Prérequis :

L'ensemble du TD s'effectuera dans la machine virtuelle SR07-GNS3.

3 possibilités :
1/ faire tourner la machine virtuelle sur sa propre machine sur Virtualbox (recommandé)
2/ faire tourner la machine virtuelle sur les machines de la sale de TD sur Virtualbox
3/ faire tourner la machine virtuelle sur Promox (serveur https://smesr07.int.utc.fr:8006)

Celle-ci est téléchargeable à l'adresse :
https://www.utc.fr/~quetwilf/sr07/SR07-GNS3.ova
et également disponible dans le repertoire local /local2/virualbox/ova/linux/SR07-GNS3.ova (en lecture seule).

Sur les machines de la sale, un espace local sans quota est disponible pour chaque compte SR06 ou SR07 sous les répertoires /user2/users/sr06/<nom_du_compte> ou /user2/users/sr07/<nom_du_compte>

ex : pour le compte sr06a001 : /user2/users/sr06/sr06a001

Cet espace permettra de stocker la machine virtuelle finale une fois déployée.

Au préalable, celle-ci doit être importée dans VirtualBox (menu Fichier -> Import Appliance).

Configuration demandée sur la machine virtuelle :
- RAM = 3Go (2,5 Min)
- Disque = 10 Go
- 1ère carte réseau de type NAT
- 2nde carte réseau de type bridged avec l'interface n'ayant pas accès à internet.

ATTENTION : l'option "**câble connecté**" doit être cochée dans les propriétés avancées des cartes réseau.

Une fois la machine virtuelle démarée, la connexion s'effectuera avec l'utilisateur **root** , mot de passe **sr07sr07**

Exercice 1 : conteneur Docker

Dans la machine virtuelle SR07-GNS3, lancer un terminal.

Télécharger une image docker de type debian :

```
# docker pull debian
```

Démarrer une instance de cette image dans un conteneur et lancer un shell (/bin/bash) en intéractif (-i) :

```
# docker run -i --dns=195.83.155.55 -t debian /bin/bash
```

Lancer quelques installation dans le docker :

```
root@06d6690783f3 # export http_proxy= « http://proxyweb.utc.fr :3128 »
root@06d6690783f3 # export https_proxy= « https://proxyweb.utc.fr :3128 »
root@06d6690783f3 # apt-get update
root@06d6690783f3 # apt-get install net-tools
root@06d6690783f3 # apt-get install curl
root@06d6690783f3 # apt-get install wget
root@06d6690783f3 # apt-get install openssh-server
root@06d6690783f3 # apt-get install apache2
root@06d6690783f3 # apt-get install proftpd
root@06d6690783f3 # apt-get install nmap
root@06d6690783f3 # apt-get install tcpdump
root@06d6690783f3 # apt-get install nano
root@06d6690783f3 # apt-get install isc-dhcp-client
root@06d6690783f3 # apt-get install isc-dhcp-server
root@06d6690783f3 # apt-get install rsyslog
root@06d6690783f3 # apt-get install freeradius
root@06d6690783f3 # useradd sr07
root@06d6690783f3 # passwd sr07          (> entrer le mot de passer sr07sr07)
root@06d6690783f3 # exit
```

Réperer l'ID du conteneur :

```
# docker ps -a
```

Commiter les changement sur l'image debian :

```
# docker commit 06d6690783f3 debian
```

A partir de ce moment nous avons une image debian contenant toutes les modifications qu'on lui a apporté.

Créer maintenant, une image debian-sr07 permettant d'avoir une persistance du répertoire /etc :

```
# mkdir debian-sr07
# cd debian-sr07
# echo "FROM debian" > Dockerfile
# echo "CMD /bin/bash" >> Dockerfile
# echo "VOLUME /etc" >> Dockerfile
# docker build -t debian-sr07 . (attention au point à la fin)
# cd ..
```

Nous obtenons maintenant une image debian-sr07 où les données stockées dans /etc seront persistantes.

Exercice 2 : Outil de simulation d'infrastructure GNS3

Dans un terminal , activer l'interface bridge qui permettra la communication avec l'extérieur :

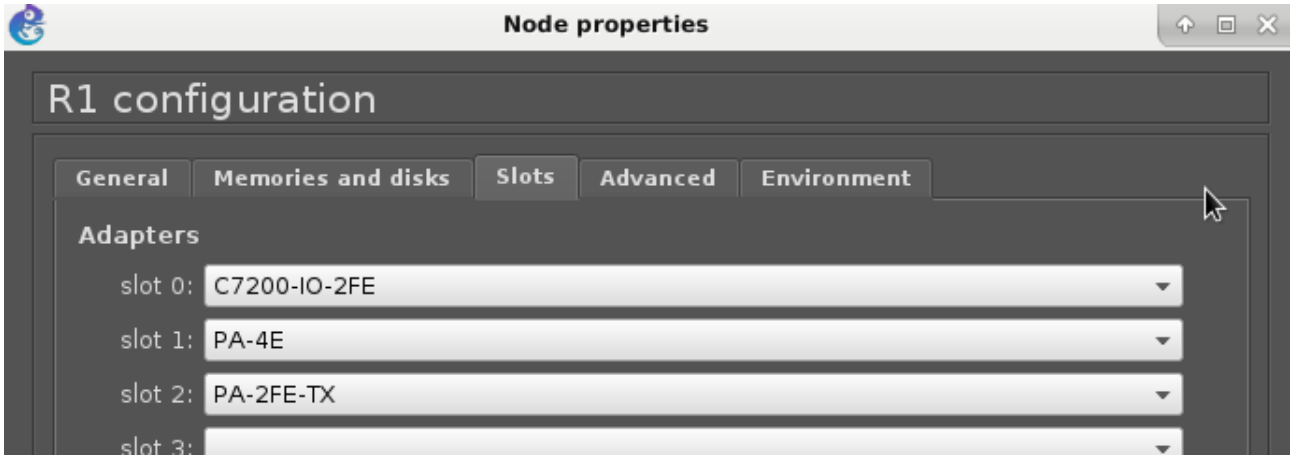
```
#virsh net-start default
#virsh net-autostart default
```

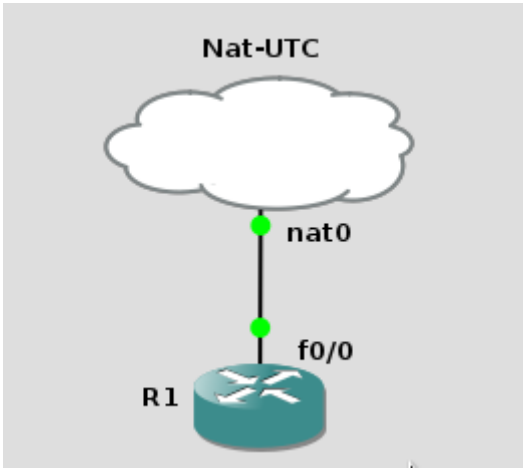
Remarque : cela permettra d'éviter l'erreur "virb0 manquante" au démarrage de la topologie GNS3

Dans un terminal lancer la commande **gns3**

Créer un nouveau projet.

Créer un routeur virtuel de type C7200 connecté à un réseau de type NAT.
Le routeur doit posséder les slots suivants :





Attention : la CPU de la machine virtuelle monte à 100% une fois le routeur démarré.

Pour remédier au problème, lancer la commande IDLE-PC sur le routeur (clic droit→ Auto IDLE PC)

```
R# configure terminal
R(config)# int Fastethernet0/0
R(config-if)# ip address dhcp
R(config-if)# end
R# write
```

Attention, le commande **write** permet de rendre la configuration persistante

Exercice 3: Routage simple

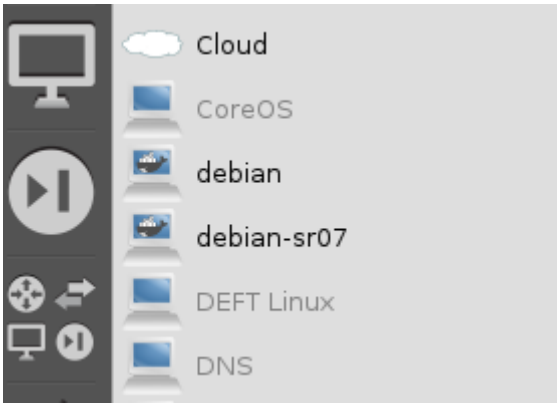
La 1ère étape consiste à créer les containers Dockers :

Dans le menu Edit→ Preferences → Docker → Docker Containers, cliquer sur le bouton New.

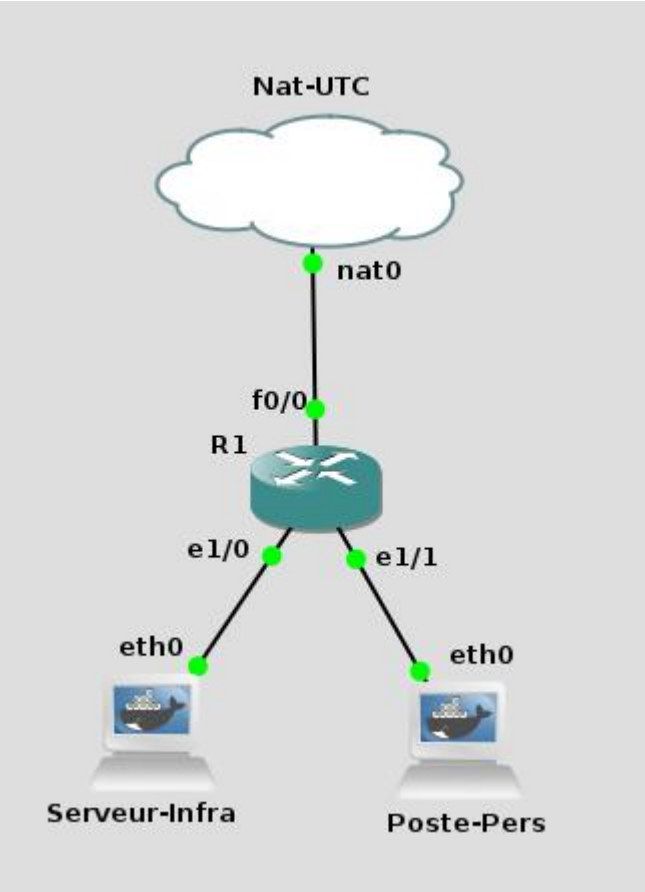
Sélectionner l'image Docker **debian-sr07:latest** créée à l'exercice 1.
Adaptater =1
Start Command = --VIDE--
Console=telnet
Variables = --VIDE --

→ Finish

Le type de machine debian apparait dans la liste des devices.



Ensuite, créer l'architecture suivante :



Sur la console du routeur R1 :

```
R1# conf t
R1 (config)# int e1/0
R1 (config-if)# ip address 10.0.1.1 255.255.255.0
R1 (config-if)# no shutdown
R1 (config-if)# exit
R1 (config)# int e1/1
R1 (config-if)# ip address 10.0.2.1 255.255.255.0
R1 (config-if)# no shut
R1 (config-if)# end
R1 # wr
```

Sur la Console du conteneur debian-sr07-1 :

```
# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel
state DOWN mode DEFAULT group default qlen 1000
```

```
    link/ether fc:15:b4:eb:65:a2 brd ff:ff:ff:ff:ff:ff
```

Repérer le nom du device correspond à l'interface réseau (eth0).

Donner une adresse IP au conteneur **Serveur-Infra**:

```
# ip address add 10.0.1.2/24 dev eth0
```

Vérifier que l'adresse est bien présente sur l'interface :

```
# ip address
.....
1: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
    link/ether c4:d9:87:6d:ea:20 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.2/24 brd 10.0.10.255 scope global dynamic noprefixroute wlan0
```

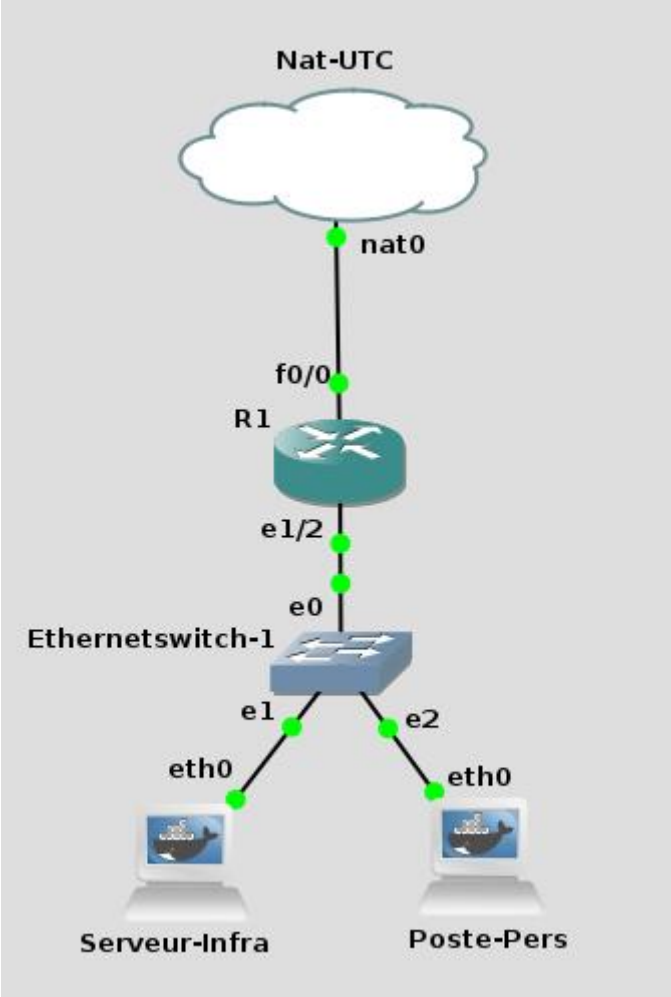
Vous devez pouvoir atteindre le routeur R1 :

```
# ping 10.0.1.1
```

Configurer le conteneur **Poste-Pers** de manière à ce qu'il soit configuré avec l'@IP : **10.0.2.2/24.**

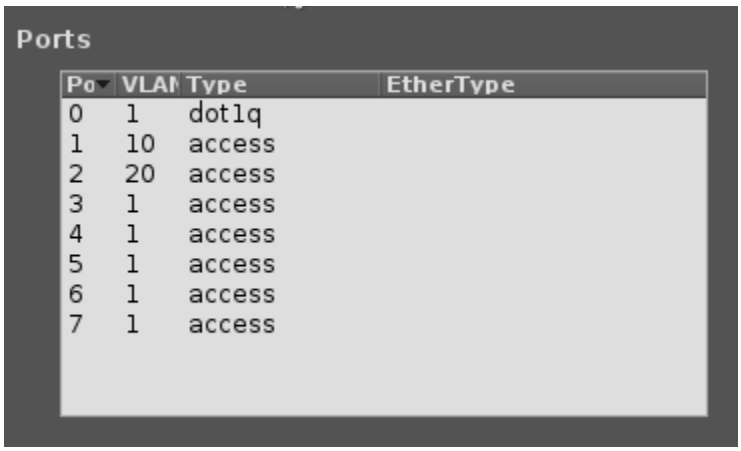
Exercice 4: Vlans

Modifier la topologie en intercallant un switch ethernet entre les conteneur Docker et le routeur en respectant les interfaces d'interconnexion du schéma.



2 vlans vont être crées sur le switch et sur le routeur :
- Vlan Infra : id 10
- Vlan Pers : id 20

La configuration du switch sera la suivante :



Le lien entre R1 et Ethernetswitch-1 est de type 802.1q de manière à pouvoir transporter plusieurs vlans.

La configuration de l'interface e1/2 du routeur s'effectuera de la manière suivante:

```
R1# conf t
R1 (config)# int e1/2
R1 (config-if)# no shutdown
R1 (config-if)# int e1/2.10
R1 (config-if)# encapsulation dot1Q 10
R1 (config-if)# ip address 10.0.10.1 255.255.255.0
R1 (config-if)# no shutdown
R1 (config-if)# int e1/2.20
R1 (config-if)# encapsulation dot1Q 20
R1 (config-if)# ip address 10.0.20.1 255.255.255.0
R1 (config-if)# no shutdown
R1 (config-if)# end
R1 # write mem
```

Modifier l'adresse IP des conteneurs à l'aide de la commande "ip address add" :
Debian-sr07-1 = 10.0.10.2/24
Debian-sr07-2 = 10.0.20.2/24

Le ping vers le routeur doit fonctionner.

Tester le ping vers le conteneur voisin.

Faire en sorte que le ping fonctionne entre les 2 conteneurs , grâce à la commande "ip route add default via xxxxx".

Exercice 5 : DHCP

Sur le routeur , configurer le service dhcp de manière à ce qu'il offre des adresses au 2 conteneurs :

```
R# conf t
R(config)# ip dhcp pool Infra-10
R(dhcp-config)# network 10.0.10.0 255.255.255.0
R(dhcp-config)# default-router 10.0.10.1
R(dhcp-config)# dns-server 195.83.155.55
R(dhcp-config)# exit
R(config)# ip dhcp pool Pers-20
R(dhcp-config)# network 10.0.20.0 255.255.255.0
R(dhcp-config)# default-router 10.0.20.1
R(dhcp-config)# dns-server 195.83.155.55
R(dhcp-config)# exit
R(config)# ip dhcp excluded-address 10.0.10.1
R(config)# ip dhcp excluded-address 10.0.20.1
R(config)# ip dhcp excluded-address 10.0.10.2
R(config)# ip dhcp excluded-address 10.0.20.2
R(config)# end
R# write mem
```

Sur les conteneurs Serveur-Infra et Poste-Pers, lancer la commande :

```
# ip address del 10.0.x.2/24 dev eth0          (avec x=10 ou 20 suivant le conteneur)
# dhclient -v
```

Les conteneurs vont maintenant changer d'@IP et être adressés en 10.0.x.**3**/24

Exercice 6 : NAT

A cette étape les conteneurs n'ont pas accès à Internet.

```
R1# sh int f0/0
FastEthernet0/0 is up …
…
  Internet address is 192.168.122.196/24
…
```

```
R1# conf t
R1(config)# int f0/0
R1(config-if)# ip nat outside
R1(config-if)# int e1/2.10
R1(config-if)# ip nat inside
R1(config-if)# int e1/2.20
R1(config-if)# ip nat inside
R1(config-if)# exit
R1(config)# access-list 1 permit 10.0.10.3
R1(config)# access-list 1 permit 10.0.20.3
R1(config)# ip nat pool SR07 192.168.122.196 192.168.122.196 prefix-length 24
R1(config)# ip nat inside source list 1 pool SR07 overload
R1(config)# end
R1# write
```

Exercice 7: Filtrage par ACL

Sur le conteneur Serveur-Infra, démarrer le service sshd :

```
# /etc/init.d/ssh restart
```

Vérifier le bon fonctionnement du service en local :

```
# netstat -laputen
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode      PID/Program name
.....
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      0          36652      -
.....
```

```
# ssh sr07@localhost
sr07@localhost's password:                  ← entrer sr07sr07
$ exit
#
```

Vérifier ensuite l'accès au conteneur debian-sr07-1 depuis le conteneur debian-sr07-2 :

```
# ssh sr07@10.0.10.3
sr07@localhost's password:                  ← entrer sr07sr07
$ exit
#
```

Sur le routeur, créer une access-list étendue d'identifiant INFRA_OUT qui permettra le ping depuis le routeur vers le conteneur debian-sr07-1 uniquement :

```
R# conf t
R(config)# ip access-list extended INFRA_OUT
R(config-ext-nacl)# permit icmp host 10.0.10.1 host 10.0.10.3
R(config-ext-nacl)# end
R#  sh ip access-lists INFRA_OUT
Extended IP access-list INFRA_OUT
        10      permit icmp host 10.0.10.1 host 10.0.10.3
```

Appliquer ensuite cette access-list sur le traffic sortant de l'interface e1/2.10 :

```
R# conf t
R(config)# int e1/2.10
R(config-if)# ip access-group INFRA_OUT out
R(config-if)# end
R# write mem
```

Tester le ping depuis le conteneur debian-sr07-2.

Ajouter à cette même access-list , la permission de pinguer depuis le conteneur debian-sr07-2 vers le conteneur debian-sr07-1 :

```
R# conf t
R(config)# ip access-list extended INFRA_OUT
R(config-ext-nacl)# permit .....
R(config-ext-nacl)# deny ip any any log
R(config-ext-nacl)# end
R#  sh ip access-lists INFRA_OUT
        10      permit icmp host 10.0.10.1 host 10.0.10.3
        20      permit ….
        30      deny ip any any log
```

Que peut on voir sur la console du routeur lorsque le conteneur debian-sr07-2 tente un ssh sur le conteneur debian-sr07-1?

Dans cette même access-list 100, défiltrer en plus des défiltrages icmp précédents uniquement le traffic ssh (tcp port 22) en provenance du conteneur debian-sr07-2 vers de conteneur debian-sr07-1.

Depuis le conteneur debian-sr07-1 , tester les commandes :

```
# export http_proxy="http://proxyweb.utc.fr:3128"
# export https_proxy="https://proxyweb.utc.fr:3128"
# apt-get install locate
```

Analyser l'erreur de la dernière commande et le résultat sur la console du routeur. Expliqueur pourquoi elle ne fonctionne pas.

La solution , **les access-lists réflexives**.

Créer une access list INFRA_IN que nous allons appliquer au traffic entrée de l'interface e1/2.10 (c'est à dire pour le traffic en provenance du conteneur debian-sr07-1 vers le routeur ou l'extérieur).

```
R# conf t
R(config)# ip access-list extended INFRA_IN
R(config-ext-nacl)# permit udp host 10.0.10.2 host 195.85.155.55 eq 53 reflect
DNSREQ
R(config-ext-nacl)# deny ip any any log
R(config-ext-nacl)# exit
R(config)# int e1/2.10
R(config-if)# ip access-group INFRA_IN in
R(config-if)# exit
R(config)# ip access-list extended INFRA_OUT
R(config-ext-nacl)# 24 evaluate DNSREQ
R# write mem
```

```
R# show access-lists
```

Que donne le résultat de la commande “**apt-get install locate**” ?

Ajouter une entrée réflexive l’autorisation de debian-sr07-1 vers 195.83.155.55 sur le port 3128 en tcp et l’évaluation correspondante.

La commande “**apt-get install locate**” doit aboutir.

Sur le conteneur debian-sr07-1 , relancer le renouvellement du bail DHCP avec la commande :

```
# dhclient -v
```

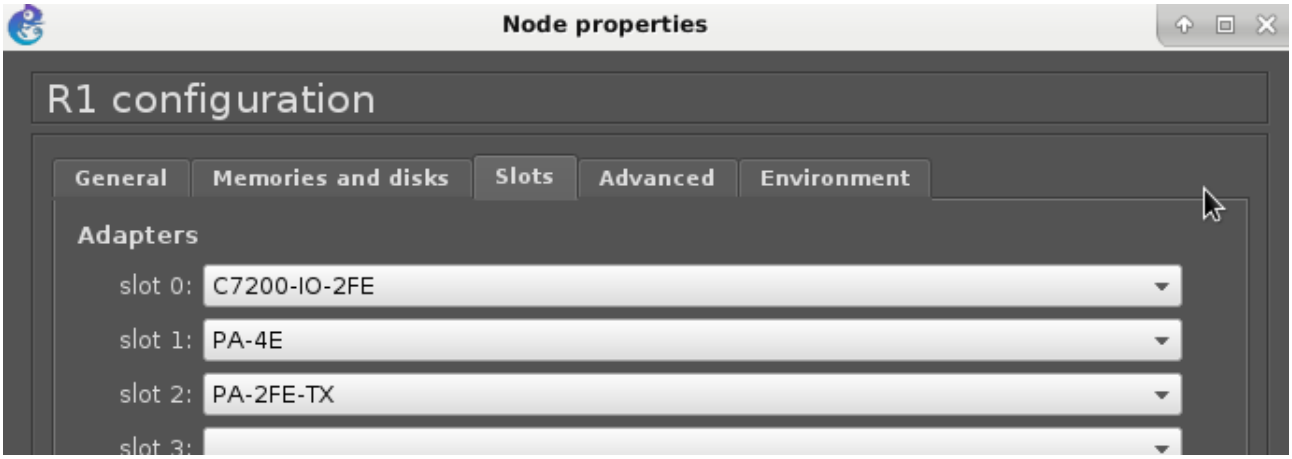
Vérifier la console du routeur? A quel niveau peut on voir un bloquage? Quel protocole est bloqué?

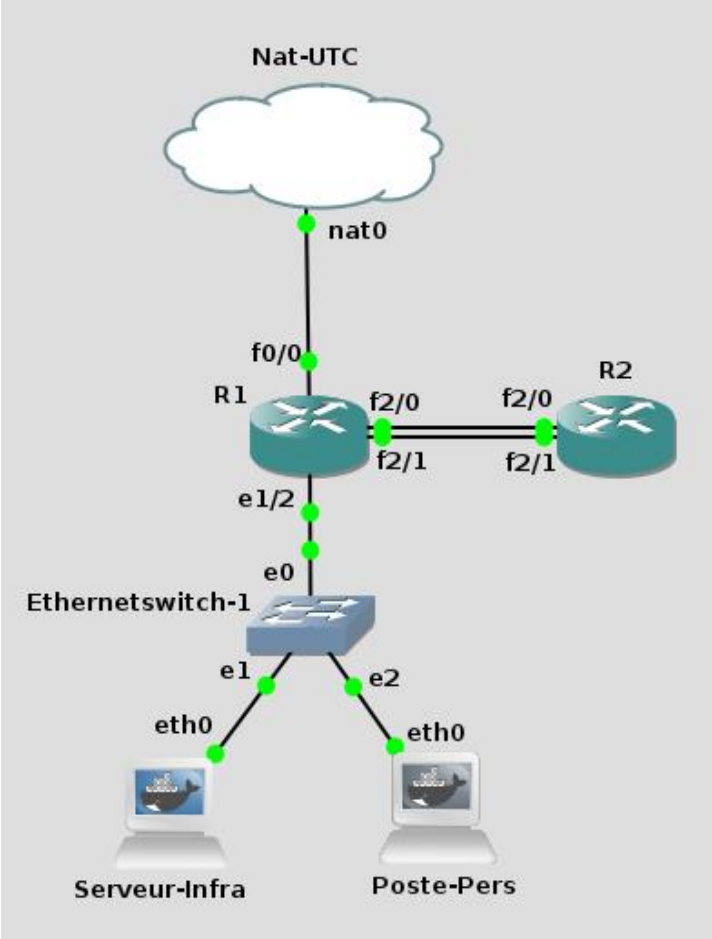
Ajouter une règle permettant de débloquer les reqêtes DHCP sur la bonne access-list.

Exercice 8 : Aggrégation de lien LACP (802.3ad)

Ajouter un routeur R2 de même type que R1 :

Interconnecté le avec R1 en respectant les 2 liens : f2/0 ↔ f2/0 et f2/1 ↔ f2/1





Sur le router R1, configurer un port-channel regroupant les 2 interfaces f2/0 et f2/1

```
R1# conf t
R1(config)# interface Port-Channel 1
R1(config)# exit
R1(config)# interface FastEthernet2/0
R1(config)# channel-group 1
R1(config)# interface FastEthernet2/1
R1(config)# channel-group 1
R1(config)# end
```

Vérifier l'état du Port-Channel :

```
R1# sh interface Port-Channel 1
.....
N° of active members.....
        Member 0 : FastEthernet2/0
        Member 1 : FastEthernet2/1
```

Faire la même configuration sur le routeur R2.

Analyser l'état du Port-Channel.

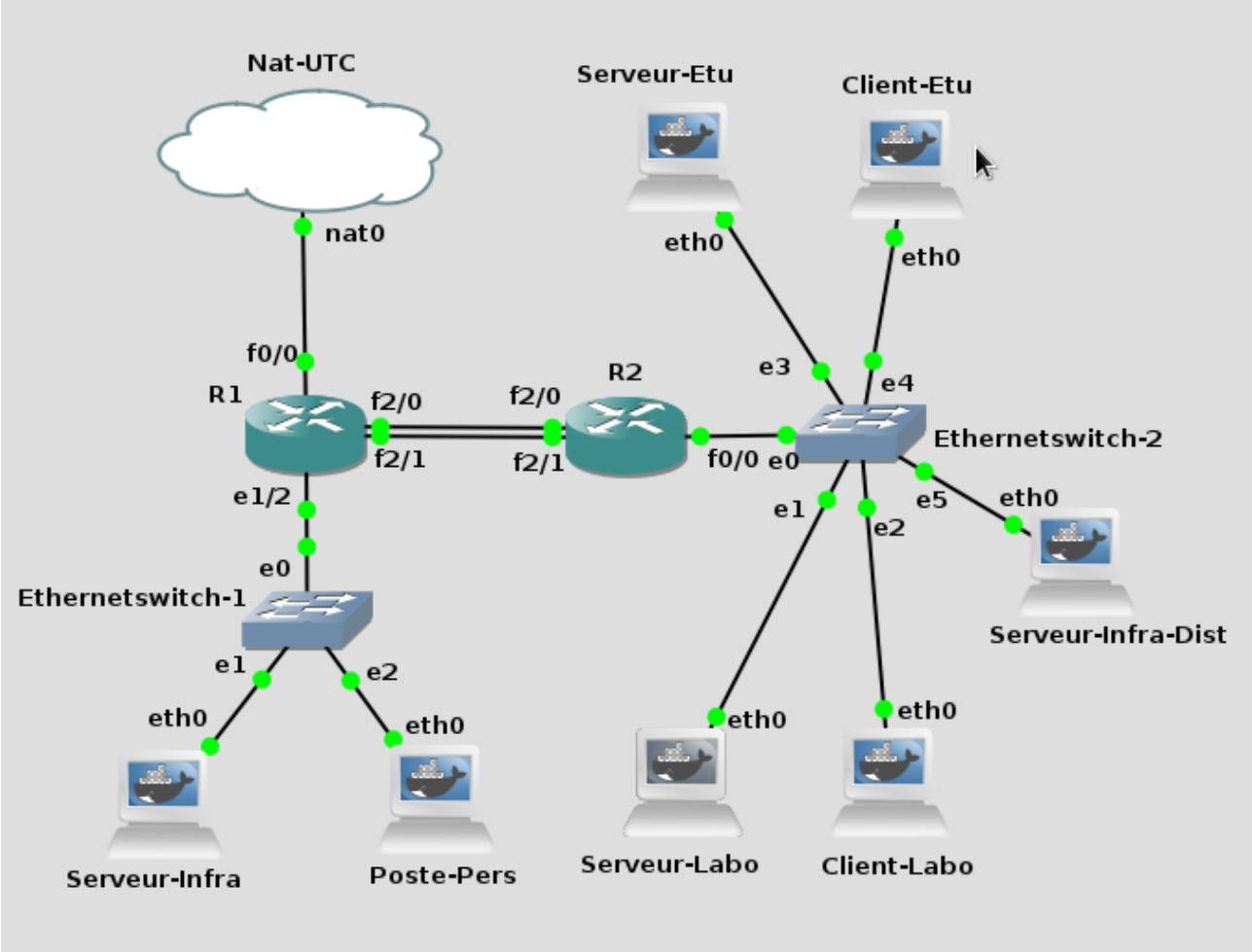
Affecter des adresses IP aux routeurs pour l'interconnexion :

@IP pour Po1 sur R1 = 10.0.99.1/24
@IP pour Po1 sur R2 = 10.0.99.2/24

Depuis le routeur R2, exécuter un ping vers le routeur R1.

Exercice 9 : Configuration d'un site distant

Connecter un switch Ethernet au routeur R2 sur l'interface FastEthernet0/0

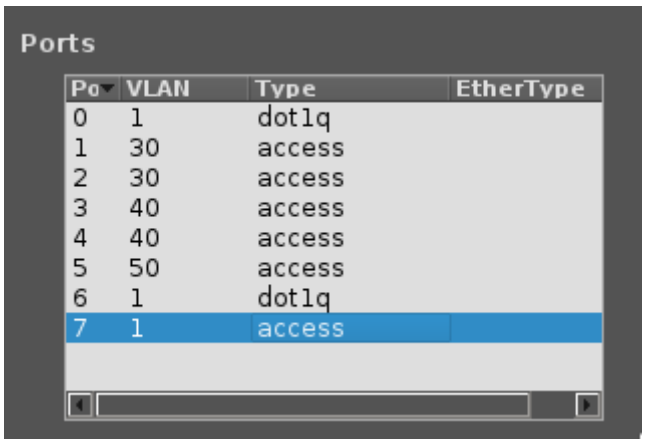


a. Vlans :

Sur le site distant, 3 vlans seront créés :

- Vlan LABO : id 30
- Vlan ETU : id 40
- Vlan INFRA_DIST : id 50

Configurer les ports du switch de la manière suivante :



Ajouter 2 conteneurs de type debian-sr07 à chaque vlan :
vlan LABO :
- Serveur-Labo : adresse MAC = 2e:00:00:00:30:03
- Client-Labo : adresse MAC = 2e:00:00:00:30:04

vlan ETU :
- Serveur-Etu : adresse MAC = 2e:00:00:00:40:03
- Client-Etu : adresse MAC = 2e:00:00:00:40:04

vlan INFRA_DIST :
- Serveur-Infra-Dist : adresse MAC = 2e:00:00:00:50:03

Pour chaque conteneur, éditer la configuration : clic droit → configure → network Configuration → edit

Ajouter les lignes :

```
auto eth0
iface eth0 inet dhcp
        hwaddress ether 2e:00:00:00:xx:xx
```

Pour le conteneur Serveur-Infra-Dist , éditer la configuration : clic droit → configure → network Configuration → edit

```
auto eth0
iface eth0 inet static
        hwaddress ether 2e:00:00:00:50:03
        ip address 10.0.50.3
        netmask 255.255.255.0
        gateway 10.0.50.1
        up echo “nameserver” 195.83.155.55 > /etc/resolv.conf
```

Créer les interfaces vlans sur le routeur R2 et suivre l’adressage suivant :

@ip sur Fa0/0.30 = 10.0.30.1/24
@ip sur Fa0/0.40 = 10.0.40.1/24
@ip sur Fa0/0.50 = 10.0.50.1/24

Pinguer le Routeur R2 sur 10.0.50.1.

Pinguer le Routeur R1 sur 10.0.99.1.
=> Si cela ne fonctionne pas, , résoudre les problèmes de route sur la source et la destination :

- sur R2 :

```
# ip route 0.0.0.0 0.0.0.0 10.0.99.1
```

- sur R1 :

```
# ip route 10.0.50.0 255.255.255.0 10.0.99.2
```

- sur R1 :

```
# int Po1
(config-if)# ip nat inside
```

b. DHCP :

Repérer les adresses MAC des 2 Serveurs et 2 Clients.

Sur le Serveur-Infra-Dist, configurer un serveur DHCP (isc-dhcp-server) de manière à offrir des adresses au vlan ETU et LABO.

Les fichiers à modifier sont les suivants :
***/etc*/dhcp/dhcpd.conf** :

```
subnet 10.0.50.0 … {
}

subnet 10.0.30.0 … {
        options routers  …..;
        options dns-name-servers 195.83.155.55;
}

subnet 10.0.40.0 … {
        options routers  …..;
        options dns-name-servers 195.83.155.55;
}

host Serveur-Labo {
        hardware ethernet xx:xx:xx:xx:xx:xx;        ← adresse repérée avec la commande ip link
        fixed-address 10.0.30.3;
}

host Client-Labo {
        hardware ethernet xx:xx:xx:xx:xx:xx;        ← adresse repérée avec la commande ip link
        fixed-address 10.0.30.4;
```

```
}

host Serveur-Etu {
        hardware ethernet xx:xx:xx:xx:xx:xx;        ← adresse repérée avec la
commande ip link
        fixed-address 10.0.40.3;
}

host Client-Etu {
        hardware ethernet xx:xx:xx:xx:xx:xx;        ← adresse repérée avec la
commande ip link
        fixed-address 10.0.40.4;
}
```

et le fichier ***/etc/*default/isc-dhcp-server**

```
INTERFACES="eth0"
```

Le service sera démarré à l'aide de la commande "service isc-dhcp-server restart".

Le routeur R2 sera configuré de manière à relayer les requêtes DHCP vers le serveur Serveur_Infra_Dist.

```
R2# conf t
R2(config) #int Fa0/0.30
R2(config) #ip helper-address 10.0.50.3
R2(config) #int Fa0/0.40
R2(config) #ip helper-address 10.0.50.3
R2(config)# end
R2# write mem
```

Valider la délivrance d'adresses par le serveur DHCP en analysant le fichier **/var/log/syslog**.

Les 2 clients doivent pouvoir pinguer les 2 serveurs.

Dans le fichier **/etc/apache2/sites-enabled/000-default.conf** de chaque serveur, configurer un ServerName avec le nom du serveur correspondant.

Démarrer ensuite apache2 avec la commande **"/etc/init.d/apache2 start**".

Le client du même vlan doit être capable de lancer la commande :"**wget 10.0.xx.3**" vers le serveur du même vlan.

c. Access-lists

Sur le routeur R2, sur chacune des interfaces Fa0.0.30 , Fa0/0.40 et Fa0/0.50, configurer une access-lists en entrée et une access-list en sortie de manière a ce que :

- Les 2 clients et les 2 serveurs obtiennent des adresses IP du serveur Serveur-Infra-Dist

- Seul Client-Labo puisse consulter le site internet du Serveur-Labo

- Client-Labo puisse consulter le site de Serveur-Etuet de Serveur-Labo.