Introduction au routage inter-VLAN

Philippe Latu
philippe.latu(at)linux-france.org

http://www.linux-france.org/prj/inetdoc/

Historique des versions		
$Revision: 1436 $	$Date: 2009-11-18 23:24:46 +0100 (mer. 18 nov. 2009) $	$Author: latu $
Année universitaire 2009-2010		

Résumé

Ce document est une introduction au routage inter-VLAN sur les systèmes GNU/Linux. Configurer ce système de routage présente de nombreux intérêts tant du point de vue conception que du point de vue exploitation. Avec un système GNU/Linux on peut combiner les fonctions de cloisonnement des domaines de diffusion avec un bon niveau de sécurité basé le filtrage réseau netfilter/iptables. De plus, sur une infrastructure hétérogène associant plusieurs générations et|ou marques de commutateurs, GNU/Linux permet d'homogénéiser l'exploitation.

Table des matières

1. Copyright et Licence

1.1. Méta-information

Cet article est écrit avec *DocBook*[1] XML sur un système *Debian GNU/Linux*[2]. Il est disponible en version imprimable aux formats PDF et Postscript : `routage.inter-vlan.pdf`[3]| `routage.inter-vlan.ps.gz`[4].

Les commandes utilisées dans ce document ne sont pas spécifiques à une version particulière des systèmes UNIX ou GNU/Linux. C'est la distribution *Debian GNU/Linux* qui est utilisée pour les tests présentés. Voici une liste des paquets contenant les commandes :

- vlan - user mode programs to enable VLANs on your ethernet devices
- iptables - administration tools for packet filtering and NAT
- net-tools - The NET-3 networking toolkit
- ifupdown - High level tools to configure network interfaces

1.2. Conventions typographiques

Tous les exemples d'exécution des commandes sont précédés d'une invite utilisateur ou *prompt* spécifique au niveau des droits utilisateurs nécessaires sur le système.

- Toute commande précédée de l'invite `$` ne nécessite aucun privilège particulier et peut être utilisée au niveau utilisateur simple.
- Toute commande précédée de l'invite # nécessite les privilèges du super-utilisateur.

2. Réseaux locaux virtuels : VLANs

2.1. Définitions

Généralement, un réseau local (LAN) est défini par un domaine de diffusion. Tous les hôtes d'un réseau local reçoivent les messages de diffusion émis par n'importe quel autre hôte de ce réseau. Par définition, un réseau local est délimité par des équipements fonctionnant au niveau 3 du modèle OSI : la couche réseau.

Un réseau local virtuel (VLAN) est un réseau local (LAN) distribué sur des équipements fonctionnant au niveau 2 du modèle OSI : la couche liaison. À priori, il n'est donc plus nécessaire d'avoir recours à un équipement de niveau 3 pour «borner» le réseau local. Cette formulation bute sur une fonction difficile à contourner : l'interconnexion des réseaux

[1] http://www.docbook.org
[2] http://www.debian.org
[3] http://www.linux-france.org/prj/inetdoc/telechargement/routage.inter-vlan.pdf
[4] http://www.linux-france.org/prj/inetdoc/telechargement/routage.inter-vlan.ps.gz

locaux. Dès que l'on a besoin de communiquer entre domaines de diffusion, il est absolument nécessaire de passer par les fonctions de routage du niveau réseau du modèle OSI.

Le réseau local est distribué sur différents équipements via des liaisons dédiées appelées *trunks*. Un *trunk* est une connexion physique unique sur laquelle on transmet le trafic de plusieurs réseaux virtuels. Les trames qui traversent le *trunk* sont complétées avec un identificateur de réseau local virtuel (VLAN id). Grâce à cette identification, les trames sont conservées dans un même VLAN (ou domaine de diffusion).

Les *trunks* peuvent être utilisés :

entre deux commutateurs
: C'est le mode de distribution des réseaux locaux le plus courant.

entre un commutateur et un hôte
: C'est le mode de fonctionnement à surveiller étroitement. Un hôte qui supporte le *trunking* a la possibilité d'analyser le trafic de tous les réseaux locaux virtuels.

entre un commutateur et un routeur
: C'est le mode fonctionnement qui permet d'accéder aux fonctions de routage ; donc à l'interconnexion des réseaux virtuels par routage inter-VLAN. La mise en œuvre de ce type de routage est l'objet de ce document.

Enfin, il ne faut pas oublier que tous les VLANs véhiculés dans le même *trunk* partagent la bande passante du média utilisé. Si un *trunk* utilise un lien 100Mbps Full-Duplex, la bande passante de tous les VLANs associés est limitée à ces 100Mbps Full-Duplex.

2.2. Réseaux locaux virtuels standards

Il existe plusieurs mécanismes de gestion des VLANs. Beaucoup sont propriétaires et ne fonctionnent que sur les équipements d'une seule marque.

VLANs par ports
: Cette technique fournit une méthode de division d'un équipement de niveau 2 (un commutateur) en plusieurs domaines de diffusion. La réalisation de cette division est spécifique à chaque plateforme.

  Le coût d'administration de ce genre de réseaux locaux est très important puisqu'il faut gérer manuellement sur chaque équipement la distribution des réseaux locaux.

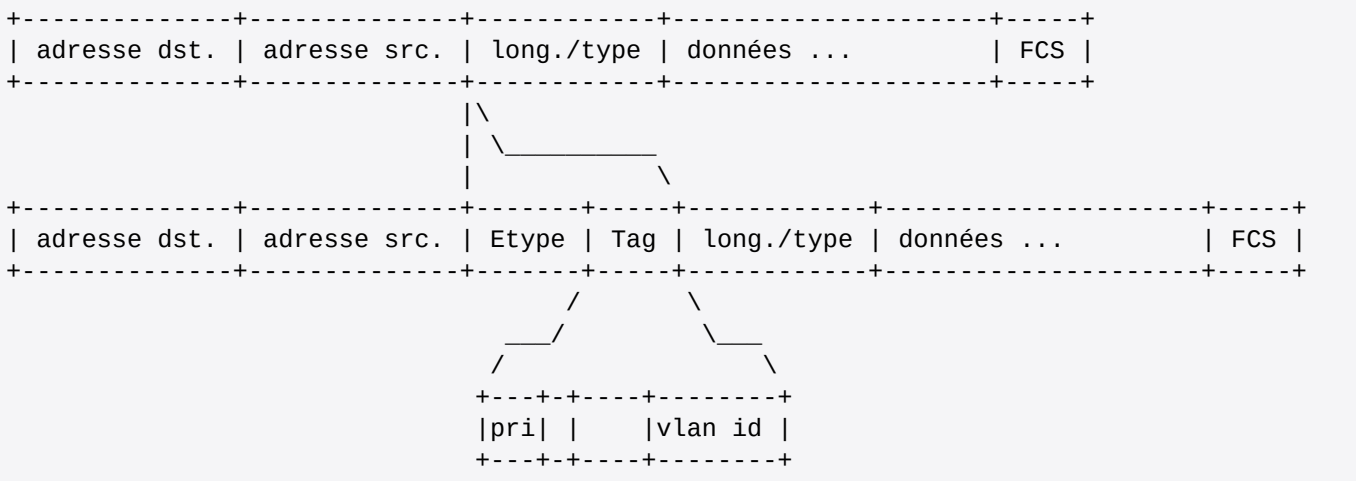
  Ceci dit, cette technique ne dépend pas d'une gestion propriétaire de l'affectation des ports dans les différents VLANs. C'est la raison principale pour laquelle elle est très répandue. Le commutateur assure une isolation complète entre la station et le VLAN auquel elle appartient.

VLANs du type Cisco Inter-Switch Link, ISL VLANs
: Cette technique a été développée spécifiquement pour les équipements Cisco™. Elle complète les en-têtes de trames avec 30 octets répartis en 13 champs. Ce type d'encapsulation n'est plus beaucoup utilisé du fait de son incompatibilité avec le standard IEEE 802.1Q.

VLANs IEEE 802.1Q
: Le standard IEEE 802.1Q fournit un mécanisme d'encapsulation très répandu et implanté dans de nombreux équipements de marques différentes. C'est sur ce standard que s'appuie ce document. Comme dans le cas de l'encapsulation ISL précédente, l'en-tête de trame est complété par une balise de 4 octets.

2.3. Balise IEEE 802.1Q

Le standard IEEE 802.1Q définit le contenu de la balise de VLAN (*VLAN tag*) avec laquelle on complète l'en-tête de trame Ethernet. Le format de la trame Ethernet modifiée avec les 4 octets supplémentaires est présenté ci-dessous :



Il faut noter que le champ `FCS` est recalculé après l'insertion de la balise de VLAN.

Voici un extrait de capture, réalisée avec Wireshark, qui illustre les champs de la balise IEEE 802.1Q.

Note

Pour ne capturer que les trames avec balise IEEE 802.1Q, voici la syntaxe de filtrage à priori : **`# tshark -i eth0 -w sample.cap vlan`**.

```
Frame 103 (1518 bytes on wire, 1518 bytes captured)
Ethernet II, Src: 00:14:f2:75:ed:72, Dst: 00:10:5a:de:9d:d7
    Destination: 3com_de:9d:d7 (00:10:5a:de:9d:d7)
    Source: Cisco_75:ed:72 (00:14:f2:75:ed:72)
    Type: 802.1Q Virtual LAN (0x8100) ❶
802.1Q Virtual LAN
    000. .... .... .... = Priority: 0 ❷
    ...0 .... .... .... = CFI: 0      ❸
    .... 0000 0110 0100 = ID: 100     ❹
    Type: IP (0x0800)
Internet Protocol, Src: 172.17.0.2 (172.17.0.2), Dst: 172.16.80.19 (172.16.80.19)
Transmission Control Protocol, Src Port: www (80), Dst Port: 1548 (1548)
```

❶ *Tag protocol identifier*, TPID, EtherType
: Ce champ de 16 bits identifie le protocole véhiculé dans la trame. La valeur `0x8100` désigne une balise IEEE 802.1Q / 802.1P.

❷ *Priority*
: Ce champ de 3 bits fait référence au standard IEEE 802.1P. Sur 3 bits on peut coder 8 niveaux de priorités de 0 à 7. La notion de priorité dans les VLANs est sans rapport avec les mécanismes de priorité IP au niveau réseau. Ces 8 niveaux sont utilisés pour fixer une priorité aux trames d'un VLAN relativement aux autres VLANs.

❸ *Canonical Format Identifier*
: Ce champ codé sur 1 bit assure la compatibilité entre les adresses MAC Ethernet et Token Ring. Un commutateur Ethernet fixera toujours cette valeur à 0. Si un port Ethernet reçoit une valeur 1 pour ce champ, alors la trame ne sera pas propagée puisqu'elle est destinée à un port «sans balise» (*untagged port*).

❹ *VLAN Identifier*, vlan id, VID
: Ce champ de 12 bits sert à identifier le réseau local virtuel auquel appartient la trame. Il est possible de coder 4094 (`2^12-2`) réseaux virtuels (VLANs) avec ce champ.

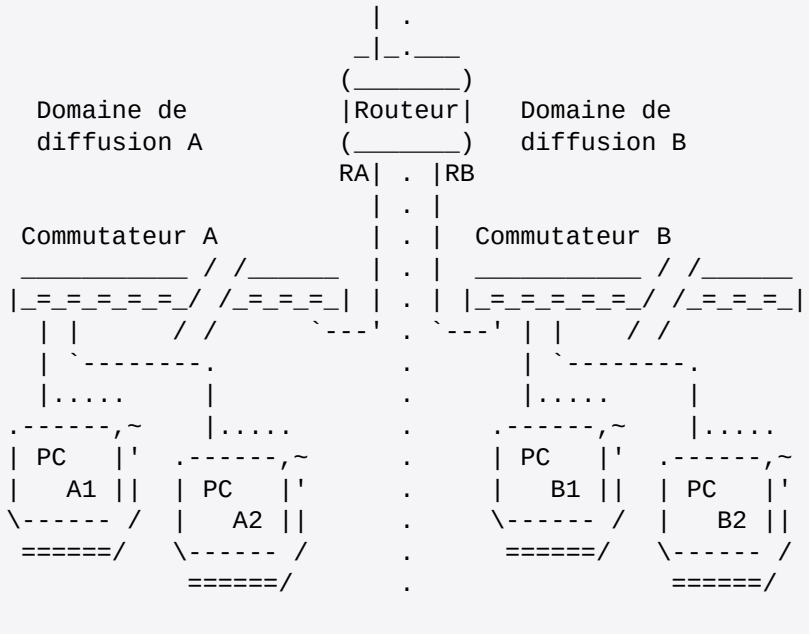
3. Routage inter-VLAN

A partir de l'argumentation développée dans l'article *La segmentation des réseaux locaux*, on dispose de deux règles de base. Sans aucune programmation particulière des équipements :

- Une interface de *commutateur* délimite un domaine de *collision*.
- Une interface de *routeur* délimite un domaine de *diffusion*.

3.1. Situation avant routage inter-VLAN

Du point de vue conception, le respect de ces deux règles impose que l'on ajoute une interface de routeur pour chaque nouveau domaine de diffusion ou périmètre de contrôle. De plus, les commutateurs appartenant à un domaine de diffusion sont dédiés à ce domaine. Il n'est pas possible de distribuer plusieurs réseaux locaux virtuels entre plusieurs domaines de diffusion «isolés» par un routeur.



Remarques sur ce type de conception :

- Si on programme le commutateur A avec 2 VLANs distincts pour chacun des PCs `A1` et `A2`, alors toute communication entre `A1` et `A2` sera impossible. De plus, ces deux PCs ne pourront communiquer avec d'autres réseaux que si l'interface du routeur `RA` appartient aux deux VLANs programmés.

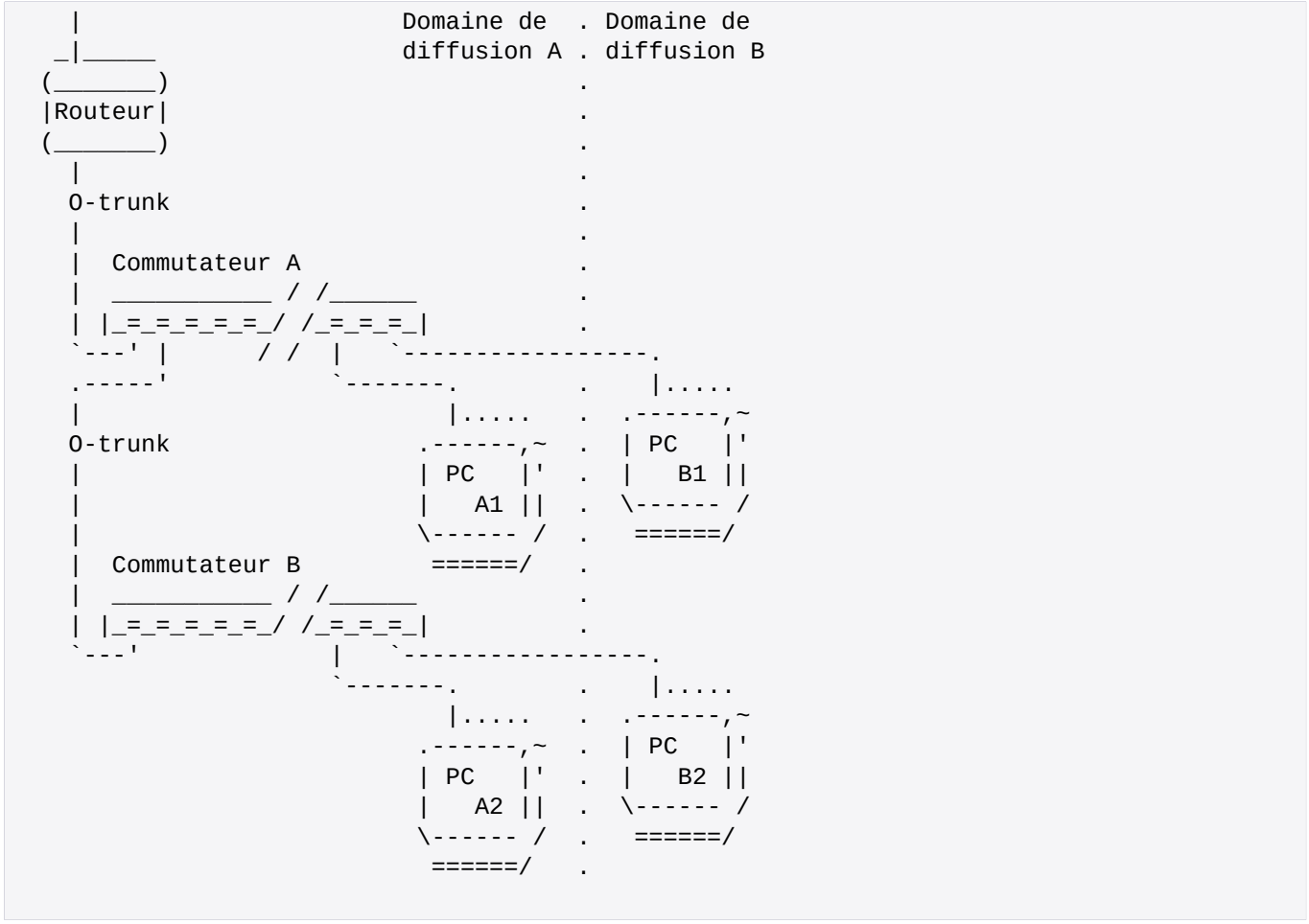
  Cette situation peut présenter des avantages du point de vue exploitation mais elle dépend beaucoup de la gestion des interfaces physiques. Ce que ne montre pas le diagramme simplifié ci-dessus, c'est que le coût d'administration devient très important dès que le nombre de réseaux virtuels augmente.
- Si l'utilisateur «associé» au PC `A1` déménage dans un lieu ou seul le domaine de diffusion B est distribué, il est nécessaire d'étendre le domaine de diffusion A jusqu'à ce nouveau lieu. En conséquence, il faudra installer un nouveau commutateur et câbler de nouvelles prises entre le point de brassage principal du domaine A et ce lieu.

  Sur une même infrastructure, on se retrouve rapidement avec des commutateurs saturés pour lesquels tous les ports disponibles sont utilisés et d'autres commutateurs pour lesquels seuls quelques ports sont utilisés.

  Ce scénario montre qu'il est excessivement difficile d'optimiser le parc des ports de commutateurs avec ce type d'architecture. Le coût de l'infrastructure augmente donc fortement puisqu'il faut passer par des réinvestissements lourds en câblage et en équipements à chaque modification des périmètres.

3.2. Situation après routage inter-VLAN

Les deux règles de base énoncées ci-dessus ne sont pas remises en question. Il s'agit maintenant de dissocier les notions d'interface physique et d'interface de routage. On n'associe plus une interface physique à chaque domaine de diffusion mais une interface «virtuelle» (encore du virtuel !).



Remarques sur ce type de conception relativement à la situation sans routage inter-VLAN :

- Le contrôle d'accès est centralisé au niveau du routeur. Il n'existe plus de «mélange des genres» entre la programmation des commutateurs et le contrôle d'accès au niveau réseau. Les communications entre les hôtes d'un même domaine de diffusion ou entre plusieurs domaines de diffusion sont gérées de la même façon.

  On obtient donc de véritables réseaux locaux distribués sur la totalité de l'infrastructure (équipements de niveau 2 + équipements de niveau 3).
- La gestion du parc des ports de commutation est optimisée. Comme les domaines de diffusion sont partagés entre tous les équipements, la gestion des évolutions est beaucoup plus souple. Les déménagements n'entraînent aucun recâblage tant que le nombre d'utilisateurs ne change pas. Il est donc possible de *concentrer* l'administration sur un nombre d'équipements plus faible que dans une architecture sans routage inter-VLAN.

3.3. Augmentation des débits

Si les avantages liés aux facilités d'exploitation ne suffisaient pas, l'augmentation des débits favorise un peu plus l'adoption d'architectures à base de routage inter-VLAN. Les évolutions techniques des routeurs conduisent à diminuer le nombre de leurs interfaces alors que les évolutions des commutateurs conduisent à augmenter considérablement le nombre de leurs ports.

En reprenant la remarque sur le partage des débits entre les VLANs à l'intérieur des *trunks* (voir Section 2.1, « Définitions »), il devient intéressant de partager le débit disponible «en fond de panier» dans les châssis de commutateurs. Le critère de choix d'un équipement, commutateur ou routeur, s'articule de plus en plus autour du rapport :

```
capacité de commutation en millions de paquets par seconde (mpps)
-----------------------------------------------------------------
                            prix
```

Ce que l'on appelle commutateur de niveau 3 correspond le plus souvent à un module de routage matériel ou logiciel inséré dans un commutateur classique de niveau 2. Si on reprend le diagramme ci-dessus avec ce genre d'équipement,

il ne faudrait plus qu'un seul châssis en lieu et place du routeur et des deux commutateurs. La répartition des rôles resterait inchangée.

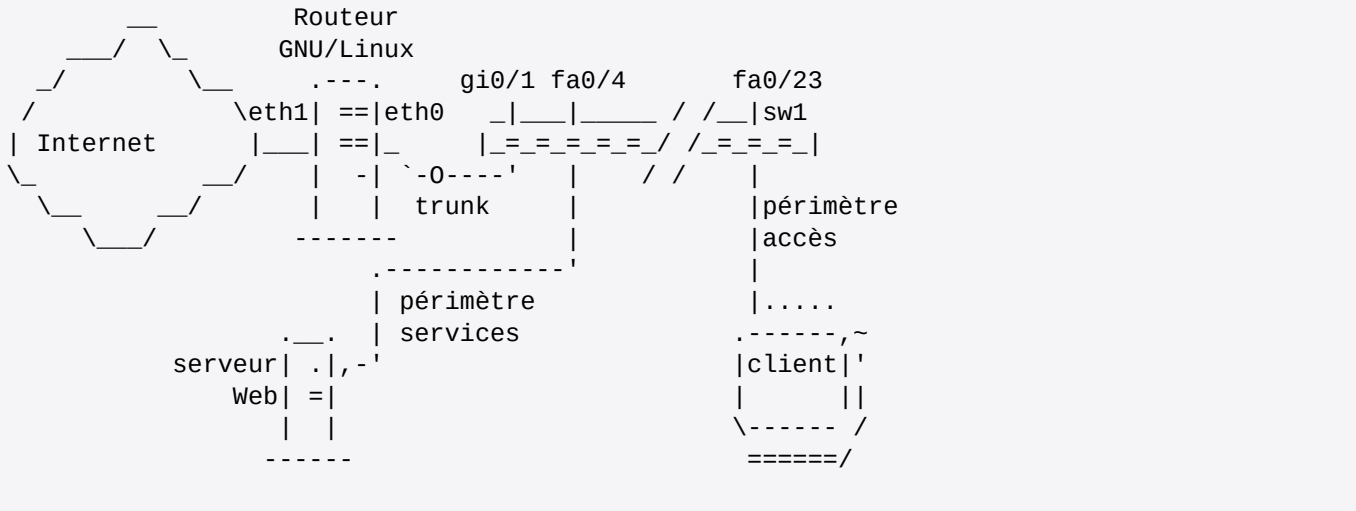
4. Etude d'une configuration type

La configuration type étudiée ici correspond à un scénario classique d'exploitation d'une infrastructure de taille moyenne. Pour les besoin des l'illustration, on n'utilise qu'un équipement de chaque type : un routeur et un commutateur.

Le routeur unique correspond bien à la réalité des réseaux modernes. Du réseau d'agence d'une centaine d'hôtes au réseau de campus de plusieurs milliers d'hôtes, seule la capacité de traitement de l'équipement varie (voir Section 3.3, « Augmentation des débits »).

Le commutateur unique correspond beaucoup moins à la réalité. Même dans un réseau d'agence, on dépasse très vite le cap des 48 ports connectés. On utilise alors un équipement avec une bonne capacité de commutation qui assure la *distribution* vers des commutateurs dédiés aux *accès* des hôtes. Tout ces commutateurs sont reliés entre eux à l'aide de *trunks* qui véhiculent les flux marqués des réseaux virtuels. Voir Section 2.1, « Définitions ».

Dans l'illustration présentée ici, les deux couches *distribution* et *accès* sont «synthétisées» sur un seul équipement. Un *trunk* sur un lien gigabit relie le routeur au commutateur. En véhiculant les flux marqués entre le routeur et le commutateur il assure la liaison entre routage et commutation. Les hôtes directement connectés au commutateur n'ont aucune connaissance des trames IEEE 802.1Q (voir Section 2.3, « Balise IEEE 802.1Q »). Ils ne nécessitent donc aucune configuration particulière.



Cette infrastructure type comprend 2 périmètres (ou réseaux privés) reliés au réseau public Internet. Un périmètre de services utilisé pour l'hébergement des services accessibles depuis le réseau public : DNS, Web, courrier électronique, etc. et un périmètre pour les postes de travail qui doivent être inaccessibles depuis le réseau public.

On ajoute aux deux périmètres classiques, un réseau particulier dédié à la gestion de l'infrastructure : configuration des équipements, métrologie, journalisation, etc.

Tableau 1. Plan d'adressage des périmètres

Nom	n° VLAN	Adresse IP
Management	2	`192.168.2.0/24`
Services	100	`192.168.100.0/24`
Accès	200	`192.168.200.0/24`

Le tableau ci-dessus établit la correspondance entre les périmètres, les réseaux virtuels et les réseaux IP à interconnecter.

4.1. Configuration du *trunk*

Communications réseau dans le périmètre *Management*

Du point de vue configuration, ce réseau est très particulier. Il véhicule les trames sans balises IEEE 802.1Q entre le routeur et le commutateur. On associe à ce périmètre le VLAN *natif* du *trunk*.

Côté routeur GNU/Linux, on configure l'interface de façon classique puisqu'il s'agit de traiter des trames Ethernet «normales».

```
# ifconfig eth0 192.168.2.2 broadcast 192.168.2.255 netmask 255.255.255.0
```

Côté commutateur, on utilise la notion de VLAN «natif» pour configurer le *trunk*.

```
!
interface GigabitEthernet0/1
 switchport trunk native vlan 2
 switchport mode trunk
 no cdp enable

<snipped/>
!
interface Vlan2
 ip address 192.168.2.1 255.255.255.0
 no ip redirects
 no ip unreachables
 no ip proxy-arp
 no ip route-cache
```

La configuration du *trunk* est la suivante :

```
#sh int gi0/1 trunk

Port        Mode         Encapsulation  Status        Native vlan
Gi0/1       on           802.1q         trunking      2

Port      Vlans allowed on trunk
Gi0/1       1-4094

Port        Vlans allowed and active in management domain
Gi0/1       1-2,100,200

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/1       1-2,100,200
```

Les règles d'utilisation des trames sans balises IEEE 802.1Q sont les suivantes :

- Toute trame appartenant au VLAN natif du *trunk* est émise sans balise IEEE 802.1Q par le commutateur.
- Toute trame reçue sans balise IEEE 802.1Q par le commutateur appartient au VLAN natif.

On complétera la configuration du commutateur de façon à ce que toutes les opérations de gestion de l'équipement passent par ce VLAN natif.

À ce niveau, les tests de communication réseau sont d'une grande trivialité !

- Côté routeur :

```
RouterA:~$ ping -c 2 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=255 time=19.4 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=255 time=1.22 ms

--- 192.168.2.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.226/10.355/19.484/9.129 ms
```

- Côté commutateur :

```
Switch#ping 192.168.2.2
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

4.2. Configuration IEEE 802.1Q sur le Routeur GNU/Linux

Communications réseau dans les périmètres *Services* et *Accès*

Cette fois-ci, il est indispensable de traiter les flux marqués avec les balises IEEE 802.1Q.

Côté routeur GNU/Linux, il faut s'assurer que le noyau à la capacité à traiter les balises IEEE 802.1Q. Tous les noyaux de distributions tels que ceux fournis par *Debian* ont un module IEEE 802.1Q appelé `8021q`.

```
# find /lib/modules/`uname -r` -name 8021q
/lib/modules/2.6.12-rc5/kernel/net/8021q
# modprobe 8021q
```

Le résultat du chargement du module se retrouve dans les messages système :

```
# dmesg
<snipped/>
802.1Q VLAN Support v1.8 Ben Greear <greearb@candelatech.com>
All bugs added by David S. Miller <davem@redhat.com>
```

Avec un noyau récent, ce module est chargé automatiquement lors d'une opération de configuration sur un VLAN.

Une fois la partie *kernelspace* traitée, on passe logiquement à la partie *userspace* : les outils de configuration utilisateur avec le paquet *Debian* `vlan`.

```
# dpkg -l vlan
Souhait=inconnU/Installé/suppRimé/Purgé/H=à garder
| État=Non/Installé/fichier-Config/dépaqUeté/échec-conFig/H=semi-installé
|/ Err?=(aucune)/H=à garder/besoin Réinstallation/X=les deux (État,Err: maj=mauvais)
||/ Nom      Version    Description
+++-========-==========-==============================================================
ii  vlan     1.9-3      User mode programs to enable VLANs on your ethernet devices
```

Tous les outils sont en place ; il ne reste plus qu'à configurer les 2 VLANs correspondant aux 2 périmètres définis.

```
# vconfig add eth0 100
Added VLAN with VID == 100 to IF -:eth0:-
# vconfig add eth0 200
Added VLAN with VID == 200 to IF -:eth0:-
```

Ces 2 opérations ont ajouté 2 sous-interfaces à l'interface physique `eth0`. On visualise le résultat avec la commande «historique» **ifconfig** (et|ou) la commande **ip** :

```
# ifconfig -a
eth0      Lien encap:Ethernet  HWaddr 00:0C:6E:B7:09:46
          inet adr:192.168.2.2  Bcast:192.168.2.255  Masque:255.255.255.0
          adr inet6: fe80::20c:6eff:feb7:946/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:2456 (2.3 KiB)  TX bytes:3180 (3.1 KiB)
          Interruption:185

eth0.100  Lien encap:Ethernet  HWaddr 00:0C:6E:B7:09:46
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
```

```
          collisions:0 lg file transmission:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

eth0.200  Lien encap:Ethernet  HWaddr 00:0C:6E:B7:09:46
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

```
# ip addr ls
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0c:6e:b7:09:46 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.2/24 brd 192.168.2.255 scope global eth0
    inet6 fe80::20c:6eff:feb7:946/64 scope link
    valid_lft forever preferred_lft forever
7: eth0.100: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
    link/ether 00:0c:6e:b7:09:46 brd ff:ff:ff:ff:ff:ff
8: eth0.200: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
    link/ether 00:0c:6e:b7:09:46 brd ff:ff:ff:ff:ff:ff
```

Les 2 nouvelles sous-interfaces se configurent manuellement de façon classique.

```
# ifconfig eth0.100 192.168.100.1 broadcast 192.168.100.255 netmask 255.255.255.0
# ifconfig eth0.200 192.168.200.1 broadcast 192.168.200.255 netmask 255.255.255.0
```

Sur un système *Debian GNU/Linux*, il est possible de rendre cette configuration permanente en éditant le fichier `/etc/network/interfaces` comme suit :

```
<snipped/>
auto eth0
iface eth0 inet static
        address 192.168.2.2
        netmask 255.255.255.0
        broadcast 192.168.2.255

auto eth0.100
iface eth0.100 inet static
        address 192.168.100.1
        netmask 255.255.255.0
        broadcast 192.168.100.255

auto eth0.200
iface eth0.200 inet static
        address 192.168.200.1
        netmask 255.255.255.0
        broadcast 192.168.200.255
```

Une fois la configuration des interfaces en place, on obtient la table de routage suivante :

```
# route -n
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0.100❶
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0❷
aaa.bbb.ccc.0   0.0.0.0         255.255.255.0   U     0      0        0 eth1❸
192.168.200.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0.200❹
0.0.0.0         aaa.bbb.ccc.1   0.0.0.0         UG❺    0      0        0 eth1
```

```
# ip route ls
192.168.100.0/24 dev eth0.100  proto kernel  scope link  src 192.168.100.1
192.168.2.0/24 dev eth0  proto kernel  scope link  src 192.168.2.2
aaa.bbb.ccc.0/24 dev eth1  proto kernel  scope link  src aaa.bbb.ccc.7
192.168.200.0/24 dev eth0.200  proto kernel  scope link  src 192.168.200.1
```

```
default via aaa.bbb.ccc.1 dev eth1
```

❶ L'interface `eth0.100` est associée au VLAN numéro 100. Sa configuration réseau correspond au périmètre *Services*. Les trames de ce réseau qui circulent sur le *trunk* sont complétées avec une balise IEEE 802.1Q comprenant l'identificateur de VLAN 100.

❷ L'interface `eth0` sert de *trunk* entre le routeur et le commutateur. Sa configuration réseau correspond au périmètre *Management*. Le réseau auquel appartient l'interface utilise des trames *sans balises IEEE 802.1Q*. Dans le vocabulaire Cisco™, ce VLAN est qualifié de «natif».

❸ L'interface `eth1` est directement connectée au réseau «public». Elle n'a aucune connaissance du trafic issu des différents périmètres sans configuration spécifique.

❹ L'interface `eth0.200` est associée au VLAN numéro 200. Sa configuration réseau correspond au périmètre *Accès*. Les trames de ce réseau qui circulent sur le *trunk* sont complétées avec une balise IEEE 802.1Q comprenant l'identificateur de VLAN 200.

❺ L'interface `eth1` a la possibilité d'acheminer le trafic issu du Routeur GNU/Linux vers l'Internet via une passerelle par défaut.

Côté commutateur, il faut que la base de données des VLANs connus contienne les mêmes identificateurs que ceux affectés sur le Routeur GNU/Linux.

Le fichier de configuration du commutateur doit contenir les informations suivantes :

```
vlan 2
 name management
!
vlan 100
 name services
!
vlan 200
 name access
```

Ensuite, on affecte les ports du commutateurs aux différents VLANs ou périmètres.

```
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int range fastEthernet 0/1 - 12
Switch(config-if-range)#switchport access vlan 100
Switch(config-if-range)#exit
Switch(config)#int range fastEthernet 0/13 - 48
Switch(config-if-range)#switchport access vlan 200
Switch(config-if)#^Z
Switch#
07:10:45: %SYS-5-CONFIG_I: Configured from console by console
```

On visualise le résultat des affectations de ports en mode accès de la façon suivante.

```
Switch#sh vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active
2    management                       active
100  services                         active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                Fa0/9, Fa0/10, Fa0/11, Fa0/12
200  access                           active    Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                Fa0/25, Fa0/26, Fa0/27, Fa0/28
                                                Fa0/29, Fa0/30, Fa0/31, Fa0/32
                                                Fa0/33, Fa0/34, Fa0/35, Fa0/36
                                                Fa0/37, Fa0/38, Fa0/39, Fa0/40
                                                Fa0/41, Fa0/42, Fa0/43, Fa0/44
                                                Fa0/45, Fa0/46, Fa0/47, Fa0/48
```

4.3. Activation de la fonction routage

Avec la configuration actuelle, le Routeur GNU/Linux ne remplit pas sa fonction. Par exemple, les hôtes du périmètre *Accès* ne peuvent pas communiquer avec les serveurs du périmètre *Services*. Il est nécessaire d'activer la fonction routage au niveau du noyau Linux pour que les paquets IP puissent être transmis (ou routés) entre des réseaux différents.

La présentation des fonctions réseau d'une interface pilotée par le noyau Linux sort du cadre de ce document. Il faut consulter le support *Configuration d'une interface de réseau local* pour obtenir les informations nécessaires.

Voici une copie du fichier `/etc/sysctl.conf` comprenant l'ensemble des réglages appliqués au noyau Linux du Routeur de la configuration type. Pour appliquer ces paramètres, il suffit d'utiliser la commande **`sysctl -p`** et de validé la valeur de la «clé» `ip_forward`. Si cette valeur est à `1`, le routage est actif au niveau du noyau Linux.

```
# /etc/sysctl.conf - Configuration file for setting system variables
# See sysctl.conf (5) for information.
#
# Refuser la prise en charge des requêtes ARP pour d'autres hôtes
net.ipv4.conf.all.proxy_arp = 0

# Ignorer les mauvais messages d'erreurs ICMP
net.ipv4.icmp_ignore_bogus_error_responses = 1

# Ignorer les messages de diffusion ICMP
net.ipv4.icmp_echo_ignore_broadcasts = 1

# Journaliser les adresses sources falsifiées ou non routables
net.ipv4.conf.all.log_martians = 1

# Refuser les adresses sources falsifiées ou non routables
net.ipv4.conf.all.rp_filter = 1

# Refuser les messages ICMP redirect
net.ipv4.conf.all.accept_redirects = 0

net.ipv4.conf.all.send_redirects = 0

# Refuser le routage source
net.ipv4.conf.all.accept_source_route = 0

# Activer le routage
net.ipv4.ip_forward = 1
```

Les tests de communications entre les réseaux des différents périmètres peuvent être effectués depuis le commutateur.

```
Switch#ping 192.168.2.2❶

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
Switch#ping 192.168.100.1❷

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/202/1000 ms
Switch#ping 192.168.200.1❸

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.200.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

```
Switch#ping aaa.bbb.ccc.7❹

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to aaa.bbb.ccc.7, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/202/1004 ms
Switch#ping aaa.bbb.ccc.1❺

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to aaa.bbb.ccc.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

❶ Test de communication ICMP sur le périmètre *Management*. Ce test n'utilise pas la fonction routage puisqu'il est effectué entre les 2 extrémités du *trunk*.

❷ Test de communication ICMP sur le périmètre *Services*. Ce test utilise la fonction routage entre le réseau `192.168.2.0/24` et le réseau `192.168.100.0/24`.

❸ Test de communication ICMP sur le périmètre *Accès*. Ce test utilise la fonction routage entre le réseau `192.168.2.0/24` et le réseau `192.168.200.0/24`.

❹ Test de communication ICMP vers le réseau public. Ce test utilise la fonction routage entre le réseau `192.168.2.0/24` et le réseau `aaa.bbb.ccc.0/24`.

❺ Test de communication ICMP vers l'Internet. Ce test échoue puisque le Routeur GNU/Linux n'échange pas sa table de routage avec les autres routeurs de l'Internet.

Ces tests montrent qu'il faut compléter la configuration pour que les échanges réseau entre les périmètres et l'Internet soient possibles. Comme ces échanges réseau entre l'Internet et les périmètres ne peuvent pas se faire dans n'importe quelles conditions, il est nécessaire d'introduire la fonction de filtrage pour obtenir une interconnexion satisfaisante.

5. Interconnexion et filtrage réseau

L'étude du filtrage réseau avec le noyau Linux sort du cadre de ce document. Il faut consulter les versions françaises du *Guide Pratique du Filtrage de Paquets sous Linux 2.4* et du *Guide Pratique du NAT sous Linux 2.4* pour obtenir les informations nécessaires.

D'un point de vue général, on dispose de deux solutions distinctes pour interconnecter les périmètres réseau administrés avec l'Internet.

- Partager la table de routage des périmètres administrés avec les routeurs de l'Internet via un protocole de routage tel qu'OSPF. Consulter le guide *Initiation au routage, 3ème partie* pour obtenir des exemples complets d'exploitation du protocole de routage OSPF avec les services du logiciel GNU/Linux Quagga.
- Camoufler les périmètres administrés derrière une adresse IP publique accessible depuis l'Internet. Cette opération est réalisée avec les fonctions de filtrage réseau du noyau Linux : netfilter pour la partie *kernelspace* et iptables pour la partie *userspace*.

C'est la seconde proposition qui offre le plus de facilités de contrôle immédiat sur les flux réseau. L'outil de camouflage (*masquerading*) généralement utilisé est appelé traduction d'adresses (*Native Address Translation* ou NAT).

5.1. Fonctionnement minimal

Après avoir activé le routage au niveau noyau (voir Section 4.3, « Activation de la fonction routage »), la fontion de camouflage est simple à mettre en oeuvre :

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

Cette règle réalise une traduction d'adresse source. Tout paquet IP sortant par l'interface `eth1` voit son adresse IP source réécrite avec l'adresse IP de l'interface.

L'exécution de la règle entraîne le chargement des modules de gestion de la traduction d'adresses et du suivi dynamique de communication (*stateful inspection*).

```
# dmesg |grep ip_
ip_tables: (C) 2000-2002 Netfilter core team
ip_conntrack version 2.1 (4095 buckets, 32760 max) - 248 bytes per conntrack
```

```
# lsmod |grep ip
iptable_filter          3200  0
ipt_MASQUERADE          3712  1
iptable_nat            23516  2 ipt_MASQUERADE
ip_conntrack           44728  2 ipt_MASQUERADE,iptable_nat
ip_tables              22528  3 iptable_filter,ipt_MASQUERADE,iptable_nat
ipv6                  255936  12
```

Le suivi dynamique de communication consiste à conserver une empreinte de paquet sortant de façon à identifier les paquets retour relatifs à cette «demande». Les empreintes sont stockées dans la table `/proc/net/ip_conntrack` du système de fichiers virtuel du noyau Linux.

```
# cat /proc/net/ip_conntrack
tcp❶     6 431999 ESTABLISHED❷ src=192.168.200.2❸ dst=192.168.200.1❹ \
         sport=33450 dport=22❺ packets=417 bytes=31133 \
         src=192.168.200.1 dst=192.168.200.2 \
         sport=22 dport=33450 packets=306 bytes=111969 [ASSURED] mark=0 use=1
tcp      6 431999 ESTABLISHED src=192.168.200.2❻ dst=64.236.34.4❼ \
         sport=33449 dport=80❽ packets=7075 bytes=368009 \
         src=64.236.34.4 dst=aaa.bbb.ccc.7❾ \
         sport=80 dport=33449 packets=9219 bytes=12839148 [ASSURED] mark=0 use=1
```

❶ Protocole de transport utilisé.

❷ État de la connexion TCP.

❸❻ Adresse IP source. Cette adresse correspond à un poste client appartenant au périmètre *Accès*.

❹❼ Adresses IP destination. Dans le premier cas, la communication est interne au réseau du périmètre *Accès*. Dans le second cas, il s'agit d'une adresse sur l'Internet.

❺❽ À partir du port destination du paquet sortant on peut identifier le service Internet utilisé : SSH et HTTP. Plus loin sur la même ligne, on retrouve les adresses IP source et destination attendues.

❾ L'adresse IP destination attendue pour un paquet retour est l'adresse publique du Routeur GNU/Linux. Cette ligne montre bien que le routeur à la connaissance des réseaux internes et du réseau public. C'est à partir de ces correspondances d'adresses IP que les décision d'acheminement sont prises. Dans le cas de la traduction d'adresses par camouflage, l'adresse IP retour est réécrite avec l'adresse IP de l'hôte du périmètre *Accès*.

Si cette configuration a le mérite d'illustrer le fonctionnement du routage inter-VLAN de façon simple, elle ne correspond pas à un niveau de contrôle d'accès suffisant. L'objet de la section suivante est justement de chercher à augmenter ce niveau de contrôle.

5.2. Meilleur contrôle d'accès

Dans un premier temps, il faut garantir que tous les paquets IP non autorisés sont bloqués ; ce qui revient à appliquer la règle *«tout ce qui n'est pas autorisé est interdit»*.

La traduction de cette règle en termes de configuration revient à jeter tous les nouveaux paquets par défaut sur les «chaînes» d'entrée et de traversée des interfaces réseau

```
# iptables -P INPUT DROP
# iptables -P FORWARD DROP
```

En toute rigueur, il faudrait faire de même avec la chaîne de sortie `OUTPUT`. Cette présentation ayant pour but premier d'illustrer les concepts, ajouter les traitements de la chaîne `OUTPUT` ne ferait qu'alourdir les scripts sans apporter d'élément nouveau.

Dans un deuxième temps, il faut affiner la configuration du suivi de communication dynamique. La règle d'or du filtrage avec la fonction *stateful inspection*, c'est *la description la plus fine possible du premier paquet qu'on autorise à passer*.

La traduction de cette règle en termes de configuration contient 2 parties :

- Un bloc de règles qui organise le suivi de communication pour chaque chaîne sur laquelle on appliqué la politique par défaut `DROP`.

```
-A <CHAINE> -p udp -m state --state RELATED,ESTABLISHED -j ACCEPT
-A <CHAINE> -p tcp -m state --state ESTABLISHED -m tcp ! --syn -j ACCEPT
-A <CHAINE> -p tcp -m state --state RELATED -m tcp --syn -j ACCEPT
-A <CHAINE> -p icmp -m state --state ESTABLISHED -j ACCEPT
```

- Des règles spécifiques à chaque flux autorisé. C'est à la rédaction de ces règles qui correspondent au premier paquet autorisé qu'il faut apporter le plus grand soin. Un exemple pour les paquets IP émis depuis le périmètre *Accès* sur la chaîne `FORWARD` :

```
-A FORWARD -i eth0.200 -s 192.168.200.0/24 \
   -p tcp -m tcp --syn --sport 1024: -m state --state NEW -j ACCEPT
-A FORWARD -i eth0.200 -s 192.168.200.0/24 \
   -p udp -m udp --sport 1024: -m state --state NEW -j ACCEPT
-A FORWARD -i eth0.200 -s 192.168.200.0/24 \
   -p icmp --icmp-type echo-request -m state --state NEW -j ACCEPT
```

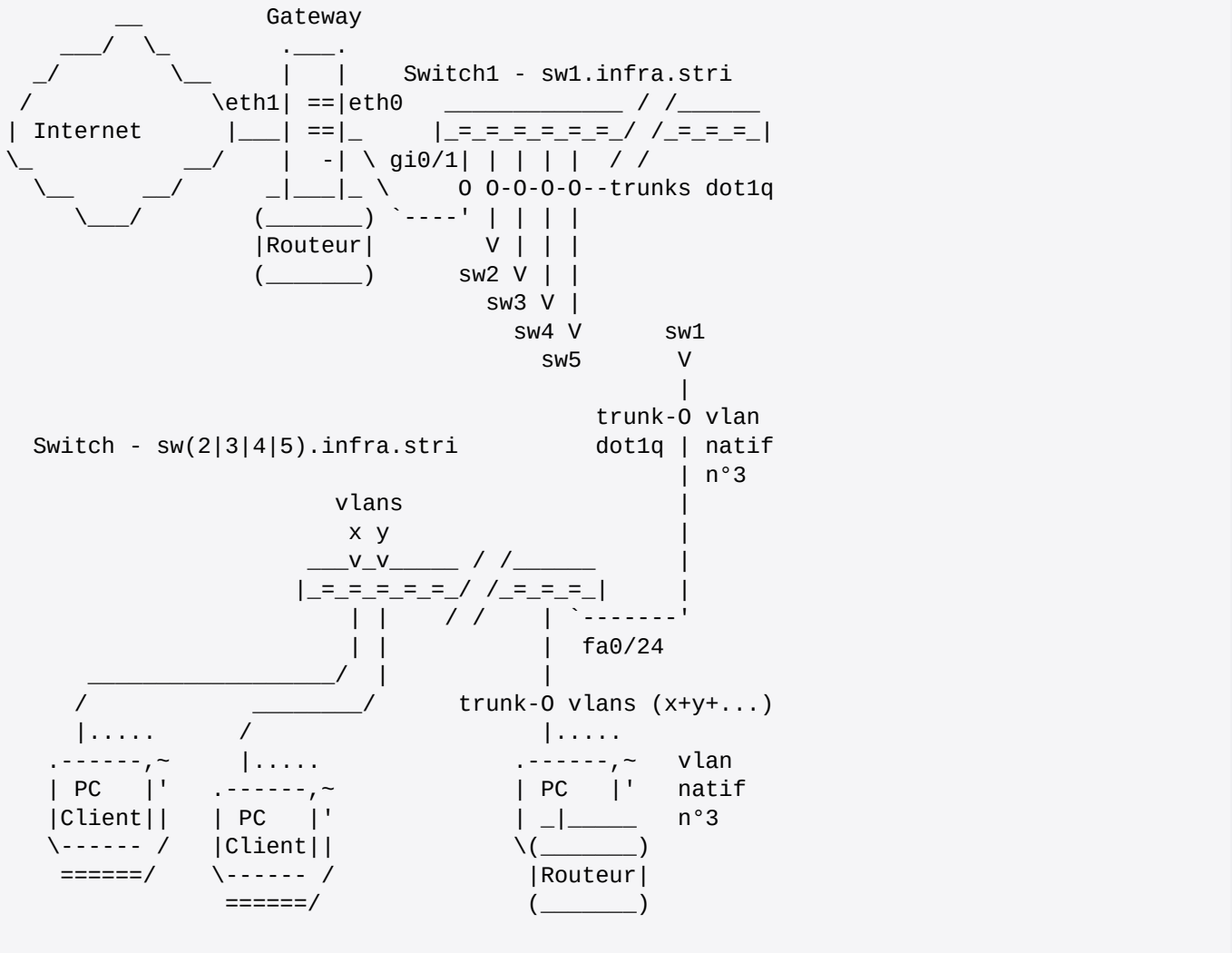
Voici une version intermédiaire de script de configuration du filtrage pour le périmètre *Accès*. En supposant que le fichier des règles est stocké dans le répertoire `/var/lib/iptables/`, on active les règles avec une commande du type **`iptables-restore </var/lib/iptables/active`**.

```
# Configuration type du filtrage réseau
#
# !!Attention!! Toutes les lignes coupées avec '\' doivent être réalignées
# avant de pouvoir utiliser ce script avec la commande iptables-restore
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Tables de traduction d'adresses
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A POSTROUTING -o eth1 -p tcp --tcp-flags SYN,RST SYN \
               -m tcpmss --mss 1400:1536 -j TCPMSS --clamp-mss-to-pmtu
-A POSTROUTING -o eth1 -j MASQUERADE
COMMIT
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Tables de filtrage
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
#
# -> Chaîne INPUT
# . suivi de communication
-A INPUT -p udp -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p tcp -m state --state ESTABLISHED -m tcp ! --syn -j ACCEPT
-A INPUT -p tcp -m state --state RELATED -m tcp --syn -j ACCEPT
-A INPUT -p icmp -m state --state ESTABLISHED -j ACCEPT
-A INPUT -p icmp --icmp-type destination-unreachable -m state --state RELATED -j ACCEPT
-A INPUT -p icmp --icmp-type time-exceeded -m state --state RELATED -j ACCEPT
# . toutes les communications internes sont autorisées
-A INPUT -i lo -m state --state NEW -j ACCEPT
-A INPUT -i eth0.200 -m state --state NEW -j ACCEPT
# . administration du Routeur GNU/Linux avec SSH
-A INPUT -i eth1 -p tcp -m tcp --dport 22 -m state --state NEW -j ACCEPT
# . services de gestion du commutateur vers le Routeur GNU/Linux
-A INPUT -i eth0 -s 192.168.2.1 -p udp \
         -m multiport --dports 69,123,162,514 -m state --state NEW -j ACCEPT
```

```
# . poubelle propre
-A INPUT -m state --state INVALID -j DROP
-A INPUT -p tcp -j REJECT --reject-with tcp-reset
-A INPUT -p udp -j REJECT --reject-with icmp-port-unreachable
#
# -> Chaîne FORWARD
# . suivi de communication
-A FORWARD -p udp -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -p tcp -m state --state ESTABLISHED -m tcp ! --syn -j ACCEPT
-A FORWARD -p tcp -m state --state RELATED -m tcp --syn -j ACCEPT
-A FORWARD -p icmp -m state --state ESTABLISHED -j ACCEPT
-A FORWARD -p icmp --icmp-type destination-unreachable \
           -m state --state RELATED -j ACCEPT
-A FORWARD -p icmp --icmp-type time-exceeded -m state --state RELATED -j ACCEPT
# . communications des hôtes du périmètre Accès
-A FORWARD -i eth0.200 -s 192.168.200.0/24 \
           -p tcp --syn --sport 1024: -m state --state NEW -j ACCEPT
-A FORWARD -i eth0.200 -s 192.168.200.0/24 \
           -p udp --sport 1024: -m state --state NEW -j ACCEPT
-A FORWARD -i eth0.200 -s 192.168.200.0/24 \
           -p icmp --icmp-type echo-request -m state --state NEW -j ACCEPT
# . poubelle propre
-A FORWARD -m state --state INVALID -j DROP
-A FORWARD -p tcp -j REJECT --reject-with tcp-reset
-A FORWARD -p udp -j REJECT --reject-with icmp-port-unreachable
COMMIT
```

6. Travaux pratiques

6.1. Topologie type de travaux pratiques



Pour les besoins de ces travaux pratiques, les configurations des 4 commutateurs : `sw5.infra.stri`, `sw6.infra.stri`, `sw7.infra.stri` et `sw8.infra.stri` sont effacées ainsi que leurs bases de données de VLANs.

Comme indiqué dans la topologie type ci-dessus, trois postes de travaux pratiques sont associés à un commutateur. Un poste joue le rôle de routeur inter-VLAN et les deux autres sont des postes clients appartenant chacun à un VLAN ou réseau IP différent.

Le seul point de configuration imposé est le raccordement au réseau d'interconnexion avec le routeur principal de la salle de travaux pratiques. Ce raccordement utilise le port `fa0/24` de chaque commutateur qui doit être configuré en mode *trunk* en utilisant le VLAN natif numéro 3. Le réseau IP correspondant au VLAN numéro 3 a l'adresse : `172.16.0.0/20`

Point important, la lecture de la section «Plan d'adressage» du document *Architecture réseau des travaux pratiques*[5] donne les adresses des deux routeurs ayant accès au réseau de Campus.

- Routeur `casper.infra.stri` : `172.16.0.2/20`
- Routeur `cooper.infra.stri` : `172.16.0.4/20`

[5] http://www.linux-france.org/prj/inetdoc/cours/archi.tp/

6.2. Affectation des postes de travail

Les affectations données dans la table ci-dessous ne sont pas figées pour la durée des travaux pratiques. Une fois la configuration validée sur un groupe de trois postes, il est vivement conseillé de permuter les rôles de façon à mieux maîtriser les étapes de configuration.

Tableau 2. Affectation des rôles, des numéros de VLANs et des adresses IP

Groupe	Commutateur	Poste	Rôle	VLAN	Réseau
1	sw5.infra.stri	alderaan	client	300	192.168.1.2/25
		bespin	client	301	192.168.1.130/25
		centares	routeur	3	172.16.0.30/20
				300	192.168.1.1/25
				301	192.168.1.129/25
2	sw6.infra.stri	coruscant	client	302	192.168.2.2/25
		dagobah	client	303	192.168.2.130/25
		endor	routeur	3	172.16.0.32/20
				302	192.168.2.1/25
				303	192.168.2.129/25
3	sw7.infra.stri	felucia	client	304	192.168.3.2/25
		geonosis	client	305	192.168.3.130/25
		hoth	routeur	3	172.16.0.34/20
				304	192.168.3.1/25
				305	192.168.3.129/25
4	sw8.infra.stri	mustafar	client	306	192.168.4.2/25
		naboo	client	307	192.168.4.130/25
		tatooine	routeur	3	172.16.0.36/20
				306	192.168.4.1/25
				307	192.168.4.129/25

Le positionnement des 4 commutateurs est référencé dans le support *Architecture réseau des travaux pratiques*.

6.3. Configuration des postes de travaux pratiques

1. Dans un groupe de trois postes tel qu'il a été défini ci-dessus, quel(s) poste(s) nécessite(nt) une configuration spécifique pour l'utilisation des réseaux locaux virtuels ?

2. Toujours dans un groupe de trois postes, comment doivent être programmés les ports de commutateur sur lesquels les postes clients sont raccordés ?

3. Encore dans un groupe de trois postes, comment doivent être programmés les ports de commutateur sur lesquels les routeurs sont raccordés ?

4. Dans la configuration d'un *trunk*, qu'est-ce qui distingue un VLAN natif ?

5. À partir du tableau des affectations ci-dessus, pourquoi les trois postes d'un groupe ne peuvent-ils pas appartenir au même réseau IP ?

6. Quel type de poste reçoit les trames complétées par des balises IEEE 802.1Q ?

Une fois le plan d'adressage IP défini, reprendre la Section 4, « Etude d'une configuration type » pour le groupe de postes de travaux pratiques.

1. Quel est le paquet qui contient les outils de configuration des interfaces réseau correspondant à chaque VLAN à router ?

2. Une fois les interfaces de chaque VLAN configurées sur le poste routeur, quelles sont les opérations à effectuer pour que le transfert des paquets IP d'un réseau local à l'autre soit effectif ?

3. Pourquoi doit-on utiliser la traduction d'adresses pour les flux réseau sortants du poste routeur vers l'Internet ? Que deviennent les paquets IP de ces flux sans traduction d'adresses ? Si la traduction d'adresses n'était pas disponible, quelle autre technique faudrait-il employer ?

4. Donner la séquence des tests ICMP à effectuer pour valider la connectivité entre :

   - les postes clients et le poste routeur,
   - les postes clients et l'ensemble des autres interfaces du routeur,
   - les postes clients entre eux,
   - les postes clients et l'Internet.

5. À l'aide de l'analyseur Wireshark, capturer des flux réseau mettant en évidence le marquage des trames avec les balises IEEE 802.1Q. Relever les numéros d'identification des VLANs vus par les interfaces du routeur. Quelle interface faut-il utiliser pour la capture de façon à visualiser l'ensemble du trafic ?

6. Pourquoi les flux réseau capturés contiennent-ils autant de trames STP (*Spanning Tree Protocol*) ?

7. Pourquoi la majorité des trames STP capturées sont-elles considérées comme ayant le type *Ethernet II* ? Quel aurait du être le type d'une trame STP si les balises IEEE 802.1Q n'étaient pas utilisées ?

7. Documents de référence

IEEE 802.1Q Standard
: *IEEE 802.1Q Standard*[6]

How LAN Switches Work, Document ID: 10607
: Documentation Cisco™ : *How LAN Switches Work*[7]

Standards d'encapsulation dans les *trunks*
: Documentation Cisco™ : *InterSwitch Link and IEEE 802.1Q Frame Format*[8]

Configuring InterVLAN Routing and ISL/802.1Q Trunking, Document ID: 14976
: Documentation Cisco™ décrivant une configuration simple sur le routage inter-VLAN : *Configuring InterVLAN Routing and ISL/802.1Q Trunking*[9].

La segmentation des réseaux locaux
: *La segmentation des réseaux locaux*[10] : argumentation sur les fonctions de commutation et de routage.

Configuration d'une interface de réseau local
: *Configuration d'une interface de réseau local*[11] : présentation complète sur la configuration des interfaces réseau avec un système GNU/Linux. La section sur les *Fonctions réseau d'un interface* traite des réglages possibles au niveau du noyau Linux. C'est à ce niveau que l'on retrouve l'activation du routage. Voir Section 4.3, « Activation de la fonction routage ».

[6] http://standards.ieee.org/getieee802/download/802.1Q-1998.pdf
[7] http://www.cisco.com/en/US/tech/tk389/tk689/technologies_tech_note09186a00800a7af3.shtml
[8] http://www.cisco.com/en/US/tech/tk389/tk689/technologies_tech_note09186a0080094665.shtml
[9] http://www.cisco.com/en/US/tech/tk389/tk815/technologies_configuration_example09186a00800949fd.shtml
[10] http://www.linux-france.org/prj/inetdoc/articles/segmentation.lan/
[11] http://www.linux-france.org/prj/inetdoc/cours/config.interface.lan/

Guide Pratique du Filtrage de Paquets sous Linux 2.4

Guide Pratique du Filtrage de Paquets sous Linux 2.4[12] : présentation des concepts du filtrage réseau avec le noyau Linux.

Guide Pratique du NAT sous Linux 2.4

guide NAT-HOWTO[13] : présentation des concepts de la fonction de traduction d'adresses IP avec le noyau Linux.

Initiation au routage, 3ème partie

Initiation au routage, 3ème partie[14] : guide complet sur l'utilisation du logiciel Quagga pour transformer un système GNU/Linux en routeur OSPF.

Architecture réseau des travaux pratiques

Le support *Architecture réseau des travaux pratiques*[15] présente la topologie physique de la salle de travaux pratiques avec la *Disposition des équipements dans l'armoire de brassage*[16] ainsi que les configurations par défaut des équipements. On y trouve aussi le plan d'adressage IP utilisé avec les autres supports de travaux pratiques, le plan de numérotations des VLANs et les affectations des groupes de ports des commutateurs.

[12] http://www.linux-france.org/prj/inetdoc/guides/packet-filtering-HOWTO/
[13] http://www.linux-france.org/prj/inetdoc/guides/NAT-HOWTO/NAT-HOWTO-4.html#ss4.1
[14] http://www.linux-france.org/prj/inetdoc/guides/zebra.ospf/
[15] http://www.linux-france.org/prj/inetdoc/cours/archi.tp/
[16] http://www.linux-france.org/prj/inetdoc/cours/archi.tp/archi.tp.brassage.html