RAPPORT DU PROJET DU MODULE INFRASTRUCTURE RÉSEAUX

MASTER 1 CRYPTIS

CHOUGAR MALIK - EL KARN ADHAM

Mai 2021

Introduction

Ce projet consiste à mettre en place un tunnel L2TPv3 entre deux parties du réseaux, à mettre en place deux vlans le tunnel, servira à faire communiquer les réseaux appartenant aux même vlan ensemble.

Nous allons dans ce rapport, présenter et détailler les différents travaux que nous avons réalisés.

Nous détaillerons notre travail avec des captures d'écrans

1) Configuration du dhcp et attributions des adresses mac

Nous attribuons statiquement à nos hôtes des adresses mac :

```
113
114 ip netns exec post3 ifconfig post3-eth0 hw ether 66:20:00:00:00:00
115 ip netns exec post4 ifconfig post4-eth0 hw ether 66:03:03:03:03:03
116 ip netns exec post1 ifconfig post1-eth0 hw ether 66:06:06:06:06:06
117 ip netns exec post2 ifconfig post2-eth0 hw ether 66:09:09:09:09:09
118
119
```

ce choix est du au fait que nous devons donner des routes par défaut différentes aux hôtes selon leur emplacement dans le réseaux.

Nous voulons que les poste 3 et 1 appartiennent au vlan 200 et les poste 2 et 4 au vlan 100. et nous voulons que les poste 1 et 2 emprunte comme route par défaut le routeur 2 au lieu du routeur 1 (aucune raison d'emprunter le tunnel pour accèder à internet).

Attribution des ips :

nous lançons dans le netns du routeur 1 la commande:

```
# dnsmasq -d -C dhcp_config -z
```

le fichier dhcp_config est un fichier de configuration dhcp personnalisé, il contient :

```
dhcp-host=66:03:03:03:03:03,set:post4

dhcp-host=66:09:09:09:09:09,set:post2

dhcp-host=66:20:00:00:00:00,set:post3

dhcp-host=66:06:06:06:06:06,set:post1

dhcp-option=tag:post3,option:router,192.168.200.254
dhcp-range=tag:post3,192.168.200.1,192.168.200.252,255.255.255.0

dhcp-option=tag:post4,option:router,192.168.100.254
dhcp-range=tag:post4,192.168.100.1,192.168.100.252,255.255.255.0

dhcp-option=tag:post1,option:router,192.168.200.253
dhcp-range=tag:post1,192.168.200.1,192.168.200.252,255.255.255.0

dhcp-option=tag:post2,option:router,192.168.100.253
dhcp-range=tag:post2,192.168.100.1,192.168.100.252,255.255.255.0
```

2) Traffic Vlan entre un hôte et un routeur

Nous allons montrer comment le le poste 1 communique avec le poste 3 à travers le tunnel et passant par le routeur r1.

Informations sur le poste 3 :

adresses ip et mac :

```
2: post3-eth0.200@post3-eth0: <BRO
 default qlen 1000
    link/ether 66:20:00:00:00:00 b
    inet 192.168.200.131/24 brd 19
```

route par défault :

```
default via 192.168.200.254 dev post3-eth0.200
192.168.200.0/24 dev post3-eth0.200 proto kernel scope link src 192.168.200.131
```

Informations sur le poste 1 :

adresses ip et mac :

```
2: post1-eth0.200@post1-eth0: <BRO
ault qlen 1000
    link/ether 66:06:06:06:06:06 b
    inet 192.168.200.53/24 brd 192
```

route par défault :

```
root@aelk-Aspire-A715-73G:/home/aelk# ip r
default via 192.168.200.253 dev post1-eth0.200
192.168.200.0/24 dev post1-eth0.200 proto kernel scope link src 192.168.200.53
root@aelk-Aspire-A715-73G:/home/aelk#
```

Poste 3 (côté gauche du tunnel) a bien routeur r1 comme passerelle par défault et Poste 1 (côté droit du tunnel) a bien r2 comme passerelle par défaut.

Nous allons écouter sur r1 (interface tunnel) pour voir comme se déroule un ping entre poste3 et post1.

```
root@aelk-Aspire-A715-73G:/home/aelk# tcpdump -i tunnel -nvvl icmp
tcpdump: listening on tunnel, link-type EN10MB (Ethernet), capture size 262144 bytes
23:24:43.610204 IP (tos 0x0, ttl 64, id 14165, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.200.53 > 192.168.200.131: ICMP echo request, id 34863, seq 1, length 64
23:24:43.610447 IP (tos 0x0, ttl 64, id 19592, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.200.131 > 192.168.200.53: ICMP echo reply, id 34863, seq 1, length 64
```

Les ip correspondent.

Maintenant nous sniffer un ping entre poste 4 (coté gauche du tunnel et le routeur r2)

ips du routeur r2 :

```
4: tunnel.100@tunnel: <BROADCAST,MULTICAST,UP,LOWER_UP>
n 1000
    link/ether 0a:62:59:e6:7c:61 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.253/24 scope global tunnel.100
       valid_lft forever preferred_lft forever
    inet6 fe80::862:59ff:fee6:7c61/64 scope link
       valid_lft forever preferred_lft forever
5: tunnel.200@tunnel: <BROADCAST,MULTICAST,UP,LOWER_UP>
n 1000
    link/ether 0a:62:59:e6:7c:61 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.253/24 scope global tunnel.200
```

ips du poste 4 :

```
       valid_lft forever preferred_lft
2: post4-eth0.100@post4-eth0: <BROADCA
ault qlen 1000
    link/ether 66:03:03:03:03:03 brd f
    inet 192.168.100.186/24 brd 192.16
       valid_lft 3534sec preferred_lft
```

```
root@aelk-Aspire-A715-73G:/home/aelk# tcpdump -i any -nvvl icmp
tcpdump: listening on any, link-type LINUX_SLL (Linux cooked v1), capture size 262144 bytes
23:33:25.229693 ethertype IPv4, IP (tos 0x0, ttl 64, id 6195, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.100.186 > 192.168.100.253: ICMP echo request, id 42086, seq 1, length 64
23:33:25.229693 ethertype IPv4, IP (tos 0x0, ttl 64, id 6195, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.100.186 > 192.168.100.253: ICMP echo request, id 42086, seq 1, length 64
23:33:25.229693 IP (tos 0x0, ttl 64, id 6195, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.100.186 > 192.168.100.253: ICMP echo request, id 42086, seq 1, length 64
23:33:25.229742 IP (tos 0x0, ttl 64, id 61905, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.100.253 > 192.168.100.186: ICMP echo reply, id 42086, seq 1, length 64
23:33:25.229745 ethertype IPv4, IP (tos 0x0, ttl 64, id 61905, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.100.253 > 192.168.100.186: ICMP echo reply, id 42086, seq 1, length 64
```

on voit bien l'utilisation du tunnel le trafic est répliqué.

Nous allons tester une connexion socat entre poste 1 et routeur r1.

On écoute sur r1 :

```
root@aelk-Aspire-A715-73G:/home/aelk# socat TCP4-LISTEN:8000,reuseaddr -
```

On se connecte depuis poste 1:

```
root@aelk-Aspire-A715-73G:/home/aelk# socat - TCP4:192.168.200.254:8000
```

On sniffe sur r2 sur l'interface du tunnel :

```
root@aelk-Aspire-A715-73G:/home/aelk# tcpdump -i tunnel -nvvl tcp
tcpdump: listening on tunnel, link-type EN10MB (Ethernet), capture size 262144 bytes
23:47:07.569408 IP (tos 0x0, ttl 64, id 22983, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.200.53.59324 > 192.168.200.254.8000: Flags [S], cksum 0x072d (correct), seq 80041820, win 642
40, options [mss 1460,sackOK,TS val 831889065 ecr 0,nop,wscale 7], length 0
23:47:07.569776 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.200.254.8000 > 192.168.200.53.59324: Flags [S.], cksum 0x27f4 (correct), seq 3585187055, ack
80041821, win 64676, options [mss 1418,sackOK,TS val 1364400553 ecr 831889065,nop,wscale 7], length 0
23:47:07.569868 IP (tos 0x0, ttl 64, id 22984, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.200.53.59324 > 192.168.200.254.8000: Flags [.], cksum 0x12ac (incorrect -> 0x5144), seq 1, ac
k 1, win 502, options [nop,nop,TS val 831889066 ecr 1364400553], length 0
^C
```

On voit le tcp handshake sur l'interface du tunnel côté r2.

Si on sniffe sur l'interface vlan du routeur r2 :

```
tcpdump: listening on tunnel.200, link-type EN10MB (Ethernet), capture size 262144 bytes
23:50:52.242336 IP (tos 0x0, ttl 64, id 23984, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.200.53.59328 > 192.168.200.254.8000: Flags [S], cksum 0x8f50 (correct), seq 877212172, win 64
240, options [mss 1460,sackOK,TS val 832113738 ecr 0,nop,wscale 7], length 0
23:50:52.243132 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.200.254.8000 > 192.168.200.53.59328: Flags [S.], cksum 0x7a65 (correct), seq 623187337, ack 8
77212173, win 64676, options [mss 1418,sackOK,TS val 1364625227 ecr 832113738,nop,wscale 7], length 0
23:50:52.243363 IP (tos 0x0, ttl 64, id 23985, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.200.53.59328 > 192.168.200.254.8000: Flags [.], cksum 0x12ac (incorrect -> 0xa3b5), seq 1, ac
k 1, win 502, options [nop,nop,TS val 832113739 ecr 1364625227], length 0
23:51:06.683707 IP (tos 0x0, ttl 64, id 23986, offset 0, flags [DF], proto TCP (6), length 55)
    192.168.200.53.59328 > 192.168.200.254.8000: Flags [P.], cksum 0xf4ce (correct), seq 1:4, ack 1, win
502, options [nop,nop,TS val 832128179 ecr 1364625227], length 3
23:51:06.684545 IP (tos 0x0, ttl 64, id 31302, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.200.254.8000 > 192.168.200.53.59328: Flags [.], cksum 0x32dd (correct), seq 1, ack 4, win 506
, options [nop,nop,TS val 1364639668 ecr 832128179], length 0
```

On voit bien le trafic qui est répliqué sur cette interface.

Nous vérifions que le protocole arp fonctionne :

on fait une requête du poste 2 vers google.com

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netns exec post2 wget google.com
--2021-05-19 12:04:29--  http://google.com/
Resolving google.com (google.com)... 142.250.75.238, 2a00:1450:4007:806::200e
Connecting to google.com (google.com)|142.250.75.238|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://www.google.com/ [following]
--2021-05-19 12:04:29--  http://www.google.com/
Resolving www.google.com (www.google.com)... 216.58.204.132, 2a00:1450:4007:812::2004
Connecting to www.google.com (www.google.com)|216.58.204.132|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: 'index.html.1'

index.html.1                [ <=>                                ]  13,49K  --.-KB/s    in 0,02s
```

on sniffe sur le routeur B (routeur de sortie de côté droit du réseaux) :

```
255 aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netns exec rB tcpdump -i any -nvvl arp
tcpdump: listening on any, link-type LINUX_SLL (Linux cooked v1), capture size 262144 bytes
12:04:23.748074 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 172.16.2.254 tell 172.16.2.253, length 28
12:04:23.748113 ARP, Ethernet (len 6), IPv4 (len 4), Reply 172.16.2.254 is-at 12:4b:06:53:2b:f9, length 28
^C
2 packets captured
```

Donc la grace à notre config dhcp, les poste 1 et 2 passe par le routeur r2 et non par le routeur r1.

4) Etude du protocole L2TPv3 et de la technologie VXLAN :

A - Présentation du protocole L2TPv3 :

Standardisé dans la RFC 3931, le protocole **Layer 2 Tunneling Protocol Version 3** ou **L2TPv3** est une amélioration du protocole L2TPv2. Contrairement au protocole L2TPv2 qui n'autorise que l'encapsulation du protocole PPP dans IP, le protocole L2TPv3 autorise l'encapsulation de n'importe quel protocole de niveau 2 dans IP. Le protocole L2TPv3 est aussi considéré comme étant plus complet que GRE, du fait qu'il peut être associé à une authentification AAA (Authentication, Authorization and Accounting).

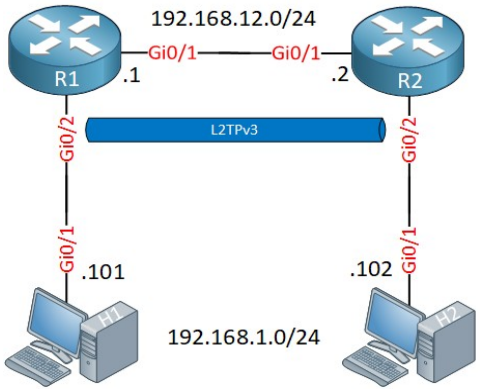


Schéma illustrant l'utilisation du protocole L2TPv3

B - Présentation de la technologie des VXLANs :

Standardisé dans la RFC 7348, le **Virtual Extensible Local Area Network** ou **VXLAN** est une technologie qui est proche des VLANs. VXLAN est une technologie de virtualisation réseau qui augmente l'évolutivité dans les environnements cloud (informatique en nuage). Cette technologie permet d'encapsuler des trames Ethernet de niveau 2 (correspondant à la couche liaison du modèle OSI) dans des datagrammes UDP de niveau 4 (correspondant à la couche transport du modèle OSI). L'Internet Assigned Numbers Authority (IANA) attribut aux VXLANs le numéro de port de destination UDP n° 4789.

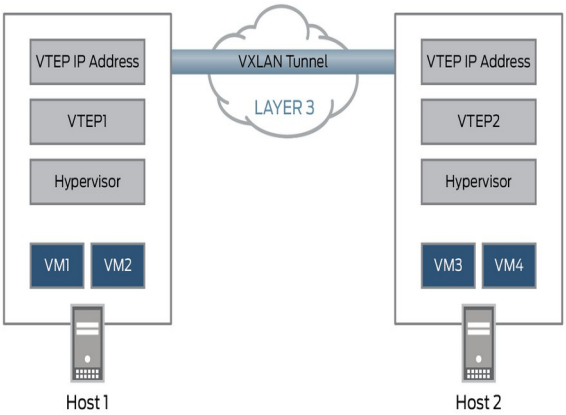


Schéma illustrant l'utilisation de la technologie VXLAN

C - Comparaisons entre les deux solutions :

La différence que l'on peut citer est que L2TPv3 encapsule n'importe quel protocole de niveau 2 dans du niveau 3 alors que la technologie VXLAN encapsule du niveau 2 dans du niveau 4. Elle possède également des mise en œuvre totalement différentes dans Linux.

D – Mise en œuvre dans Linux des VXLAN dans Open vSwitch :

Par défaut, Open vSwitch utilise le port de destination numéro 4879 pour VXLAN mais on peut le changer à la main et choisir le numéro de post 8472, comme le montre l'illustration ci-dessous :

```
$ ovs-vsctl add-br br0
$ ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1 type=vxlan \
    options:remote_ip=192.168.1.2 options:key=flow options:dst_port=8472
```

E - Solution de chiffrement du trafic L2TPv3 ou VXLAN :

Pour les deux solutions, le trafic circule en clair tout en apportant une isolation entre les différents hôtes connectés. On peut tout de même inclure du chiffrement en utilisant IPSec.

F – Comparaison avec MPLS :

Standardisé dans la RFC 3031, **MultiProtocol Label Switching** ou **MPLS** est un protocole appelé protocole de couche « 2. 5 » ou « 2/3 ». Il représente un mécanisme de transport de données basé sur la commutation de labels, c'est-à-dire qu'il y a des étiquettes qui sont insérées à l'entrée du réseau MPLS et qui sont retirées à sa sortie. MPLS permet de transporter tout type de trafic, par exemple des paquets IPV4,IPV6, trames Ethernet ou ATM, chose qui le différencie par rapport à L2TPv3 et VXLAN.

Par rapport à MPLS, VXLAN va permettre d'encapsuler les trames Ethernet clientes dans des trames IP qui assureront le transport.

Sources des illustrations :

https://networklessons.com/cisco/ccie-routing-switching-written/l2tpv3-layer-2-tunnel-protocol-version-3

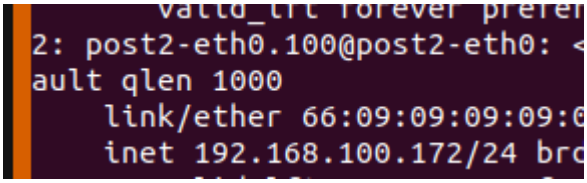
https://www.juniper.net/fr/fr/products-services/what-is/vxlan/

https://docs.openvswitch.org/en/latest/faq/vxlan/

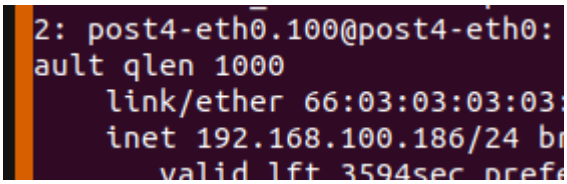
5) Comparaison de l2TPv3 en mode Ip et Udp à l'aide du protocole icmp

Nous allons communiquer de poste 2 vers poste 4 en icmp.

Adresse ip poste 2 :



Adresse ip poste 4 :



On écoute sur r1 :

A) En mode ip

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netns exec
 r1 tcpdump -i tunnel -nvvl -w ip.pcap
```

Le pcap correspondant :

```
        0x0010:  0121 0000 0000 0000 a9fe 0121           .!.........!
elk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo tcpdump  -qns
0 -X -r ip.pcap
eading from file ip.pcap, link-type EN10MB (Ethernet)
2:35:39.383396 IP 192.168.100.172.68 > 192.168.100.254.67: UDP, length 300
        0x0000:  4500 0148 426c 4000 4011 ac3d c0a8 64ac  E..HBl@.@..=..d.
        0x0010:  c0a8 64fe 0044 0043 0134 3000 0101 0600  ..d..D.C.40.....
        0x0020:  db34 0b2c 00fc 0000 c0a8 64ac 0000 0000  .4.,......d.....
        0x0030:  0000 0000 0000 0000 6609 0909 0909 0000  ........f.......
        0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0050:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

B)En mode Udp

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netns exec
 r1 tcpdump -i tunnel -nvvl -w udp.pcap
```

Le pcap correspondant :

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo tcpdump  -qns
 0 -X -r udp.pcap
reading from file udp.pcap, link-type EN10MB (Ethernet)
13:29:59.557590 IP 0.0.0.0.68 > 255.255.255.255.67: UDP, length 300
        0x0000:  4510 0148 0000 0000 8011 3996 0000 0000  E..H......9.....
        0x0010:  ffff ffff 0044 0043 0134 9d6c 0101 0600  .....D.C.4.l....
        0x0020:  7410 892d 00ff 0000 0000 0000 0000 0000  t..-............
        0x0030:  0000 0000 0000 0000 6620 0000 0000 0000  ........f.......
        0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0050:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0080:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0090:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

6) Difference Gre et L2tp

Mtu gre :

```
5: eoip1@NONE: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1476 qdisc fq
    link/ether 26:52:80:e1:7d:a4 brd ff:ff:ff:ff:ff:ff
```

Mtu L2tp :

```
6: l2tpeth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel master tunnel state UNKNOWN mode DEFAULT group default qlen 1000
    link/ether 5a:8f:00:37:da:6d brd ff:ff:ff:ff:ff:ff
```

On voit que la valeur de la Mtu est différente.

7) Comparaison vitesse (iperf) ipsec et sans ipsec

Avec ipsec

r1 :

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netn
exec r1 iperf -s
------------------------------------------------------------
erver listening on TCP port 5001
CP window size:  128 KByte (default)
------------------------------------------------------------
  4] local 192.168.100.254 port 5001 connected with 192.168.100.253 port 56306
 ID] Interval       Transfer     Bandwidth
  4]  0.0-10.0 sec   459 MBytes   383 Mbits/sec
```

poste 1 :

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netns
exec post1 iperf -c 192.168.100.254
------------------------------------------------------------
Client connecting to 192.168.100.254, TCP port 5001
TCP window size:  212 KByte (default)
------------------------------------------------------------
[  3] local 192.168.200.53 port 56306 connected with 192.168.100.254 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   459 MBytes   384 Mbits/sec
```

r1 :

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netn
exec r1 iperf -s
------------------------------------------------------------
erver listening on TCP port 5001
CP window size:  128 KByte (default)
------------------------------------------------------------
  4] local 192.168.100.254 port 5001 connected with 192.168.100.253 port 56320
 ID] Interval       Transfer     Bandwidth
  4]  0.0-10.0 sec  1.17 GBytes  1.00 Gbits/sec
```

poste 1 :

```
aelk@aelk-Aspire-A715-73G:~/hd/computer_science_hd/university/m1_cryptis/infra/projet$ sudo ip netns
exec post1 iperf -c 192.168.100.254
------------------------------------------------------------
Client connecting to 192.168.100.254, TCP port 5001
TCP window size:  544 KByte (default)
------------------------------------------------------------
[  3] local 192.168.200.53 port 56320 connected with 192.168.100.254 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  1.17 GBytes  1.00 Gbits/sec
```

Avec ipsec c'est beaucoup plus lent, environ 700 Mb de difference par second.

8) Accès à internet

switch internet et autorisation du forwarding :

```
# activer le routage sur r1 et r2
ip link set internet up
ip a add dev internet 10.87.0.1/24
ip route add dev internet 172.16.1.0/24 via 10.87.0.253 proto static
ip route add dev internet 172.16.2.0/24 via 10.87.0.252 proto static
sudo sysctl net.ipv4.conf.all.forwarding=1
ip netns exec r1 sudo sysctl net.ipv4.conf.all.forwarding=1
ip netns exec r2 sudo sysctl net.ipv4.conf.all.forwarding=1
ip netns exec rA sudo sysctl net.ipv4.conf.all.forwarding=1
ip netns exec rB sudo sysctl net.ipv4.conf.all.forwarding=1
```

règles iptables :

```
iptables -t nat -A POSTROUTING -s 172.16.1.0/24 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 172.16.2.0/24 -j MASQUERADE

ip netns exec r1 iptables -t nat -A POSTROUTING -s 192.168.100.0/24 -j MASQUERADE
ip netns exec r1 iptables -t nat -A POSTROUTING -s 192.168.200.0/24 -j MASQUERADE

ip netns exec r2 iptables -t nat -A POSTROUTING -s 192.168.100.0/24 -j MASQUERADE
ip netns exec r2 iptables -t nat -A POSTROUTING -s 192.168.200.0/24 -j MASQUERADE
```

On fait du nat masquerade sur notre machine principale et sur les routeur r1 et r2 pour pouvoir permettre le retour des paquets.

9) Interdiction du trafic en vlan 100 et 200

avec ip rule :

```
#interdiction par ip rule
:'ip netns exec r1 ip rule add from 192.168.100.0/24 to 192.168.200.0/24  prio 16050 prohibit
ip netns exec r1 ip rule add from 192.168.200.0/24 to 192.168.100.0/24  prio 16050 prohibit
ip netns exec r2 ip rule add from 192.168.100.0/24 to 192.168.200.0/24  prio 16050 prohibit
ip netns exec r2 ip rule add from 192.168.200.0/24 to 192.168.100.0/24  prio 16050 prohibit'
```

avec iptables :

```
#interdiction par iptables
ip netns exec r1 iptables -t filter -A FORWARD -s 192.168.100.0/24 -d 192.168.200.0/24 -j DROP
ip netns exec r1 iptables -t filter -A FORWARD -s 192.168.200.0/24 -d 192.168.100.0/24 -j DROP
ip netns exec r2 iptables -t filter -A FORWARD -s 192.168.100.0/24 -d 192.168.200.0/24 -j DROP
ip netns exec r2 iptables -t filter -A FORWARD -s 192.168.200.0/24 -d 192.168.100.0/24 -j DROP
```

Conclusion

Ce projet nous a permis de manipuler les tunnels, les vlans, les règles ip rules et le firewall (iptables). La partie qui était un peu difficile c'est configurer le dhcp de façon à donner des routes par défault différentes selon les adresses mac.

Démo

1) Configuration dhcp

sudo ip netns exec r1 dnsmasq -d -z -C dhcp_config

sudo ip netns exec post2 dhclient

2) test connexion et interdictions

sudo ip netns exec post2 ping -c 1 8.8.8.8

sudo ip netns exec post2 ping -c 1 ip_post4 //ça marche

sudo ip netns exec post2 ping -c 1 ip_post3 //marche