Réseaux Avancés

Myriam BEGEL

Cachan Réseau à Normale Sup'

Mardi 21 Octobre 2014

Plan

wireshark

`wireshark` est un outil qui permet d'analyser le trafic réseau.

Try it yourself !

```
sudo apt-get install wireshark
```

Address Resolution Protocol

Quand 2 machines sont dans le même sous-réseau :

Source en broadcast :

"Quelle est l'adresse MAC associé à *cette IP* ? Réponds à *mon IP*"

Destinataire en unicast :

"C'est moi qui ait *cette IP*. Voici *ma MAC*"

Encapsuler

"L'encapsulation est un procédé consistant à inclure les données d'un protocole dans un autre protocole."

Généralement, on encapsule dans un protocole de couche supérieure.

Utilisation de scapy

scapy permet de forger des paquets en Python "à la main"
Générons un ping

```
1  monPing= IP()/ICMP()
2  ##Pour definir l'IP source et celle du destinataire :
3  monPing.src= 'adresse_IP'
4  monPing.dst= 'adresse_IP'
5  ##Pour voir le paquet genere :
6  monPing.show()
7  ##Pour l'envoyer :
8  send(monPing)
9  ##Pour recevoir une reponse :
0  rep,non_rep = srp1(monPing)
```

netcat

`netcat` est une commande qui permet de :

- ouvrir une connection TCP
- écouter sur les ports TCP et UDP

Pour ouvrir une connection depuis machine1

```
nc adresse-machine2 port
```

Pour écouter sur la machine2

```
nc -l port
```

Plan

Protocole IP insuffisant : n'indique pas le programme concerné.
On introduit : TCP et UDP, 2 protocoles de couche supérieure.

User Data Protocol

Envoi simplement les données d'un couple (IP,port) vers un autre couple (IP,port).
Encapsulé dans un paquet IP

Structure sur 32 bits

Source Port Number	Destination Port Number
Length(UDP Header + Data)	UDP Checksum
Application Data (Message)	

User Data Protocol

Avantage : rapidité par petites quantités
Inconvénient : ne garantit pas l'ordre ni l'exactitude des données reçues
Utilisation :

- Streaming
- Jeux en lignes
- DNS
- ...

UDP

The best thing about UDP jokes, is that I don't care if you get it or not

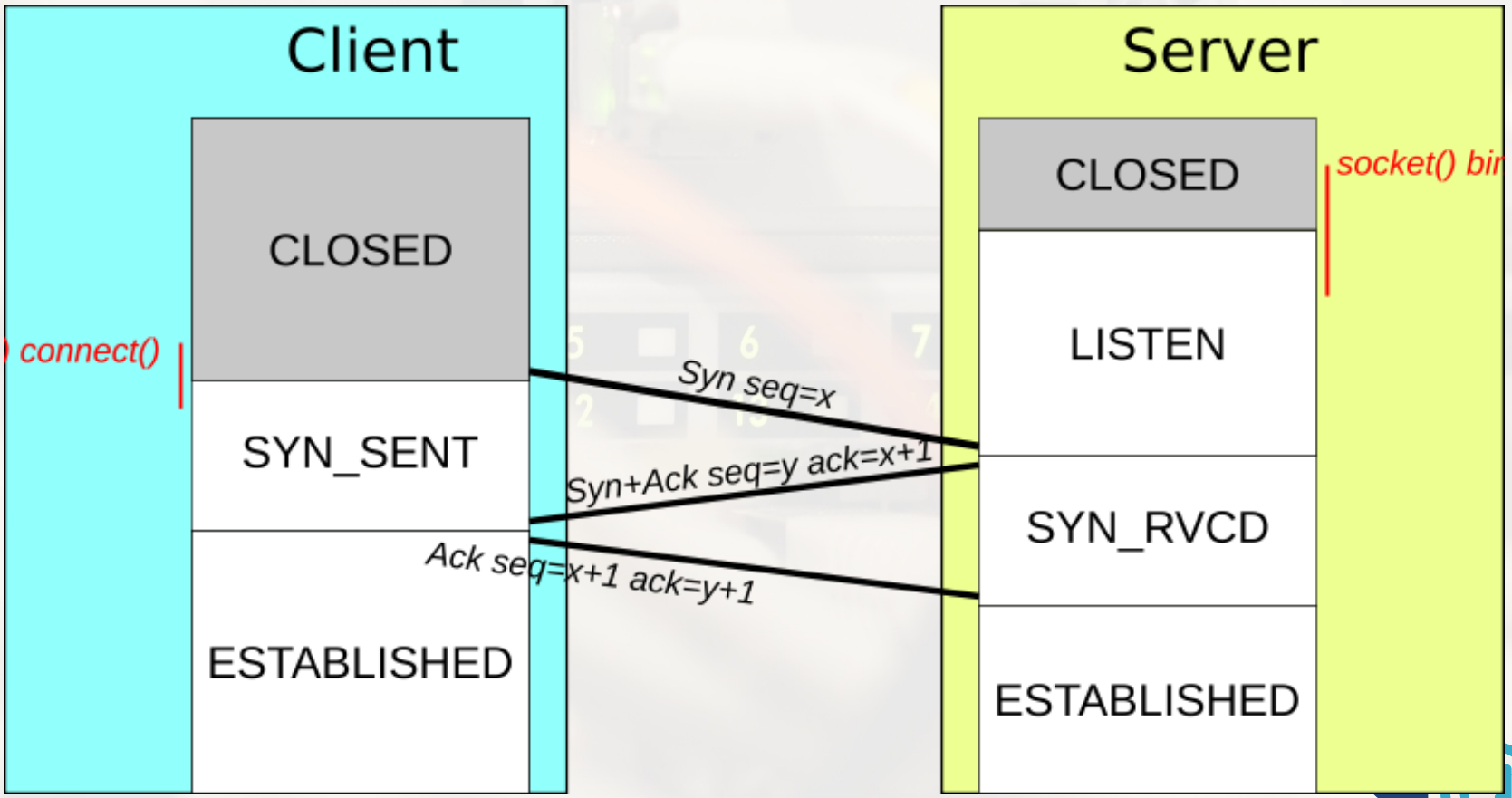
Transmission Control Protocol

Protocole qui assure l'intégrité des données mais plus complexe.

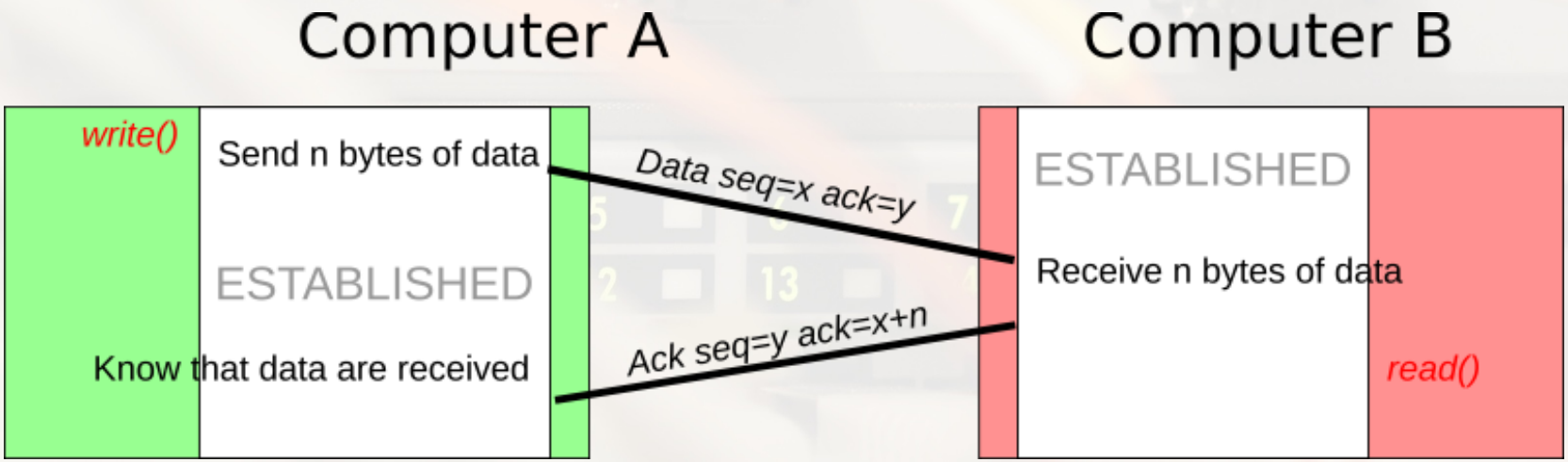
Structure :

16 bits							32 bits
Source Port							Destination Port
Sequence Number							
Acknowledgment Number (ACK)							
Offset Reserved	U	A	P	R	S	F	Window
Checksum							Urgent Pointer
Options and Padding							

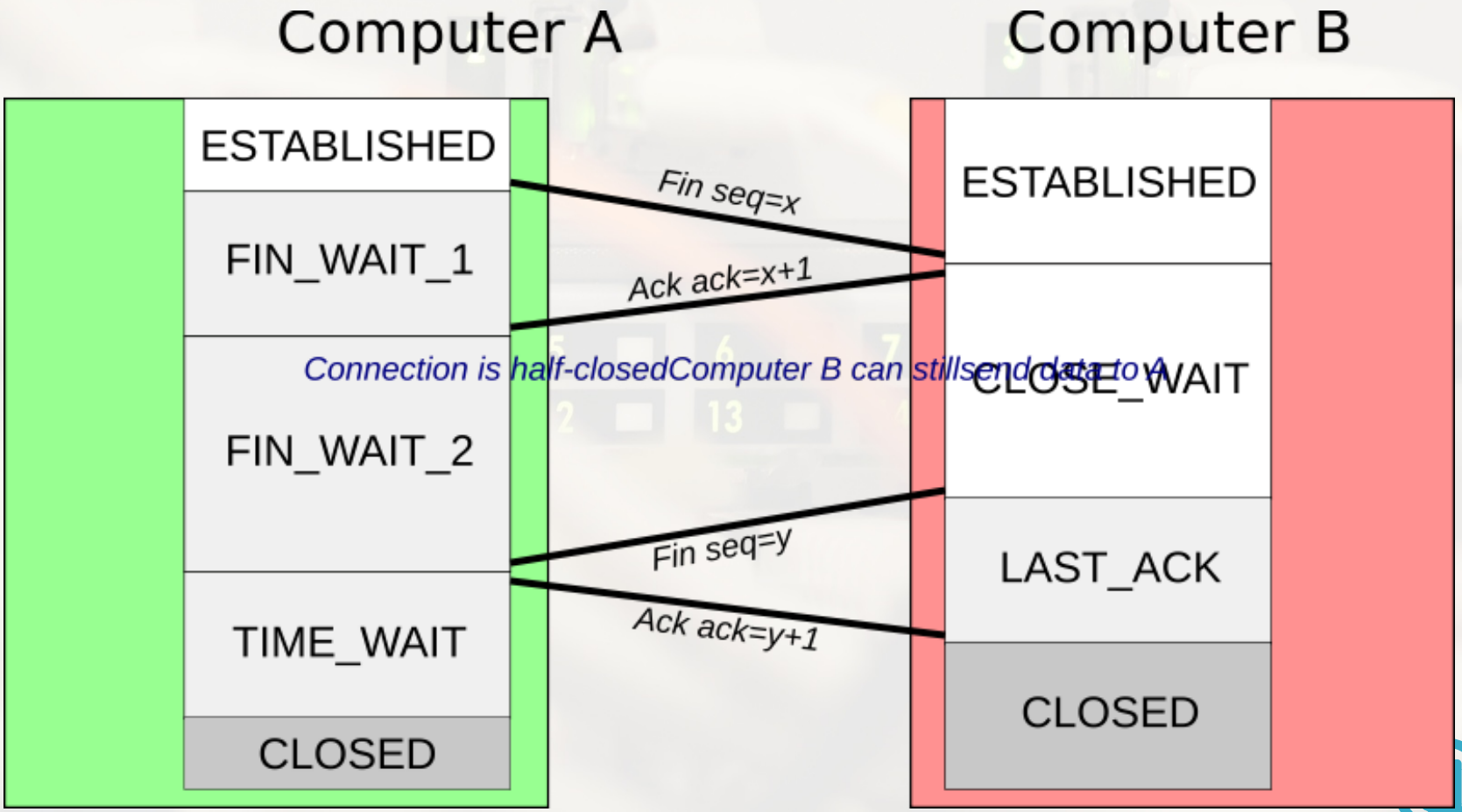
Ouverture de la connexion



Envoi des données



Arrêt de la connexion



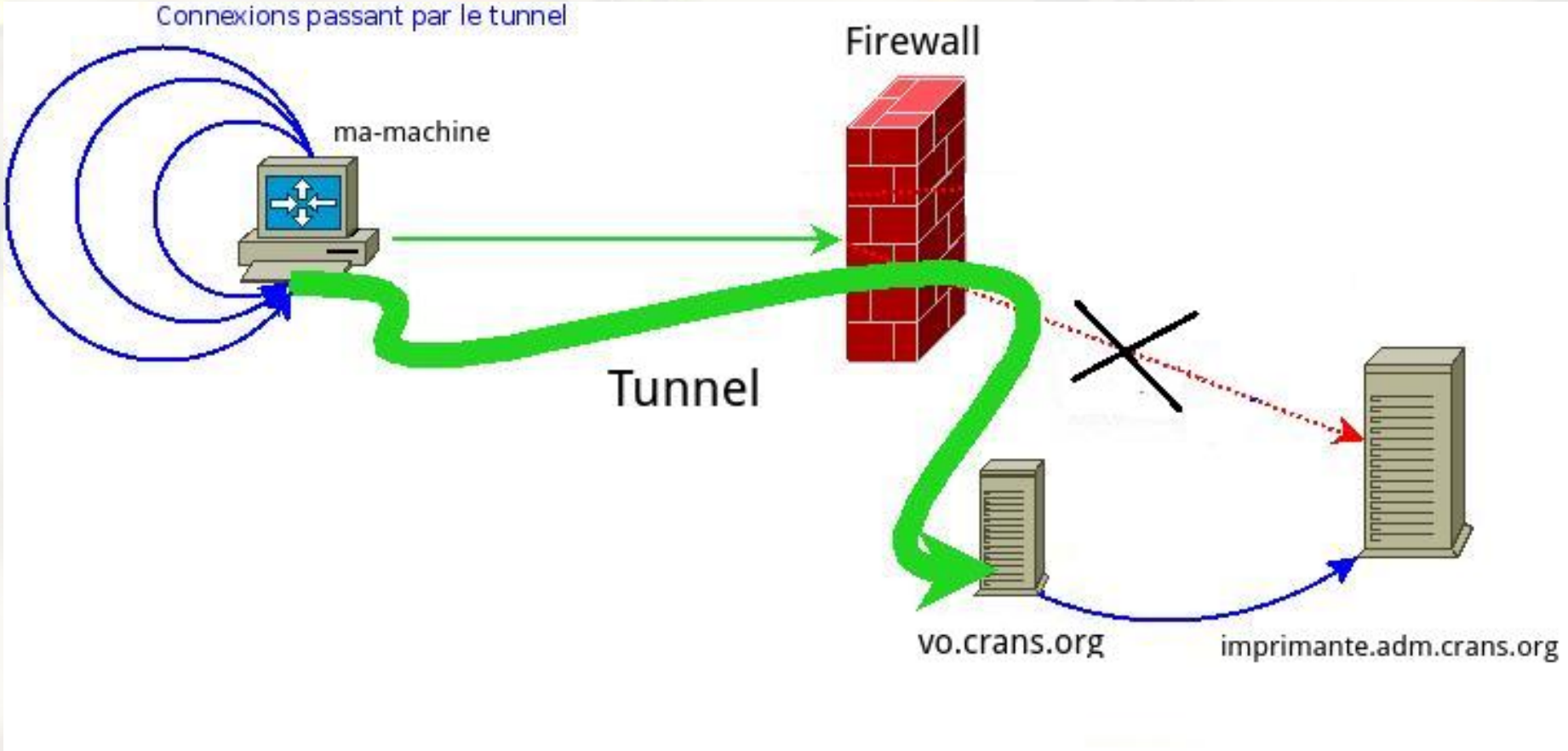
Utilisation de scapy

Ecrivons un paquet TCP :

```
1 monPaquet= IP()/TCP()
2 ##Pour definir l'IP source et celle du destinataire :
3 monPaquet.src= 'adresse_IP'
4 monPaquet.dst= 'adresse_IP'
5 ##Pour voir le paquet generer :
6 monPaquet.show()
7 ##On peut modifier de meme les valeurs de seq et ack
8 ##Pour recevoir une reponse :
9 rep , non_rep = srp1(monPaquet)
```

Plan

"Un tunnel, est une encapsulation de données d'un protocole réseau dans un autre, situé dans la même couche, ou dans une couche de niveau supérieur."



Intérêts d'un tunnel :

- Contourner un pare-feu
- Chiffrer ses données pour les isoler du reste du réseau

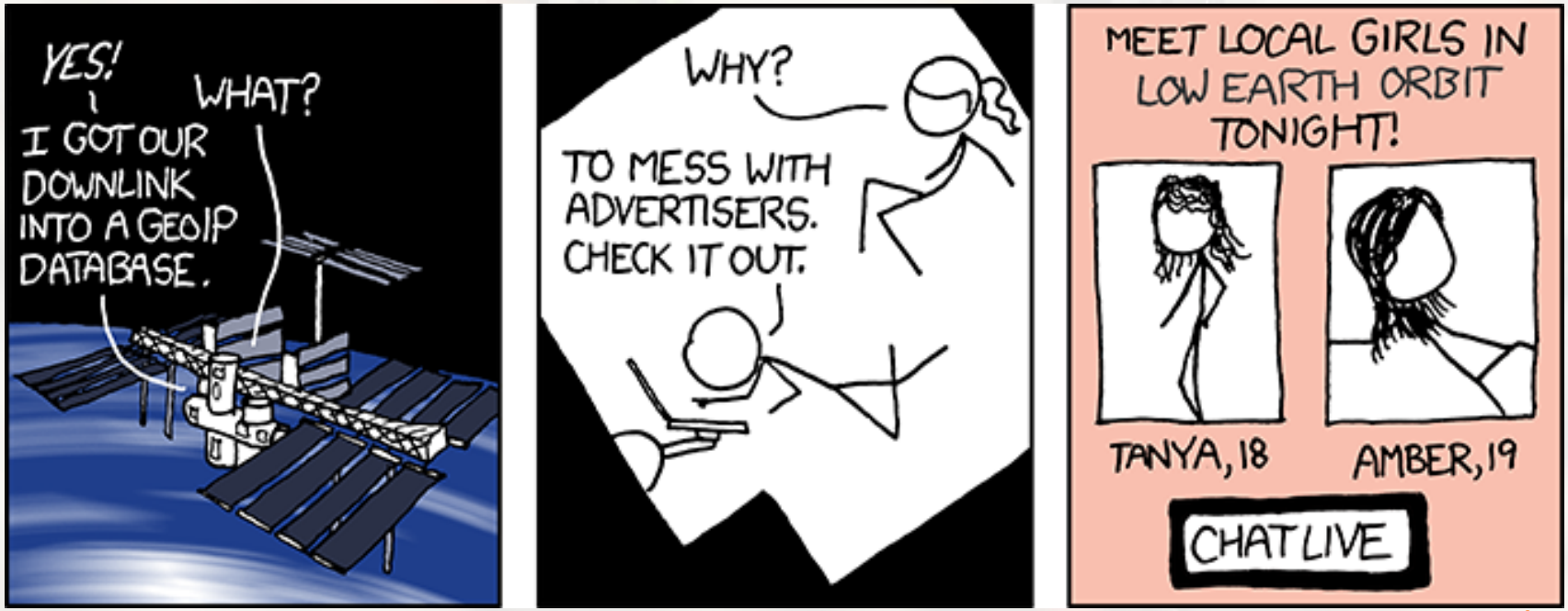
proxy SOCKS

Mise en place d'un tunnel :

```
ssh -D port user@adress
```

Configurez ensuite le proxy dans votre navigateur.
Testez votre IP sur monip.fr !

Tunnel


YES!
I GOT OUR DOWNLINK INTO A GEOIP DATABASE.
WHAT?
WHY?
TO MESS WITH ADVERTISERS. CHECK IT OUT.
MEET LOCAL GIRLS IN LOW EARTH ORBIT TONIGHT!
TANYA, 18
AMBER, 19
CHAT LIVE

Plan

Virtual Local Area Network

Normalement, chaque réseau a ses propres switch ce qui peut être coûteux et contraignant.

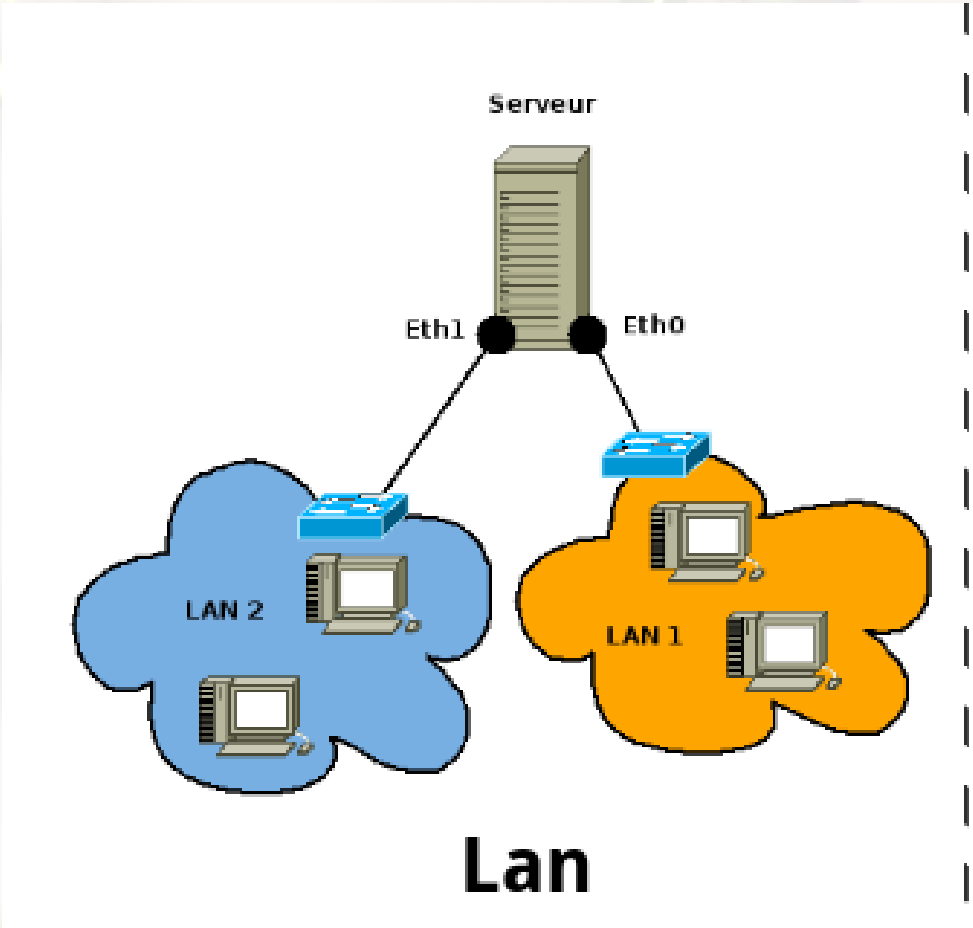
Intérêts des VLan's

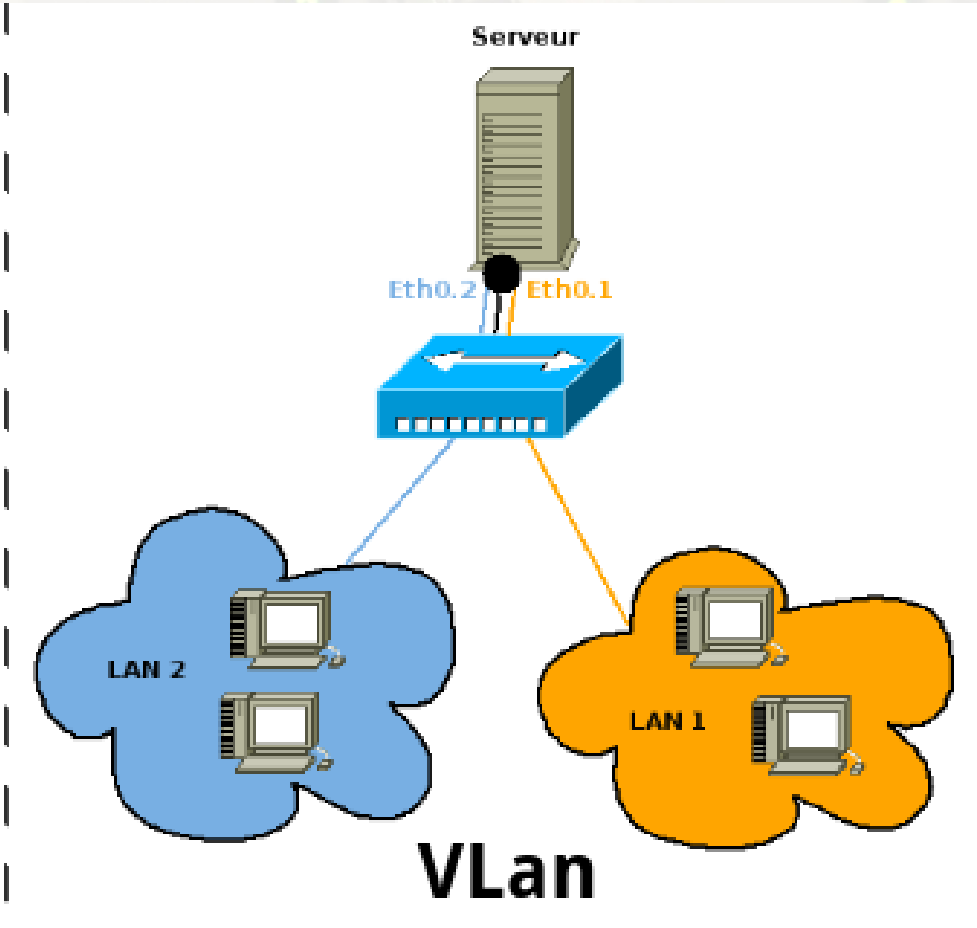
- Pouvoir séparer des flux dans un même switch
- Pouvoir changer la distribution des flux à distance

Au cr@ns, par exemple, on distingue :

- VLan Adm (administrateur)
- VLan Adhérents

VLan


Serveur
Eth1
Eth0
LAN 2
LAN 1
Lan


Serveur
Eth0.2
Eth0.1
LAN 2
LAN 1
VLan

Comment répartit-on les VLan sur les ports du switch ?

Différents niveaux :

1. VLan par ports : on associe à chaque VLan les ports du switch qui lui sont rattachés
2. VLan par MAC : de même avec les adresses MAC
3. VLan par IP : de même avec les IP

Configuration - côté serveur

Avant toute chose vous aurez besoin du paquet `VLan` et d'activer le protocole du noyau 802.1q `modprobe 8021q`

Utilisation de `vconfig`

Pour ajouter une sous interface pour le VLan 1 on utilise :

```
sudo vconfig add eth0 1
```

L'interface créée s'appelera par convention `eth0.1`
Puis il faut configurer l'interface comme une interface classique avec `ifconfig` ou `ip`
Pour supprimer cette interface il faudra exécuter :

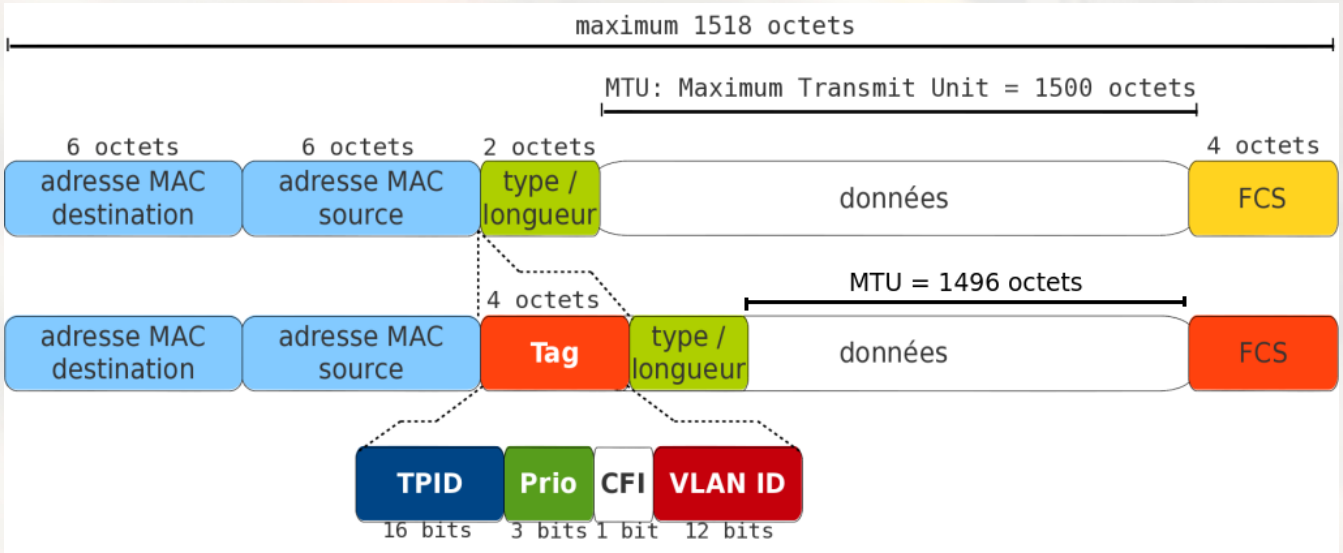
```
sudo vconfig rem eth0.1
```

Tagging

Normalement, un seul VLan peut sortir par port.
Pour remédier à cela, on peut encapsuler (norme 802.1q) le numéro du VLan dans chaque trame diffusée.
C'est le **tagging**.

Tagging

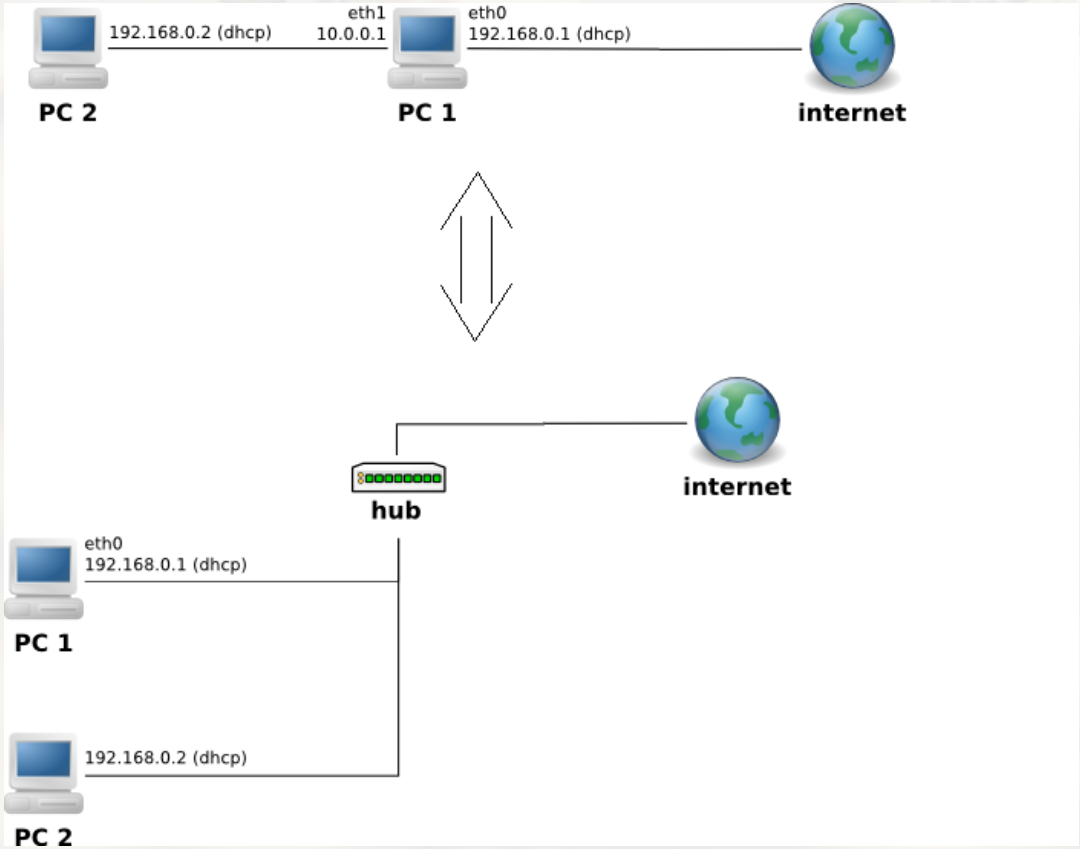
Transformation de la trame ethernet :



On remarquera bien que la MTU est diminuée de 4 octets.

Plan

Un bridge est un pont ethernet. Une machine avec 2 cartes ethernet peut servir de switch avec un bridge.



Très utile pour connecter une machine virtuelle à Internet.

3 étapes :

1. Listening : écoute tout ce qui se passe sur le réseau
2. Learning : déduit la configuration du réseau
3. Forwarding : dispatche correctement les paquets

Configuration temporaire

Il existe plusieurs méthodes, vous aurez besoin du paquet

```
bridge-utils
```

Pour créer le bridge

```
brctl addbr br0
```

Le bridge `br0` est créée il suffit maintenant de lier l'interface d'entrée avec la sortie

```
brctl addif br0 eth0 eth1
```

Pensez à vérifier la bonne configuration des interfaces créer !

Configuration plus fixe

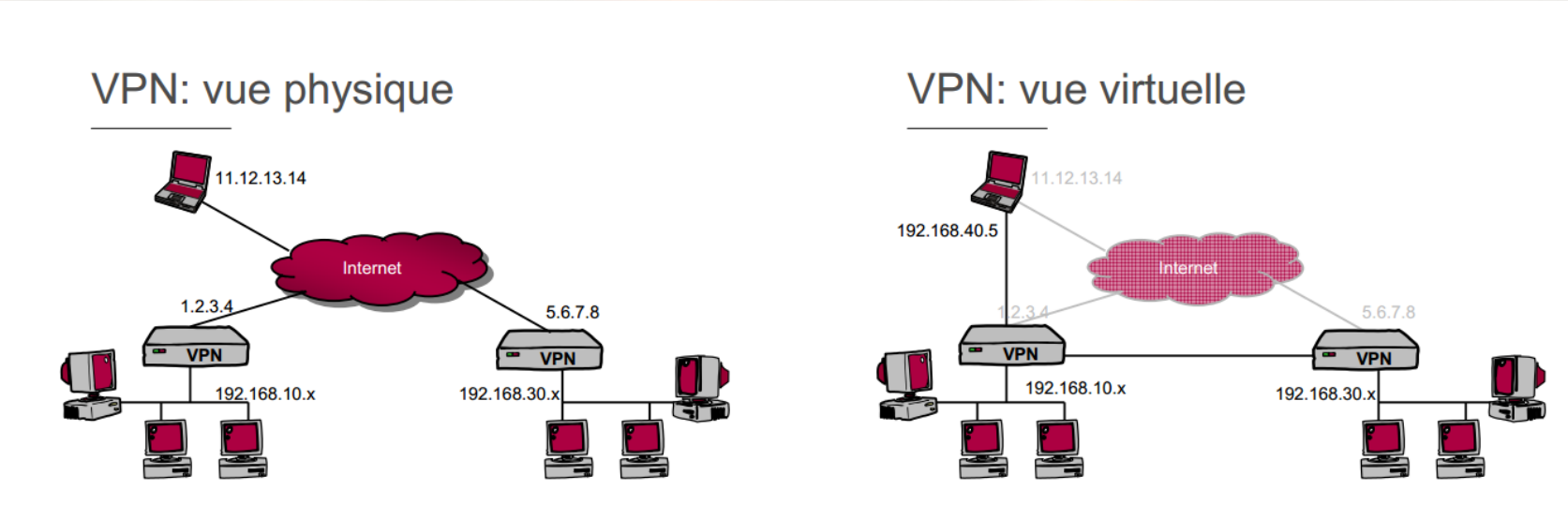
Toutefois il faudrait réitérer avec cette méthode la manipulation à chaque reboot. Pour bridger de façon plus correcte il faut modifier `/etc/network/interfaces`

```
auto br0
iface br0 inet dhcp
# Liste des interfaces qui participent au bridge
bridge_ports eth0 eth1
bridge_stp off
bridge_fd 2
bridge_maxwait 0
```

Plan

Virtual Private Network

Un VPN est un réseau virtuel sécurisé à l'intérieur d'un autre réseau (ici Internet).



Fonctionnement

Basiquement le principe du VPN repose sur les protocoles de tunnelisation, on encapsule donc les paquets à transmettre dans un protocole de même niveau OSI)
L'opération se déroule en plusieurs étapes :

- Les paquets à transmettre sont encapsulés et chiffrés par un client VPN
- Les paquets chiffrés transitent par un autre réseau (Internet par exemple)
- Les paquets sont reçus, déchiffrés et transmis par le serveur VPN

Intérets

- Accéder à un réseau local à distance (Travail à domicile, jeux en LAN etc...)
- Sécuriser la transmission de données, en permettant la connexion à distance à un réseau de confiance.

Géneration du certificat maître

Aprés installation d'OpenVPN :

```
cp /usr/share/doc/openvpn/examples/easy-rsa  /openvpn/ -R
cd  /openvpn/2.0/
```

Puis en étant sudoer transformez le fichier **vars** à votre convenance

Vous pouvez alors créer la configuration de votre certificat :

```
. ./vars
```

Supprimez les certificats déja existants

```
./clean-all
```

Et enfin génerez le certificat maître

```
./build-ca
```

Certificats et clefs pour les serveurs et clients

A présent on génere et on signe les certificats pour le client et pour le serveur

- Pour le serveur : `./build-key-server serveur`
- Pour le client : `./build-key client1`

Il reste à génerez les paramètres Diffie-Hellman pour permettre un échange sécurisé des clefs.
`./build-dh`
Vous obtenez ainsi des clefs signées pour votre serveur et votre client !

Il faut déplacer les fichiers respectivement crées dans le dossier `/etc/openvpn` de votre serveur et de votre client

Résumé

Vous devez a présent avoir
Coté Serveur :

- `ca.crt`
- `server.crt`
- `server.key`
- `dh1024.pem`

Coté Client :

- `ca.crt`
- `client.crt`
- `client.key`

Configuration du serveur

On travaillera sur le serveur `apprentis`
Vous pouvez trouver la configuration dans `/etc/openvpn/serveur.conf`
La configuration ce scinde en 3 parties importantes :

- `proto UDP/TCP` , précise si le serveur écoute sur un port TCP ou UDP
- `dev TUN` (Encapsulation IP) ou `dev TAP` (Encapsulation Ethernet) précise le type d'interface virtuelles
- Enfin configurer l'IP virtuelle du serveur et son masque de sous-réseau

Configuration du client

Au niveau du client la configuration est plus simple
Il faut éditer `/etc/openvpn/client.conf`

- Même configuration que le serveur concernant le port TCP ou UDP
- Adresse publique du serveur, la résolution DNS étant activée
  `remote apprentis.crans.org 1196`
- Vérifiez la cohérence des noms des certificats

Connexion

Pour lancer le serveur

```
sudo openvpn server.conf
```

Et pour le client

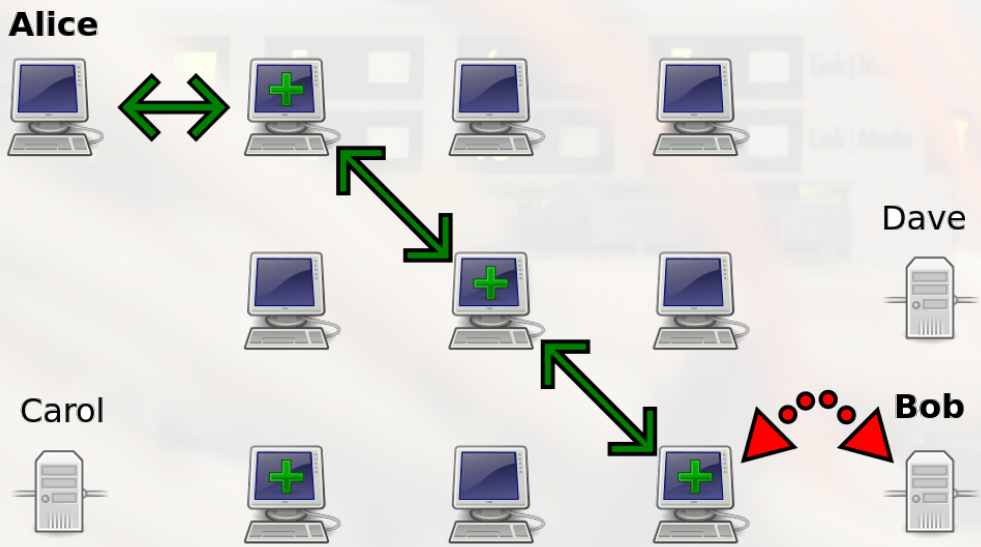
```
sudo openvpn client.conf
```

C'est fait ! ! Vous êtes connectés au VPN !

Plan

The Onion Router

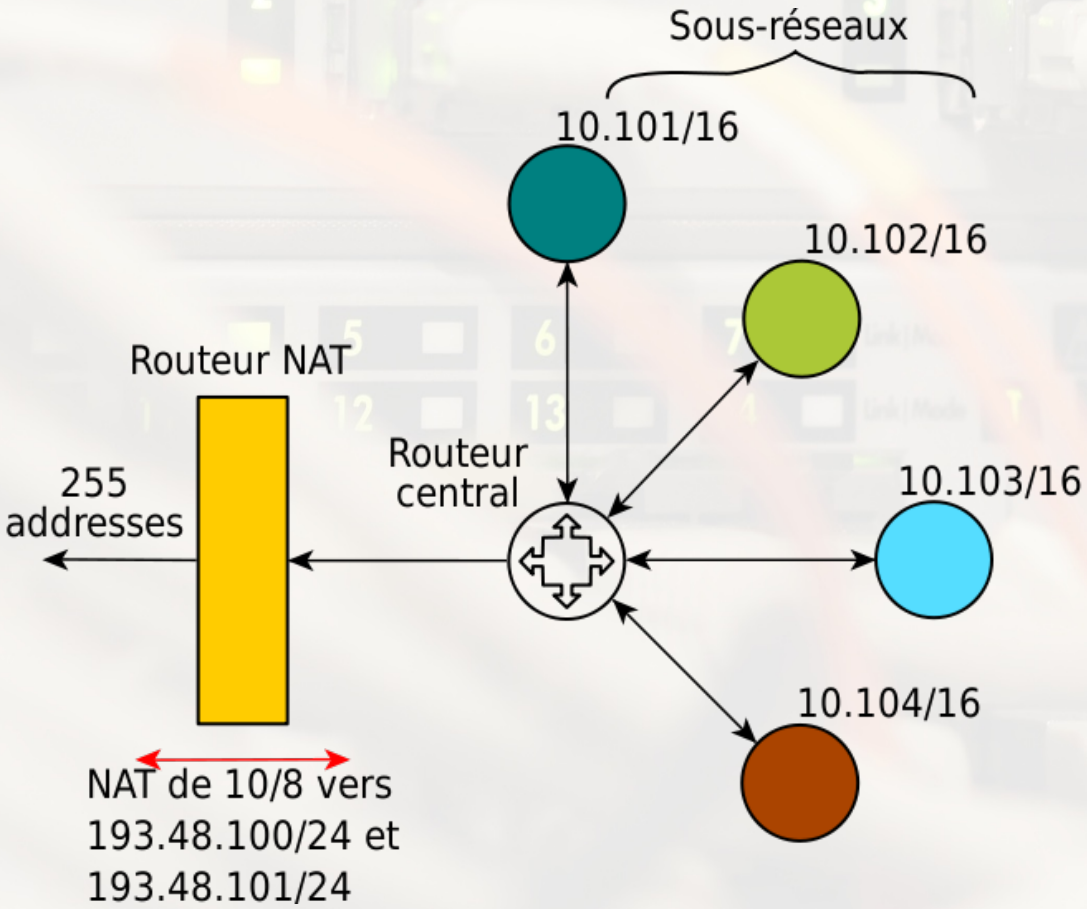
TOR est un ensemble de routeurs organisé en couches qui empêchent théoriquement l'analyse du trafic d'un utilisateur.
Pour utiliser TOR, il suffit de rediriger votre trafic par le port dédié (par défaut 9050).



Plan

Network Address Translation

Pour palier au manque d'adresse IPv4, on peut utiliser un NAT :



Questions ?