce du changement se fait par des gratuitous arp
- Le maître diffuse sa priorité (advert) sur le réseau à intervalle régulier (multicast sur 224.0.0.18)
- Les backup écoutent ces messages. Si un backup a une priorité supérieure ou s'il ne reçoit plus rien, il se déclare maître.
- Il n'y a pas de répartition de charge. Le (ou les) routeur(s) backup sont passifs.

Le protocole VRRP



Le protocole VRRP



Le protocole VRRP

Le paquetage KEEPALIVED implémente :

- Le protocole VRRP
- Le support de Linux Virtual Server (LVS)
 - Virtualisation d'un groupe de machines (cluster) permettant le partage de charge et la redondance.

Configuration VRRP avec KEEPALIVED

Configuration sur le MASTER

```
vrrp_sync_group grp1 {  
  group {  
    interne  
    externe  
  }  
}
```

```
vrrp_instance interne {  
  state MASTER  
  interface vlan10  
  virtual_router_id 100  
  priority 100  
  advert_int 5  
  authentication {  
    auth_type PASS  
    auth_pass azerty  
  }  
  virtual_ipaddress {  
    172.16.10.1/24 brd 172.16.255.255 dev vlan10  
    172.16.20.1/24 brd 172.16.255.255 dev vlan20  
    10.58.20.100/24 brd 10.58.20.255 dev eth0  
  }  
}
```

```
vrrp_instance externe {  
  state MASTER  
  interface eth0  
  virtual_router_id 101  
  priority 100  
  advert_int 5  
  authentication {  
    auth_type PASS  
    auth_pass azerty  
  }  
  virtual_ipaddress {  
    172.16.10.1/24 brd 172.16.255.255 dev vlan10  
    172.16.20.1/24 brd 172.16.255.255 dev vlan20  
    10.58.20.100/24 brd 10.58.20.255 dev eth0  
  }  
}
```

Configuration VRRP avec KEEPALIVED

```
vrrp_sync_group grp1 {  
    group {  
        interne  
        externe  
    }  
}
```

```
vrrp_instance interne {  
    state BACKUP  
    interface vlan10  
    virtual_router_id 100  
    priority 50  
    advert_int 5  
    authentication {  
        auth_type PASS  
        auth_pass azerty  
    }  
    virtual_ipaddress {  
        172.16.10.1/24 brd 172.16.255.255 dev vlan10  
        172.16.20.1/24 brd 172.16.255.255 dev vlan20  
        10.58.20.100/24 brd 10.58.20.255 dev eth0  
    }  
}
```

Configuration sur le BACKUP

```
vrrp_instance externe {  
    state MASTER  
    interface eth0  
    virtual_router_id 101  
    priority 50  
    advert_int 5  
    authentication {  
        auth_type PASS  
        auth_pass azerty  
    }  
    virtual_ipaddress {  
        172.16.10.1/24 brd 172.16.255.255 dev vlan10  
        172.16.20.1/24 brd 172.16.255.255 dev vlan20  
        10.58.20.100/24 brd 10.58.20.255 dev eth0  
    }  
}
```

Mise en oeuvre de VRRP

- Le routeur backup doit reprendre le rôle du master pourtant sa configuration réseau n'est pas strictement identique :
 - Pas les mêmes adresses réelles des interfaces
 - Pas la même priorité
- En cas de modification dans le master (ajout d'un vlan par exemple) il faut la reporter manuellement dans le backup, on ne pas simplement copier un fichier dans l'autre
- Il faut faire en sorte que les deux configurations soient strictement identiques
- Seul le rôle (master ou backup) doit différer et servir à paramétrer les configurations

Solution : Labwall

Mise en oeuvre de VRRP

- Configuration d'un routeur sans VRRP avec Labwall (un seul routeur)

Définition des interfaces externes et internes

EXTERNALDEV=eth1

INTERNALDEV=eth0

NETINIT \$EXTERNALDEV \$INTERNALDEV

#VLAN	NVlan	device	@IPnetwork	@IPgateway	netmask	broadcast	Nom
VLAN	10	\$INTERNALDEV	172.16.10.0	172.16.10.1	255.255.255.0	172.16.255.255	VLAN10
VLAN	20	\$INTERNALDEV	172.16.20.0	172.16.20.1	255.255.255.0	172.16.255.255	VLAN20

Mise en oeuvre de VRRP

■ Configuration de 2 routeurs avec VRRP avec Labwall

./etc/cdnuxparams

.....

NETINIT \$EXTERNALDEV \$INTERNALDEV

#

SET ROLE=\$ROLE

#

VRRP AUTH PASS XYZ `hostname` 172.16.10.12 admin@test.fr pooling 5

DEV INTER \$EXTERNALDEV 10.1.1.0 10.1.1.1 255.255.255.0 10.1.1.2-10.1.1.3-10.1.1.4-101*

#VLAN	NVlan	device	@IPnetwork	@IPgateway	netmask	broa
-------	-------	--------	------------	------------	---------	------

VLAN	10	\$INTERNALDEV	172.16.10.0	172.16.10.100-172.16.10.200-172.16.10.1-100*	255.255.255.0
------	----	---------------	-------------	--	---------------	-------

VLAN	20	\$INTERNALDEV	172.16.20.0	172.16.20.100-172.16.20.200-172.16.20.1-100	255.255.255.0
------	----	---------------	-------------	---	---------------	-------

Adresse réelle de l'interface
sur le MASTER initial

Adresse réelle de l'interface
sur le BACKUP initial

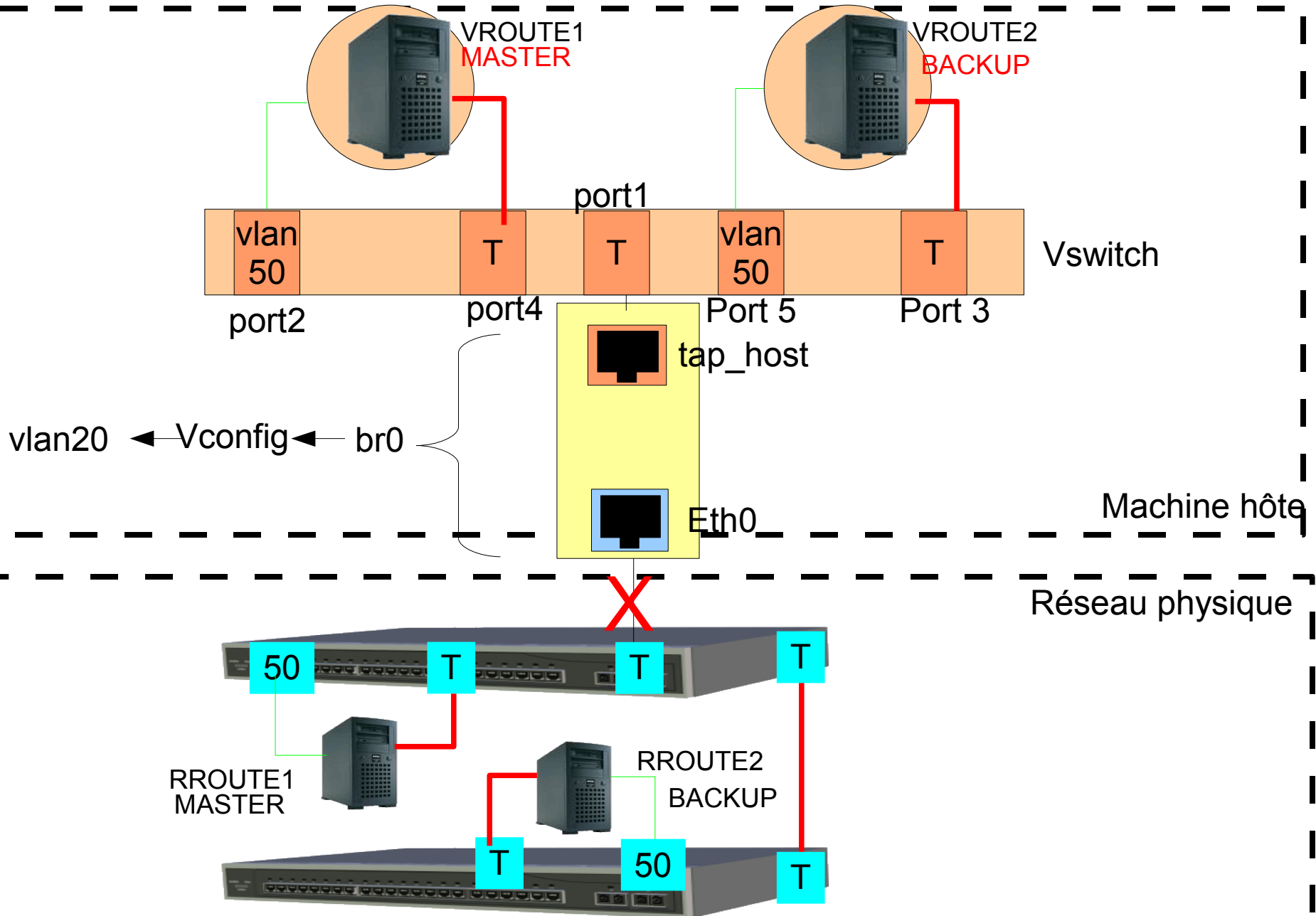
Adresse virtuelle du routeur
sur cette interface

Interface par laquelle se
fait le pooling

VRRP START GROUP eth1-vlan10

Génère la configuration keepalived

Configuration virtuelle avec VRRP



Configuration virtuelle avec VRRP

- Tests de fonctionnement des configurations
- Bonne génération des configurations réseau (adresses IP, routage...)
- Tests de validité des configurations VRRP: On test le comportement des routeurs maître/backup dans différentes situations
- Bonne génération des filtres Netfilter
- Tests des filtres : Une machine virtuelle peut être placé virtuellement à l'extérieur du réseau (virtualisation d'Internet)
- Analyse de problèmes rencontré dans le réel
- Remplacement du backup réel par une machine virtuelle

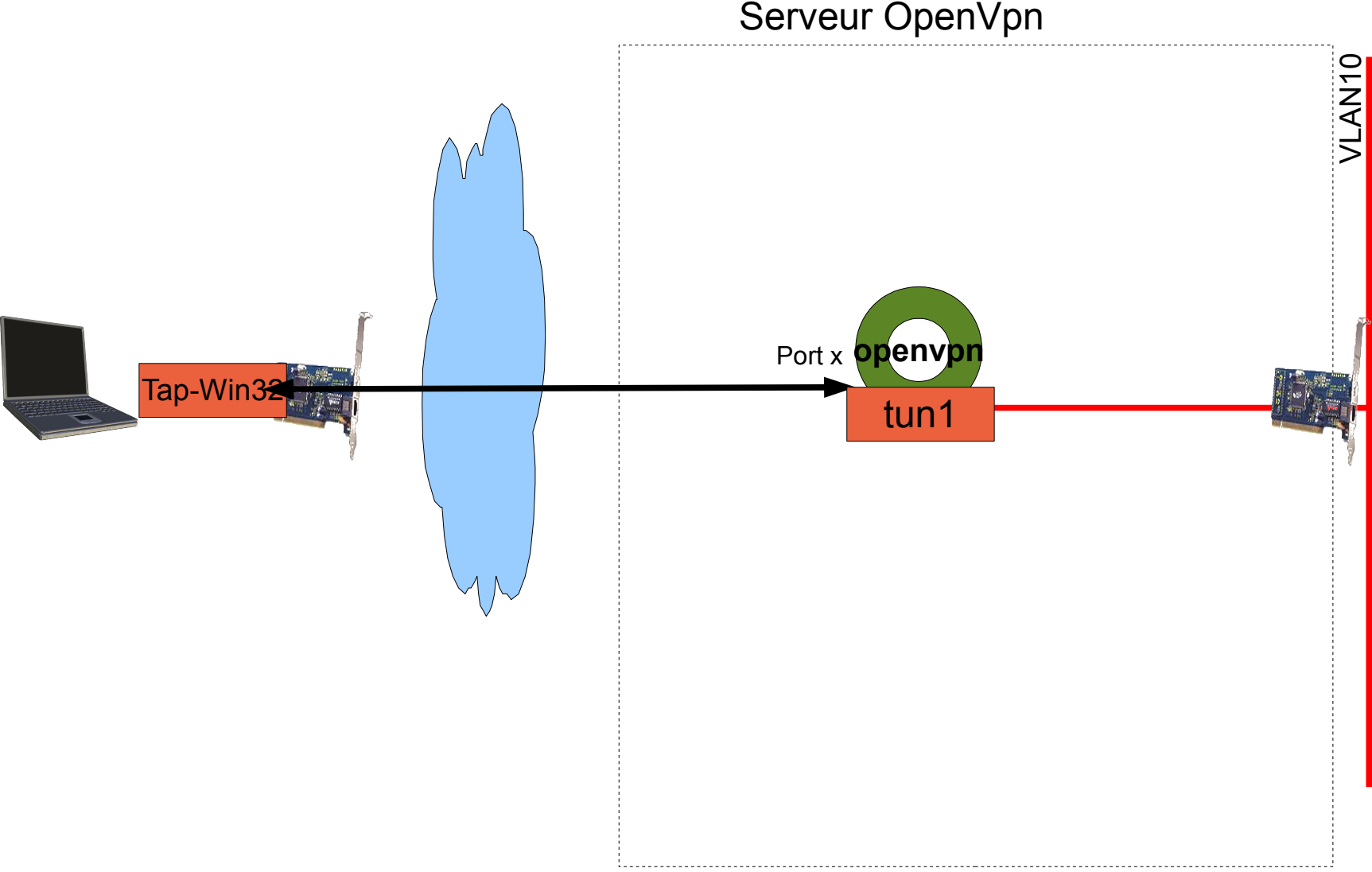


Application avec OPENVPN ou Comment virtualiser un serveur OpenVpn multi-vlan

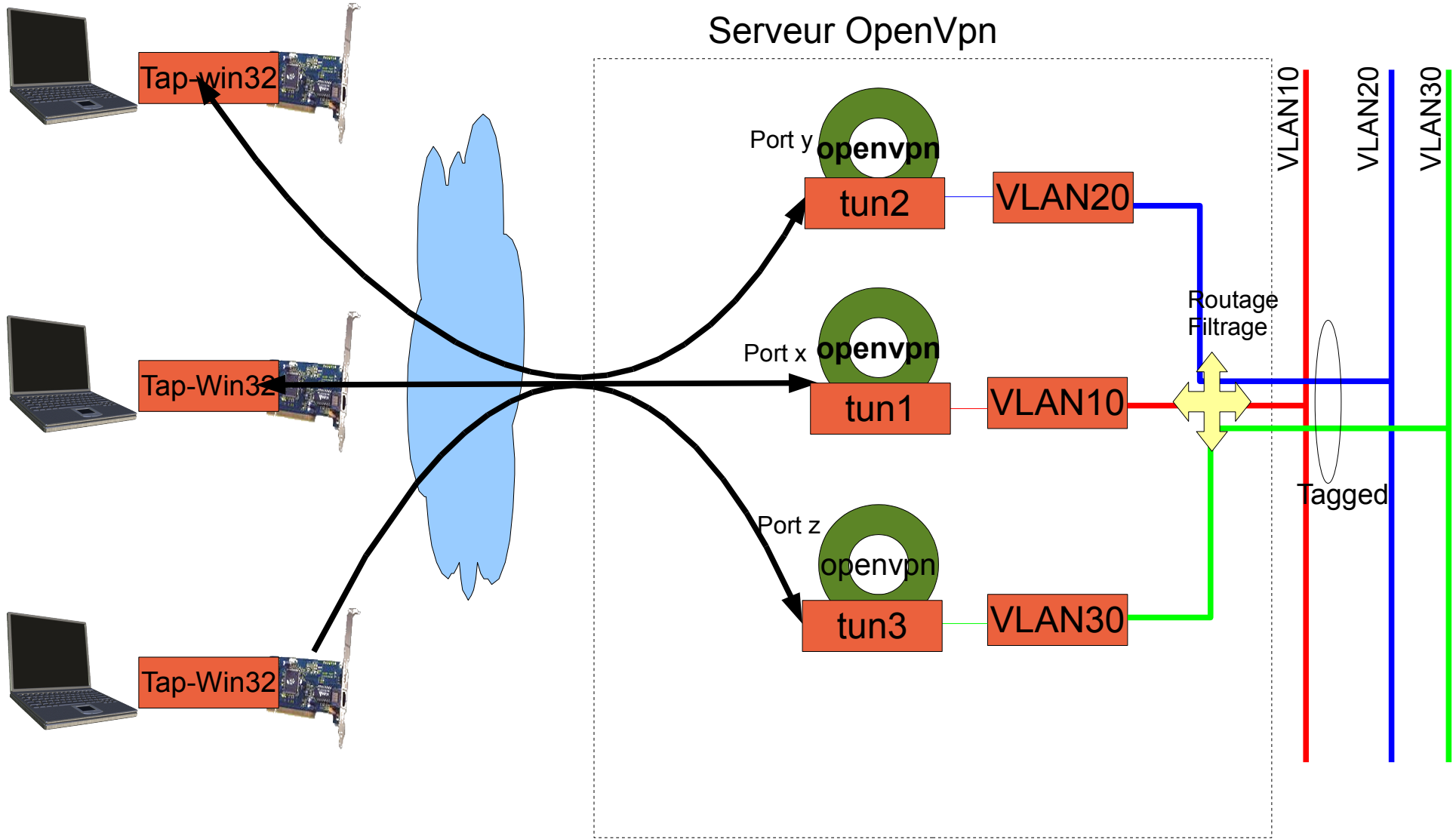
OpenVpn – single VLAN

- Le serveur VPN a une patte tournée vers l'extérieur et une patte dans un sous-réseau interne.
- Tout les utilisateurs qui se connectent sont vus sur ce sous-réseau (i.e ont une adresse IP dans ce sous-réseau)
- Les poste de travail contactent le serveur sur un port spécifique grâce à un logiciel client
- Le chiffrement est SSL
- Après authentification (certificats) un tunnel est créé
- Le tunnel est établi entre une interface virtuelle TAP-WIN32 côté client et tun/tap côté serveur.

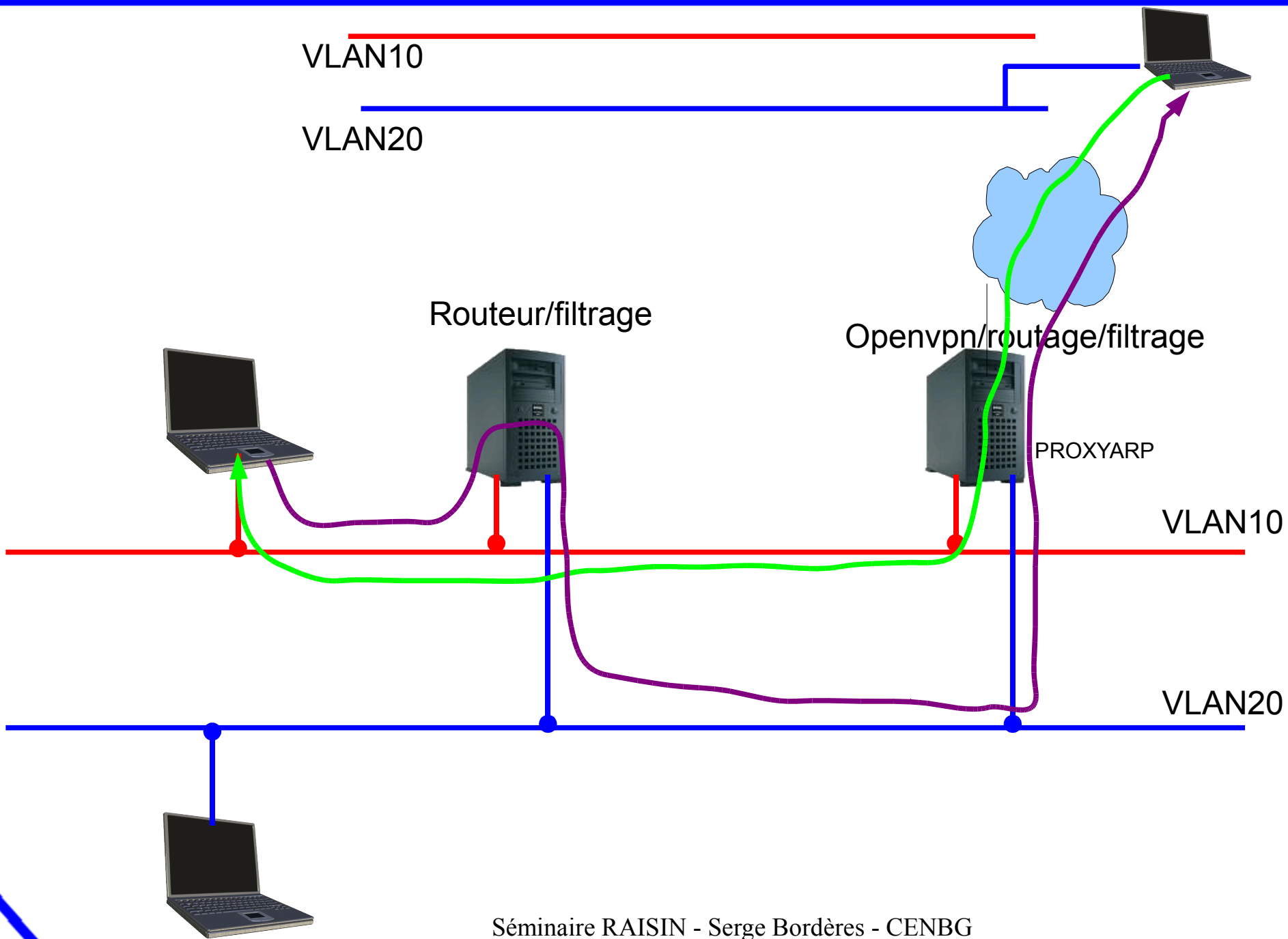
OpenVpn – single VLAN



OpenVpn - multi VLAN



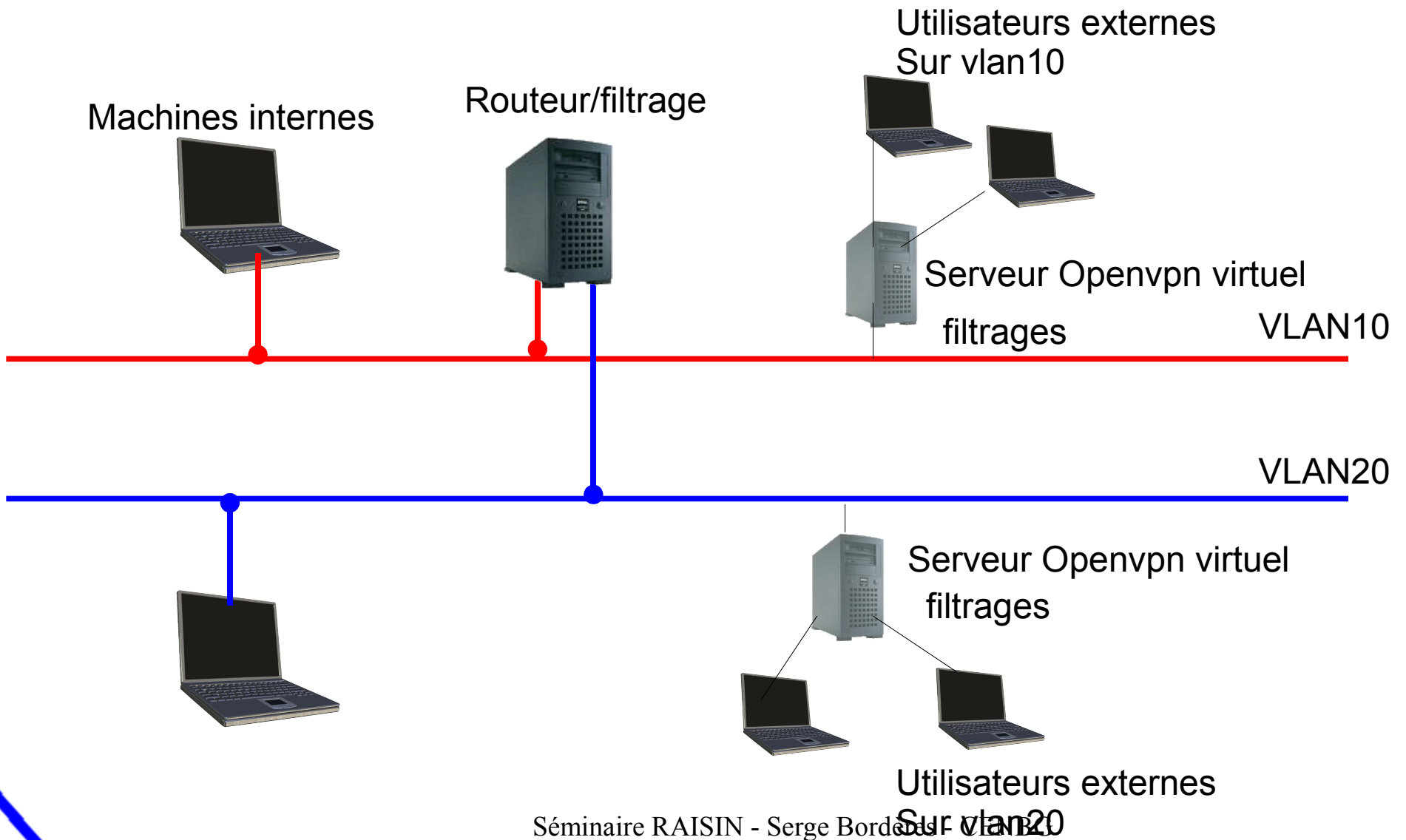
OpenVpn - multi VLAN



OpenVpn –multi VLAN

Vision logique :

Tout se passe comme s'il y avait un serveur VPN single-Vlan sur chaque VLAN.
Il filtre les communications entre tous les postes du VLAN



OpenVpn –multi VLAN

■ Problème 1

- Dans la configuration précédente les clients se connectent sur un port correspondant à leur VLAN.
- Ce qui signifie que ces ports doivent être ouverts en sortie du réseau d'où ils arrivent. Ca peut rapidement être un problème.

■ Solution : Reverse proxy

- Connexion sur un port unique en HTTP-CONNECT
- Le port final est passé dans le protocole HTTP-CONNECT
- Le port sera choisi de façon à minimiser les risques de filtrage en sortie (port standard d'Openvpn = 1194)

OpenVpn –multi VLAN

Extrait de la configuration cliente :

Remote-random

Port d'écoute du daemon du VLAN 10

```
<connection>  
remote vpn1.demo.fr
```

1110

A circle containing the number 1110, with an arrow pointing from the text 'Port d'écoute du daemon du VLAN 10' to it.

```
http-proxy vpn1.demo.fr  
</connection>
```

443

A circle containing the number 443, with an arrow pointing from the text 'Port d'écoute du Proxy' to it.

Port d'écoute du Proxy

```
<connection>  
remote vpn2.demo.fr
```

1110

A circle containing the number 1110, with an arrow pointing from the text 'Port d'écoute du daemon du VLAN 10' to it.

```
http-proxy vpn2.demo.fr  
</connection>
```

443

A circle containing the number 443, with an arrow pointing from the text 'Port d'écoute du Proxy' to it.

OpenVpn –multi VLAN

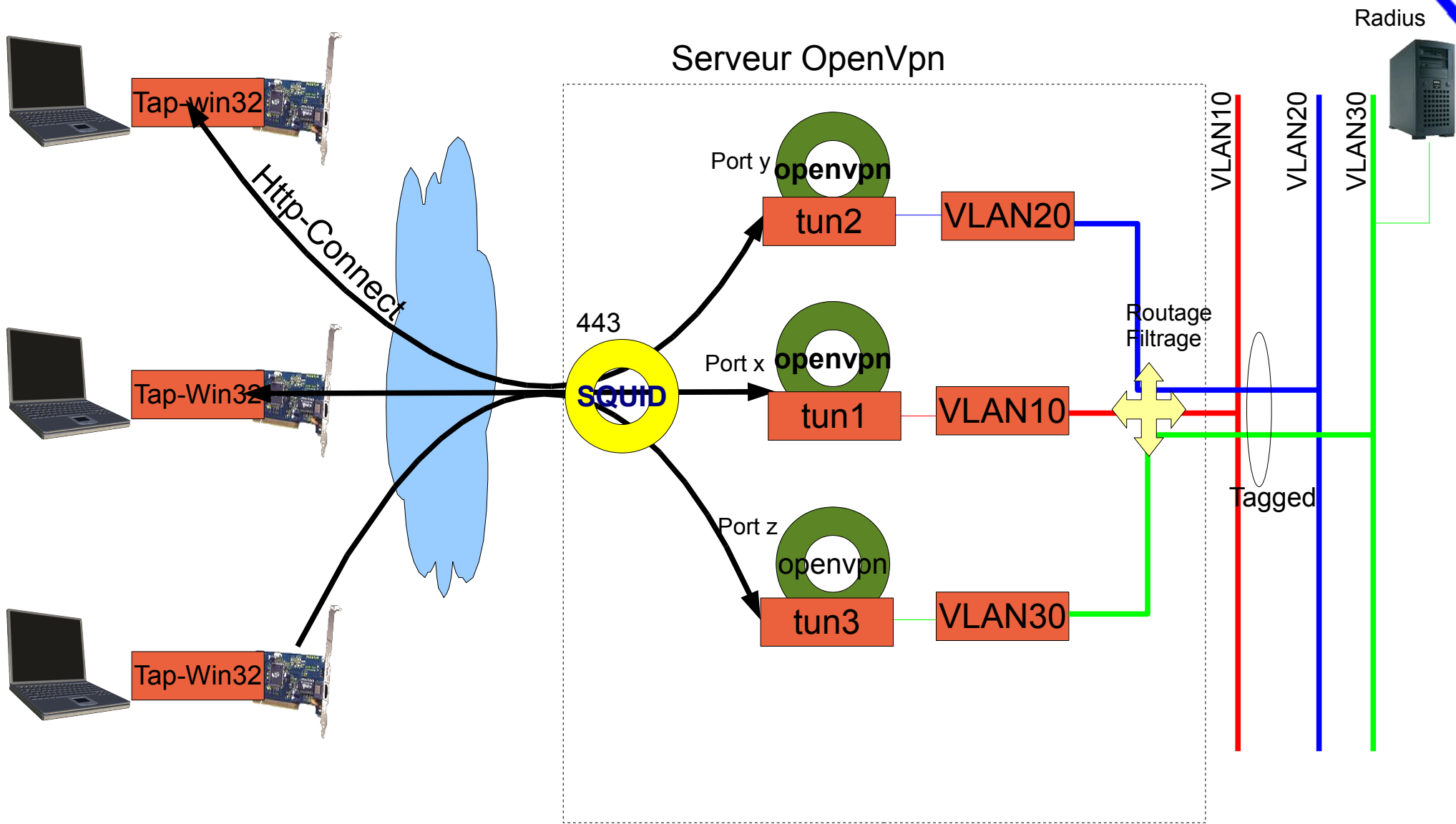
■ Problème 2

- ✦ Un client ne doit pas pouvoir se connecter sur un autre VLAN que le sien.

■ **Solution** :Utilisation l'option client-connect qui permet l'exécution d'un script qui:

- ✦ Interroge un serveur Radius pendant la phase de connexion pour obtenir l'autorisation et l'adresse IP du client.
- ✦ Vérifie s'il y a correspondance entre le VLAN demandé et l'adresse IP renvoyée par le serveur. Si non, la connexion est refusée.

OpenVpn - multi VLAN



OpenVpn –multi VLAN

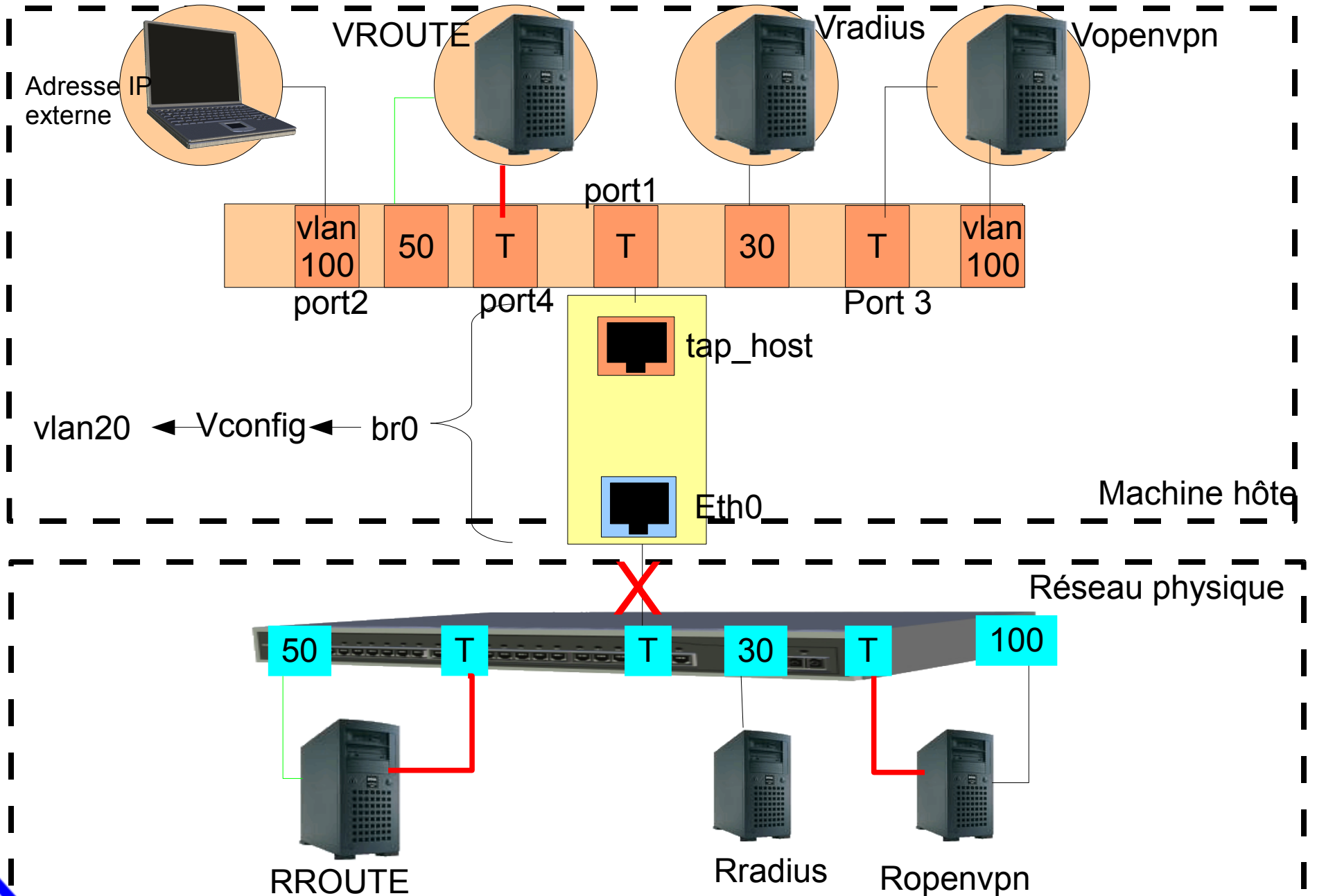
■ Pour virtualiser complètement la solution il faut :

- Virtualiser le serveur VPN avec une interface tagged et une interface sur le réseau externe
- Virtualiser le routeur gateway par défaut
- Virtualiser le serveur Radius

■ Pour virtualiser uniquement le serveur VPN il faut :

- Virtualiser le serveur VPN avec une interface tagged et une interface sur le réseau externe
- Le routeur gateway par défaut reste au niveau réel ainsi que le serveur Radius

OpenVpn virtualisé



PERFORMANCES

Performances

- Pas vraiment de tests de performance CPU réalisés
- Mais une constatation : Lenteur d'exécution de script Bash

Exemple :

- Test sur PC Latitude D430 dual-core
- KVM-65
- Fedora 9 , kernel 2.6.26

- Exécution du script Labwall de positionnement des règles Netfilter :
Environ 230 macros (shell scripts) qui génèrent environ 650 règles Netfilter

Exécution en machine virtuelle = 6 x le temps dans la machine hôte

Est-ce que pour un routeur le critère CPU est prépondérant ?

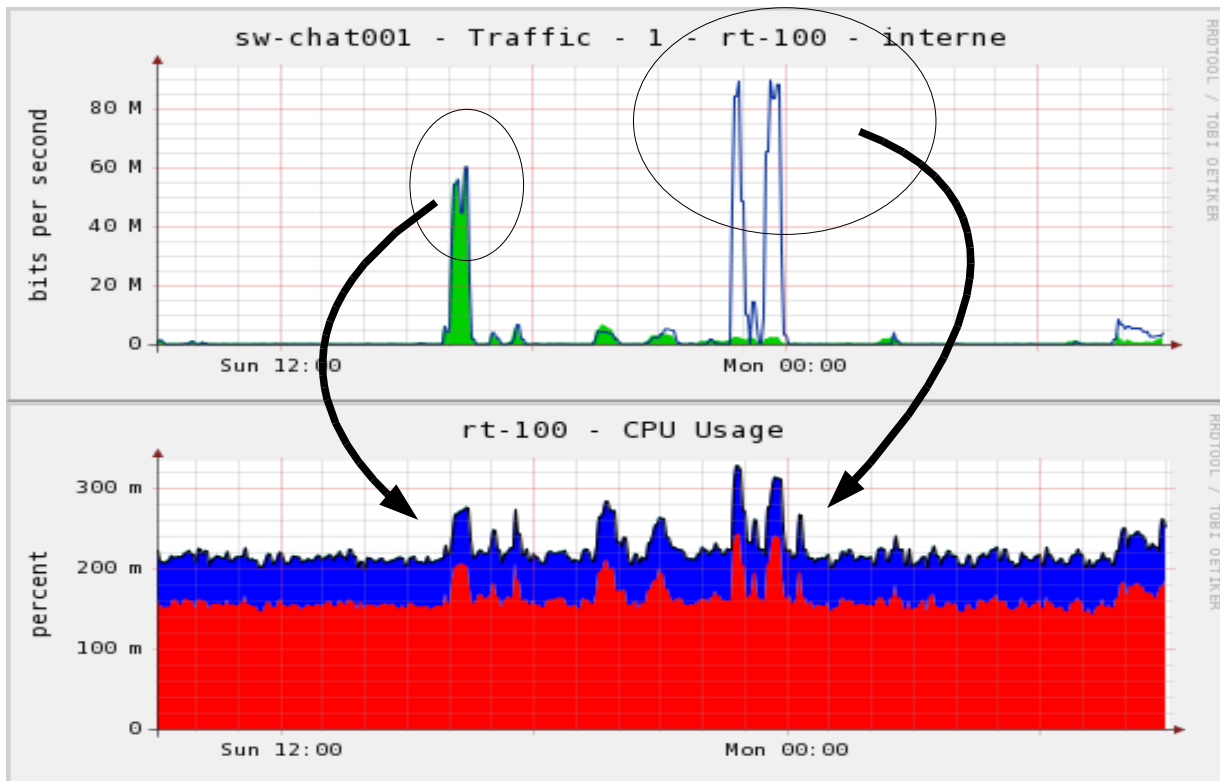
Performances

Influence du trafic sur la charge CPU d'un routeur (Linux/netfilter)

Données obtenues sur le routeur réel

DELL POWEREDGE 1950 DUAL-CORE 3,16GHz

- L'augmentation de la charge réseau a peu de conséquence sur la charge CPU
- L'élément déterminant pour la virtualisation sera plutôt la performance des E/S réseau



Performances

Comparaison des débits réseau

■ Tests de performances réalisés entre :

- + Machine virtuelle vers machine réelle
- + Machine hôte vers la même machine réelle

■ Conditions:

+ PC machine hôte :

- DELL Latitude D430
- CPU Intel double-coeur 1,2 Ghz
- Interface Ethernet: Broadcom BCM 5752

+ PC machine réelle :

- DELL Precision 380
- CPU Intel 3,2 Ghz
- Interface Ethernet: Broadcom BCM 5751

+ Liaison en 100Mb/s entre la machine hôte et la machine réelle

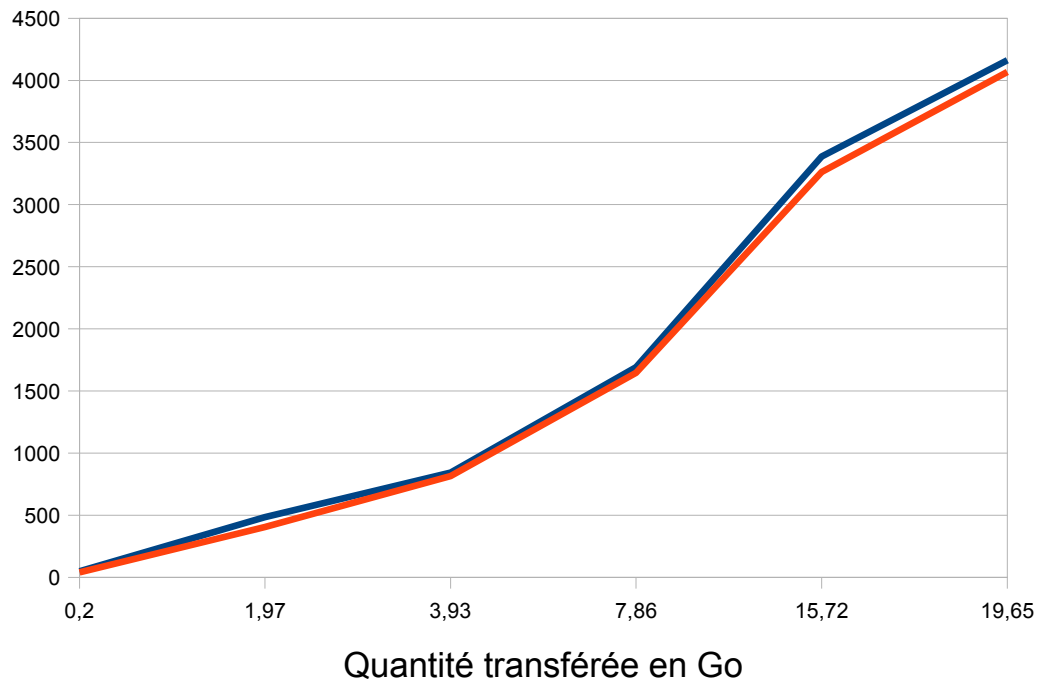
+ KVM-65

+ Utilisation du driver **virtio**

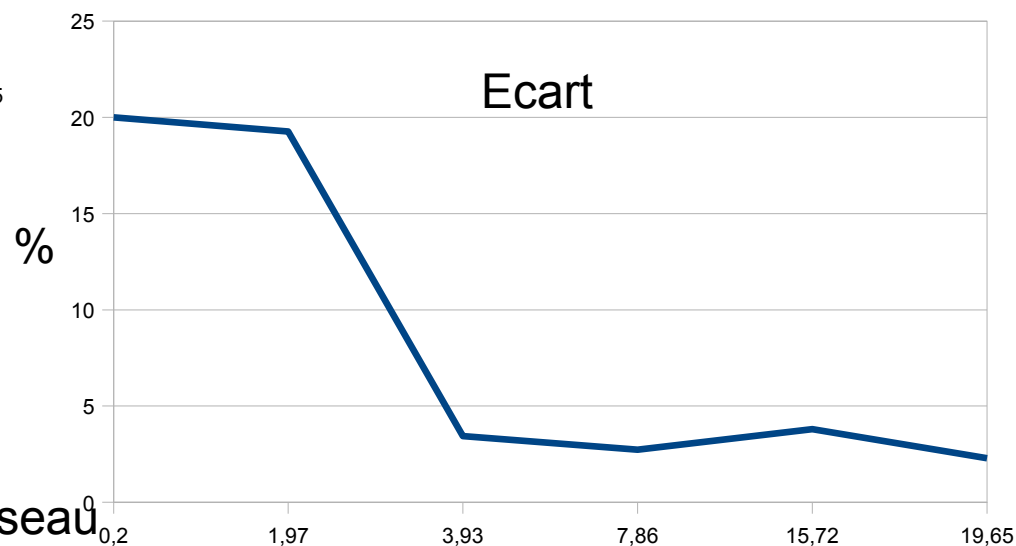
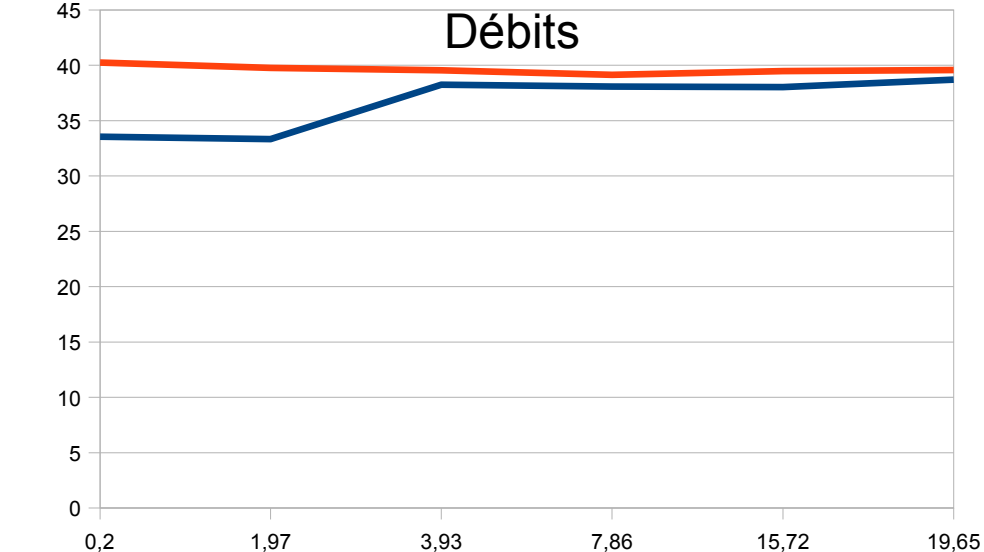
Performances

1er test : Transfert de fichiers SCP vers la machine réelle

Temps de transfert (sec)



Mb/s



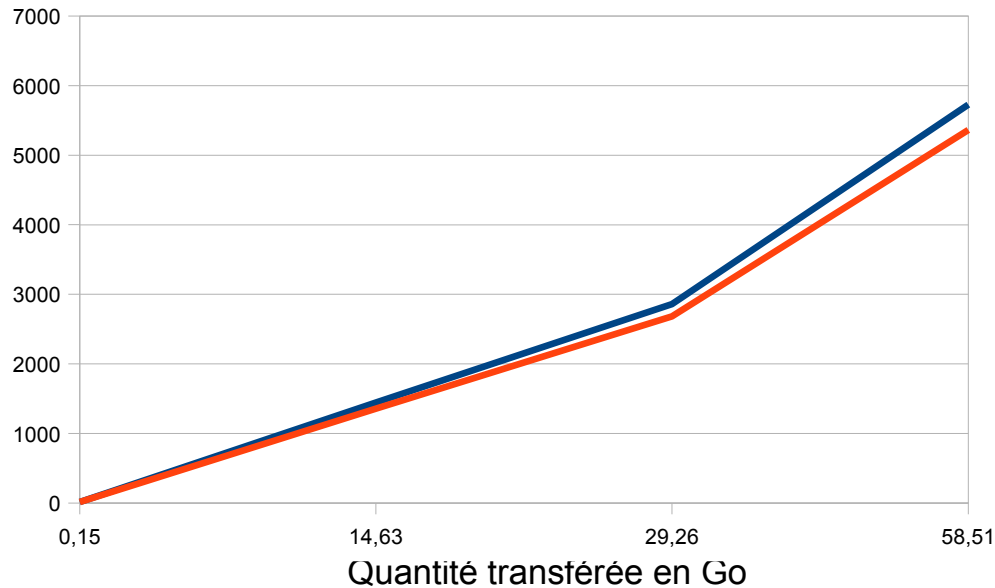
— Transfert Machine virtuelle
vers une machine réelle du réseau

— Transfert Machine hôte
vers la même machine réelle du réseau

Performances

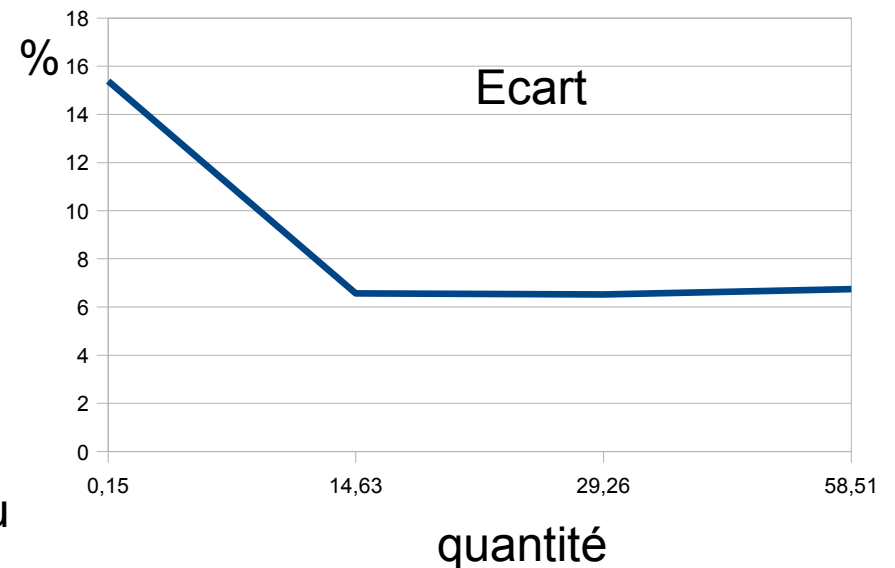
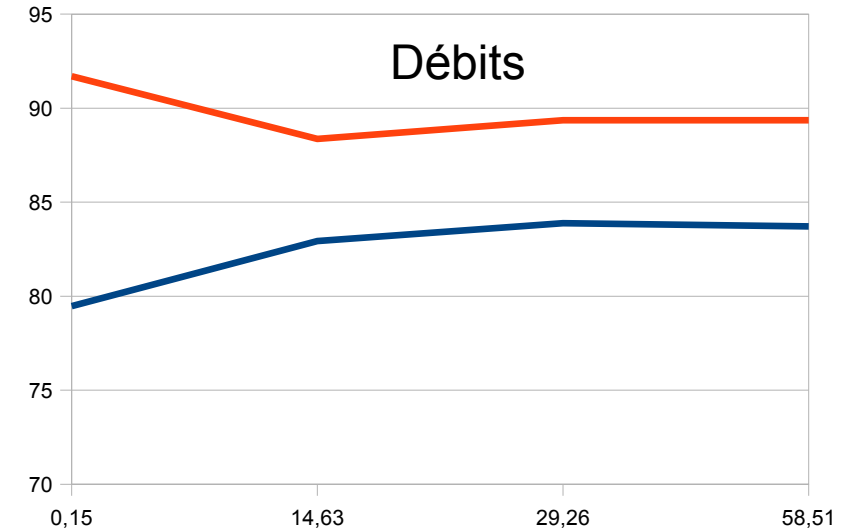
2ème test : La machine réelle reçoit les données avec **netcat** et les jette au fur et à mesure (pas de d'accès disque)

Temps de transfert



- Transfert Machine virtuelle vers une machine réelle du réseau
- Transfert Machine hôte vers la même machine réelle du réseau

Mb/s



Conclusions

- Mettre le routeur master en machine virtuelle en exploitation
 - Pas acceptable
 - Tout de même une dégradation de performances
 - Environnement nécessaire plus compliqué
 - Volonté de pouvoir booter la machine en réel dans un cadre simple
 - Sauf pour des cas provisoires ou d'urgences

- Mettre le routeur de backup en machine virtuelle
 - Très acceptable

- Mettre les 2 serveurs VPN en machine virtuelle
 - Très envisageable moyennant la vérification des performances

Globalement on ne gagne pas beaucoup en nombre de machines réelles et il y a des conséquences sur la simplicité de mise en oeuvre des fonctions vitales du réseau.

Cependant, quand on dispose de cet environnement réseau virtuel on peut ajouter d'autres machines placées sur différents VLAN (dns/radius/mail/DMZ....)

Conclusions

Il y a deux intérêts majeur dans la virtualisation de réseau

■ Le réseau virtuel peut complètement prendre le relai du réseau réel dans une seule machine dans certains cas : (réseau portable)

- + Sinistre majeur ou localisé
- + Maintenance dans la salle machine nécessitant l'arrêt de machines/clim....
- +

■ Capacité de test à l'identique avant mise en production
Associé avec le boot sur CD , la machine virtuelle testée est exactement équivalente à la machine réelle de production.

Références

KVM : <http://www.linux-kvm.org>

VDE : <http://vde.sourceforge.net>
<http://wiki.virtualsquare.org>

VRRP: http://fr.wikipedia.org/wiki/Virtual_Router_Redundancy_Protocol
<http://tools.ietf.org/html/rfc3768>
<http://www.keepalived.org/>

OPENVPN: <http://openvpn.net/>
<http://beta.openvpn.net/>