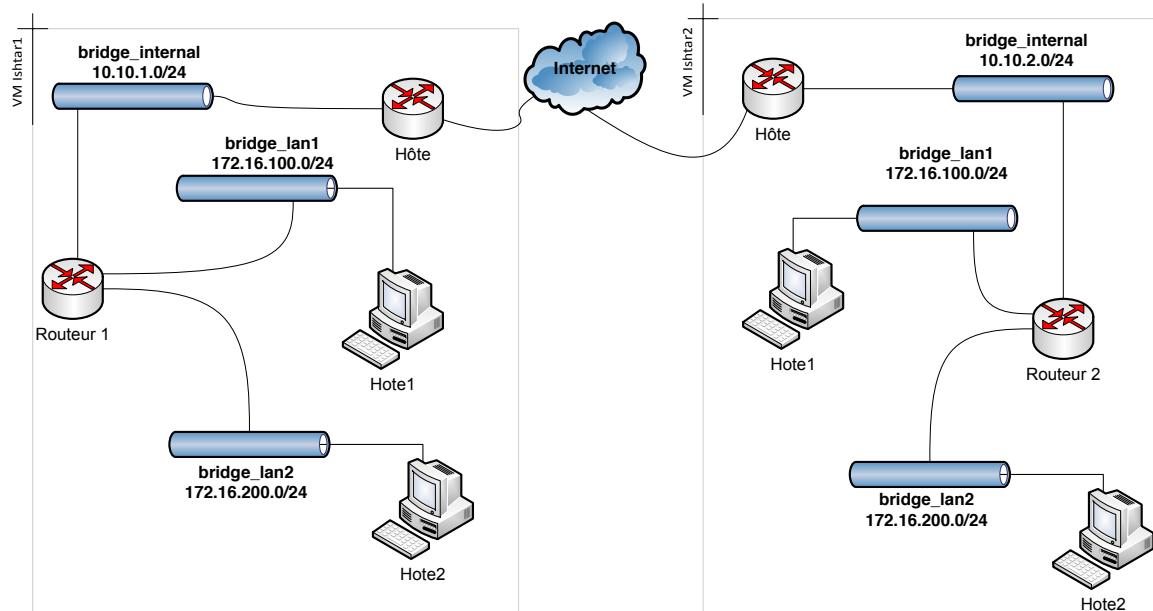


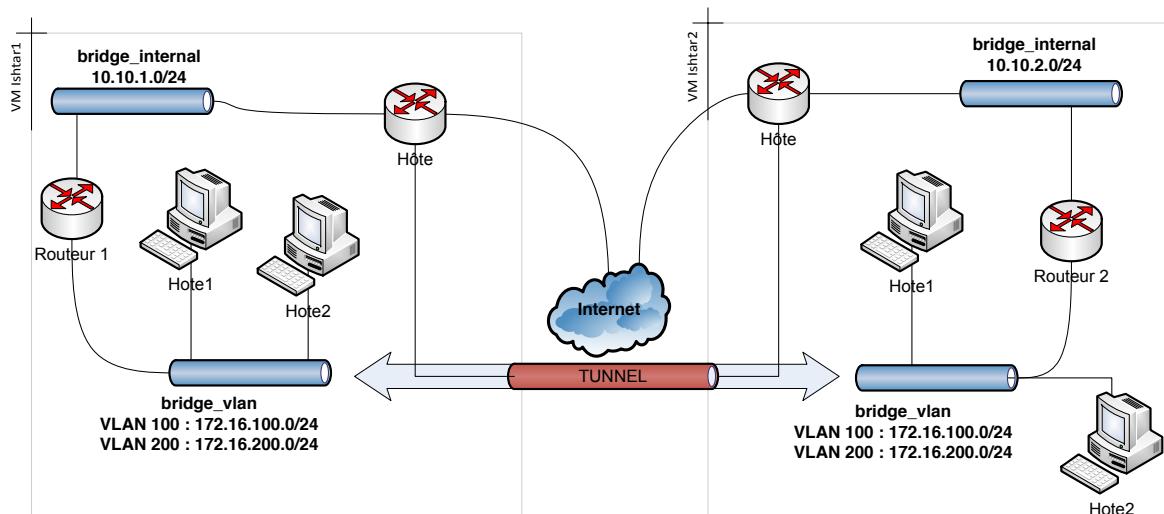
**Tunnel L2TPv3 sécurisé par IPsec pour la mise en œuvre de «trunking» de VLANs****Présentation du projet**

Le projet a pour but d'étudier le protocole L2TPv3, RFC 3931, pour faire des tunnels de niveau 2. On s'en servira pour faire du «trunking» de VLANs et de la mutualisation de service comme le DHCP. Enfin, à l'aide d'une configuration «intelligente», on limitera l'utilisation du tunnel lorsque les hôtes voudront se connecter à Internet en leur permettant de le faire directement de leur «côté d'Internet».

On dispose du réseau suivant où les réseaux sont disjoints au niveau 2 :



On voudrait arriver au schéma suivant où les réseaux sont liés au niveau 2 :



Un serveur DHCP sera mis en œuvre sur «Routeur1» afin de configurer les différents hôtes des deux VLANs (qu'ils soient ou non séparés par Internet).



Description de la mise en œuvre avec deux VMs, « *Virtual Machine* », et les containers LXC :

- \* deux VMs, « Ishtar1 » et « Ishtar2 » ;
- \* dans la VM « Ishtar1 » :
  - ◊ deux réseaux locaux : « bridge\_vlan » et « bridge\_internal » ;
  - ◊ deux containers « Hotel1 » et « Hotel2 » ;
  - ◊ un container « Routeur1 » chargé de faire le lien entre les trois réseaux, il est connecté à l'hôte (la VM) par l'intermédiaire du réseau « bridge\_internal » ;
- \* dans la VM « Ishtar2 » : la même configuration, mais avec le réseau « bridge\_internal » recevant une adresse différente (on pourra économiser le container « Hotel1 »).

## ■■■ Personnalisation de la VM

Dans le cadre de ce projet, vous pouvez utiliser deux ordinateurs physiques faisant chacun tourner une VM, ou bien un seul ordinateur faisant tourner les deux VMs. Dans les deux cas vous pouvez vouloir changer le nom de la VM pour faciliter leur exploitation simultanée.

Si vous voulez changer le nom de la VM « ishtar », vous devez modifier le contenu des fichiers « /etc/hostname » et « /etc/hosts » et remplacer « ishtar » par le nom que vous aurez choisi.

## ■■■ Linux, LXC & VLANs

Dans ce projet, nous allons utiliser des VLANs afin de :

- créer deux réseaux locaux **differents** regroupant des machines connectées au travers des mêmes éléments actifs du réseaux (bridge/switch) avec des adresses de niveau 3 différentes (adresse IP) ;
- isoler les trafics de niveau 2 de ces réseaux pour qu'ils n'interfèrent pas entre eux : ARP, DHCP, « *Neighbor Discovery Protocol* », etc.
- multiplexer le trafic des deux réseaux au sein d'un même switch/bridge à la manière du « *VLAN trunking* » :
  - ◊ la trame est « étiquetée », « *VLAN Tagging* », pour indiquer à quel VLAN elle appartient (suivant le format 802.1Q) ;
  - ◊ la mise en place de l'étiquette ou sa suppression est réalisée soit par :
    - \* l'élément actif du réseau (bridge/switch) et on parle de **VLAN de port** (sur un switch matériel le réglage est « untagged ») : la connexion ne permet d'atteindre qu'un seul VLAN et les trames ne contiennent pas d'étiquette à leur arrivée sur le port ;
    - \* la machine connectée au réseau et on parle de réglage « tagged » : cette configuration est utile, par exemple, lorsque l'on connecte sur le même port un ordinateur et un téléphone IP, utilisant le protocole « *Voice Over IP* », dont le trafic n'ira pas dans le même VLAN et sera priorisé suivant de la QoS.

Nous choisirons de faire du VLAN de port, qui offre la sécurité maximale tout en gardant la flexibilité du VLAN :

- ▷ la connexion n'appartient qu'à un seul VLAN ;
- ▷ les trames sont étiquetées dans le bridge/switch pour permettre le « *trunking* » ;
- ▷ les trames VLANs peuvent être multiplexées sur un lien pouvant être un **tunnel de niveau 2** ;
- ▷ les VLANs peuvent être répartis entre des sites, en employant le réseau IP pour assurer cette interconnexion.

### Mise en place de VLAN de port sous Linux & LXC

- Ce que l'on veut faire :
  - ◊ ajouter l'étiquette VLAN sur les trames en sortie du container ;
  - ◊ intégrer ce trafic dans un bridge pour :
    - \* les échanges avec les autres containers appartenant au même VLAN et s'exécutant sur la même VM ;
    - \* le passage dans le tunnel de niveau 2 pour relayer le trafic au travers du réseau IP vers les containers s'exécutant sur une autre VM.
- Comment fonctionne les VLANs sous Linux :
  - ◊ le VLAN s'exprime par l'ajout d'étiquette : « *VLAN tagged* » sur les trames ;
  - ◊ le VLAN crée une nouvelle interface qui s'accroche à une interface existante :
    - \* lorsqu'une trame est envoyée sans étiquette depuis la nouvelle interface VLAN créée, elle apparaît avec une étiquette VLAN sur l'interface existante ;
    - \* lorsqu'une trame est reçue avec une étiquette VLAN sur l'interface existante, elle apparaît sur la nouvelle interface créée sans l'étiquette.

- ◊ il est possible d'accrocher plusieurs interfaces VLANs sur la même interface, **seulement** si elles appartiennent à des VLANs différents.
- Comment nous allons procéder :
  1. créer pour chaque container une interface de type « veth », « virtual ethernet », qui se présente comme deux interfaces réseaux reliées par un câble : toute trame se présentant sur l'interface d'une extrémité se retrouve sur l'autre extrémité ;
  2. accrocher le container en mode VLAN sur une extrémité, c-à-d que le trafic en sortie du container sera étiqueté (fonctionnement en VLAN de port) ;
  3. ajouter l'autre extrémité dans un bridge.
- En conclusion :
  - ▷ tous les containers pourront communiquer entre eux s'ils sont connectés au même bridge et appartiennent au même VLAN ;
  - ▷ le trafic étiqueté peut être « tunnelisé » au travers d'un tunnel de niveau 2.

## ■■■■ Configuration d'interface virtuelle sous Linux

Pour la création de veth :

```
└── xterm
$ sudo ip link add name veth_filtre type veth peer name veth_tunnel
```

## ■■■■ Configuration des VLANs sous Linux

Pour ajouter le VLAN n°100 sur l'interface eth0 :

- ▷ on choisira un nom pour l'interface, comme par exemple eth0.100 pour rester compatible avec les notations utilisées par la commande vconfig qui était utilisée auparavant ;
- ▷ on utilisera la commande « ip » :

```
└── xterm
pef@cerberus:~$ sudo ip link
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 1000
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
pef@cerberus:~$ sudo ip link add link eth0 name eth0.100 type vlan id 100 loose_binding on
pef@cerberus:~$ ip link
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 1000
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
4: eth0.100@eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
```

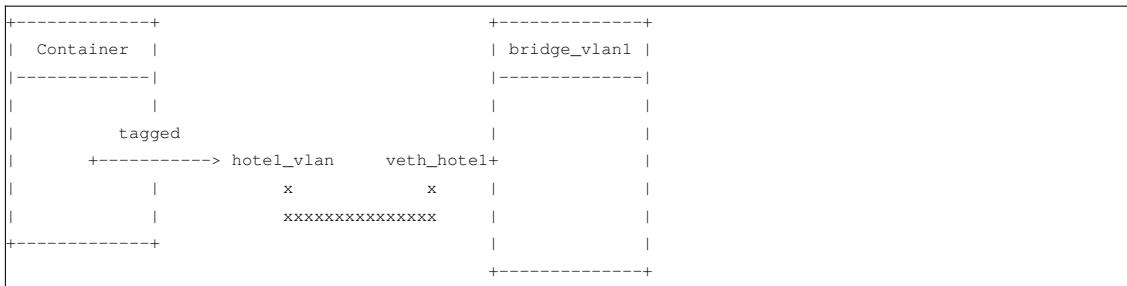
*Les trames qui passeront par cette interface possèderont un « tag » de VLAN avec l'id 100.*

*L'option « loose\_binding » permet de décorréler l'état « up/down » de l'interface VLAN de celle de l'interface à laquelle elle est accrochée.*

- ▷ on peut ensuite donner une adresse IP à cette nouvelle interface :

```
└── xterm
pef@cerberus:~$ sudo ip addr add 192.168.1.1/24 brd 192.168.1.255 dev eth0.100
pef@cerberus:~$ sudo ip link set dev eth0.100 up
pef@cerberus:~$ ip address
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN qlen 1000
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
        inet 192.168.127.213/24 brd 192.168.127.255 scope global eth0
            inet6 fe80::20c:29ff:fe0f:31a1/64 scope link
                valid_lft forever preferred_lft forever
4: eth0.100@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0.100
            inet6 fe80::20c:29ff:fe0f:31a1/64 scope link
                valid_lft forever preferred_lft forever
```

## ■■■■ Configuration d'un container LXC en VLAN de port



Pour l'ajout du lien de connexion container ↔ bridge :

```
└── xterm ━━━━━━
rezo@premiere:~$ sudo ip link add name hotel_vlan1 type veth peer name veth_hotel
rezo@premiere:~$ sudo ip link set dev hotel_vlan1 up
rezo@premiere:~$ sudo ip link set dev veth_hotel up
```

Pour la création du container avec étiquettement/désétiquettement du trafic en VLAN d'identifiant 100 :

```
└── xterm ━━━━━━
rezo@premiere:~$ more hotel.cfg
lxc.network.type = vlan
lxc.network.vlan.id = 100
lxc.network.flags = up
lxc.network.link = hotel_vlan1
lxc.network.name = eth0
```

Pour la connexion du container dans le bridge en mode « VLAN de port » :

```
└── xterm ━━━━━━
rezo@premiere:~$ sudo brctl addif bridge_vlan veth_hotel
```

Ce qui donne comme interfaces et configuration du bridge « bridge\_vlan » :

```
└── xterm ━━━━━━
rezo@premiere:~$ ip link
69: bridge_vlan: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1492 qdisc noqueue state UP mode DEFAULT
    link/ether 2a:e4:3b:4d:2b:70 brd ff:ff:ff:ff:ff:ff
70: veth_hotel: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bridge_vlan state UP
mode DEFAULT qlen 1000
    link/ether 92:db:d6:7b:c1:0f brd ff:ff:ff:ff:ff:ff
71: hotel_vlan1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT qlen 1000
    link/ether 52:e1:8b:66:4c:5b brd ff:ff:ff:ff:ff:ff
rezo@premiere:~$ brctl show
bridge name     bridge id          STP enabled      interfaces
bridge_vlan      8000.2ae43b4d2b70   no             veth_hotel
```

## ■■■■ Mise en place de tunnel L2TPv3

Pour configurer le tunnel l2TPv3, on a le choix entre une encapsulation dans IP ou bien dans UDP :

```
└── xterm ━━━━━━
pef@cerberus:~$ sudo ip l2tp add tunnel remote 192.168.127.161 local 192.168.127.213
    encap ip tunnel_id 3000 peer_tunnel_id 4000
pef@cerberus:~$ sudo ip l2tp show tunnel
Tunnel 3000, encap IP
  From 192.168.127.213 to 192.168.127.161
  Peer tunnel 4000
```

Il faut configurer l'autre poste de la même façon mais en symétrique (hellhound ↔ cerberus) :

```
└── xterm ━━━━━━
pef@hellhound:~$ sudo ip l2tp add tunnel local 192.168.127.161 remote 192.168.127.213
    encap ip tunnel_id 4000 peer_tunnel_id 3000
pef@hellhound:~$ sudo ip l2tp show tunnel
Tunnel 4000, encap IP
  From 192.168.127.161 to 192.168.127.213
  Peer tunnel 3000
```

On procéde de la même façon pour établir une « session » :

```
└── xterm ━━━━━━
pef@cerberus:~$ sudo ip l2tp add session tunnel_id 3000 session_id 1000 peer_session_id 2000
pef@cerberus:~$ ip link
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 1000
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
6: 12tpeth0: <BROADCAST,MULTICAST> mtu 1492 qdisc noop state DOWN mode DEFAULT qlen 1000
    link/ether be:f0:le:1e:e1:26 brd ff:ff:ff:ff:ff:ff
```

Et la version symétrique sur l'autre poste :

```
xfce4-terminal xterm
pef@hellhound:~$ sudo ip l2tp add session tunnel_id 4000 session_id 2000 peer_session_id 1000
pef@hellhound:~$ ip link
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 1000
    link/ether 00:0c:29:7f:2b:b6 brd ff:ff:ff:ff:ff:ff
4: l2tpeth0: <BROADCAST,MULTICAST> mtu 1492 qdisc noop state DOWN mode DEFAULT qlen 1000
    link/ether 4e:92:95:b4:81:74 brd ff:ff:ff:ff:ff:ff
```

On peut modifier, si nécessaire, la MTU de l'interface :

```
xfce4-terminal xterm
pef@cerberus:~$ sudo ip link set dev l2tpeth0 mtu 1500
pef@cerberus:~$ ip link
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 1000
    link/ether 00:0c:29:0f:31:a1 brd ff:ff:ff:ff:ff:ff
6: l2tpeth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT qlen 1000
    link/ether be:f0:le:le:e1:26 brd ff:ff:ff:ff:ff:ff
```

Et sur l'autre poste :

```
xfce4-terminal xterm
pef@hellhound:~$ sudo ip link set dev l2tpeth0 mtu 1500
pef@hellhound:~$ ip link
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 1000
    link/ether 00:0c:29:7f:2b:b6 brd ff:ff:ff:ff:ff:ff
4: l2tpeth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT qlen 1000
    link/ether 4e:92:95:b4:81:74 brd ff:ff:ff:ff:ff:ff
```

Pour ajouter la nouvelle interface dans «bridge\_vlan»

```
xfce4-terminal xterm
pef@hellhound:~$ sudo brctl addif bridge_vlan l2tpeth0
pef@hellhound:~$ sudo ip link set dev l2tpeth0 up
```

Maintenant, toute trame présente sur une extrémité du tunnel est relayée vers l'autre extrémité.

## Rappels et commandes utiles

Pour la configuration de dnsmasq :

```
xfce4-terminal xterm
$ sudo dnsmasq -d -C fichier_config
```

*Vous arrêterez le programme à l'aide de «Ctrl-C».*

Pour la destruction d'un container à partir de son nom :

```
xfce4-terminal xterm
rezo@cerberus:~$ sudo lxc-destroy -n HOST1
```

Pour l'obtention d'une configuration par DHCP :

```
xfce4-terminal xterm
$ sudo dhclient -d eth0
```

*Vous arrêterez le programme à l'aide de «Ctrl-C» une fois la configuration obtenue.*

Pour sniffer de manière détaillée le trafic entre deux hôtes :

```
xfce4-terminal xterm
$ sudo tcpdump -i eth0 -nvveX host 192.168.127.222 and host 192.168.127.228
```

## ■ ■ ■ ■ Travail à réaliser

*Vous rédigerez un rapport au format PDF contenant des explications, des commandes, des configurations, des captures d'écran et des analyses de trames.*

- 1 – Vous mettrez en œuvre le réseau conformément au schéma proposé et en vous servant des fichiers de configuration suivants pour réaliser la configuration complète :

Le fichier « config\_network\_hardware »

```
# on active le mode routeur
sudo sysctl -w net.ipv4.ip_forward=1

# on ajoute deux ponts
sudo brctl addbr bridge_internal
sudo brctl setfd bridge_internal 0
sudo ip link set bridge_internal up
sudo brctl addbr bridge_vlan
sudo brctl setfd bridge_vlan 0
sudo ip link set bridge_vlan up

# on cree les interfaces pour les VLANS des containers
sudo ip link add name hotel1_vlan1 type veth peer name veth_hotel1
sudo ip link set dev hotel1_vlan1 up
sudo ip link set dev veth_hotel1 up

sudo ip link add name hote2_vlan2 type veth peer name veth_hote2
sudo ip link set dev hote2_vlan2 up
sudo ip link set dev veth_hote2 up

sudo ip link add name routeur1_vlan1 type veth peer name veth_routeur11
sudo ip link set dev routeur1_vlan1 up
sudo ip link set dev veth_routeur11 up

sudo ip link add name routeur1_vlan2 type veth peer name veth_routeur12
sudo ip link set dev routeur1_vlan2 up
sudo ip link set dev veth_routeur12 up

# on ajoute les interfaces VLANs dans les bridges
sudo brctl addif bridge_vlan veth_hotel1
sudo brctl addif bridge_vlan veth_hote2
sudo brctl addif bridge_vlan veth_routeur11
sudo brctl addif bridge_vlan veth_routeur12
```

Le fichier « config\_network\_routing »

```
# on fixe les adresses des interfaces de l'hôte
sudo ip addr add 10.10.1.254/24 dev bridge_internal
```

Le fichier « routeur1.cfg »

```
# interface vers l'hôte
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = bridge_internal
lxc.network.name = eth0
lxc.network.ipv4 = 10.10.1.253/24
lxc.network.hwaddr = 00:FF:AA:00:00:01

# interface vers le VLAN 100
lxc.network.type = vlan
lxc.network.vlan.id = 100
lxc.network.flags = up
lxc.network.link = routeur1_vlan1
lxc.network.name = eth1
lxc.network.ipv4 = 172.16.100.254/24

# interface vers le VLAN 200
lxc.network.type = vlan
lxc.network.vlan.id = 200
lxc.network.flags = up
lxc.network.link = routeur1_vlan2
lxc.network.name = eth2
lxc.network.ipv4 = 172.16.200.254/24
```

Vous testerez le bon fonctionnement de l'encapsulation VLAN en « sniffant » le réseau à l'aide de « tcpdump ».

Vous incorporerez dans votre rapport une capture **commentée** de trafic VLAN entre un hôte et un routeur (qui sont les interlocuteurs, où le trafic est-il snifffé, à qui appartient les adresses MAC, quelle est la nature du trafic, son sens d'échange et le protocole observé).

- 2 – Vous rédigerez une description/présentation rapide du protocole L2TPv3 :

- a. en indiquant ses avantages ;
- b. en relevant les différences avec les versions précédentes du protocole ;
- c. en expliquant les notions d'identifiants et de sessions ;
- d. en commentant pourquoi on le compare à MPLS.

Le fichier « hote2.cfg »

```
lxc.network.type = vlan
lxc.network.vlan.id = 200
lxc.network.flags = up
lxc.network.link = hote2_vlan2
lxc.network.name = eth0
lxc.network.hwaddr = 00:FF:BB:00:02:02
```

Le fichier « dnsmasq\_config »

```
dhcp-range=172.16.100.10,172.16.100.100,168h
dhcp-range=172.16.200.10,172.16.200.100,168h
```

*Le choix de la bonne interface pour l'utilisation de la bonne configuration DHCP se fait automatiquement suivant l'adresse de l'interface.*

**3 – Vous établirez le tunnel L2TPv3 en mode encapsulation IP.**

Vous vérifierez que :

- a. le protocole ARP fonctionne normalement pour la découverte des machines quelle que soit leur côté de connexion par rapport à Internet (sur la VM 1 ou la VM 2) ;
- b. le service DHCP que vous n'exécuterez que sur « Routeur1 » peut être utilisé par l'hôte 2 de la deuxième VM ;
- c. une connexion TCP à l'aide de socat entre Routeur1 et Hôte2 est possible à travers le tunnel.

Pour chaque vérification vous ajouterez dans votre rapport des captures de trafic et de configuration d'interfaces permettant de justifier du bon fonctionnement de ces observations.

**4 – Vous comparerez :**

- a. la mise en œuvre de l2TPv3 :
  - ◊ en mode encapsulation IP ;
  - ◊ en mode encapsulation UDP.

Vous ferez une capture d'un paquet L2TPv3 dans chaque mode pour le protocole que vous vous voudrez (tcp, arp, icmp, etc.) et vous le détaillerez jusqu'à identifier précisément son contenu pour chacun de ses modes.

- b. Vous comparerez à un tunnel réalisé avec GRE en mode GRETAP (tunnel de niveau 2) :

```
└── xterm
$ sudo ip link add e0ip1 type gretap remote 192.168.127.161 local 192.168.127.213 nopmtudisc
$ sudo brctl addif bridge_internal e0ip1
$ sudo ip link set dev 12tpeth0 down
$ sudo ip link set dev e0ip1 up
```

*Pour passer d'un tunnel à l'autre, il suffit de mettre l'interface du tunnel à désactivé à « down » et l'autre à activer à « up ».*

Quelles différences ? La MTU est-elle la même ? etc.

**5 – En vous servant des fiches de TP, vous mettrez en place un chiffrement IPsec sur votre tunnel l2TPv3 en encapsulation IP.**

À l'aide de la commande « iperf » vous comparerez la vitesse de transfert entre « Routeur1 » et l'hôte 2 sur la VM n°2 avec et sans chiffrement.

Vous ajouterez dans votre rapport votre configuration, une capture commentée de trafic l2tpv3 avec protection IPSec et les rapports fournis par « iperf ».

**6 – Accès Internet « intelligent » :**

- a. vous ajouterez l'accès à Internet sur Routeur1 pour les postes des deux VLANs, à l'aide de règles « iptables » ;
  - b. vous vérifierez que le trafic de l'hôte2 de la VM n°2 passe bien par Routeur1 ;
  - c. vous regarderez comment en utilisant « dnsmasq », vous pouvez rediriger les postes de la VM 2, lors de leur configuration par DHCP, vers Routeur 2 au lieu de Routeur 1 pour optimiser l'accès à Internet.
- Vous ajouterez dans votre rapport la configuration de « dnsmasq » ainsi que des captures montrant que le trafic de Hôte2 vers Internet, passe bien par Routeur2 au lieu de Routeur1.

**7 – Vous interdirez le trafic entre VLAN 100 et VLAN 200 :**

- a. à l'aide de règles « iptables » ;
- b. à l'aide de la « Policy Routing ».

Vous ajouterez dans votre rapport les configurations utilisées.

**Remise du travail**

Le travail devra être déposé sur **Communities** sous forme d'une archive et du fichier PDF du rapport.

Cette archive contiendra :

- \* tous les fichiers de configuration utilisés qui permettent de déployer le réseau sur la VM n°1 et la VM n°2 ;