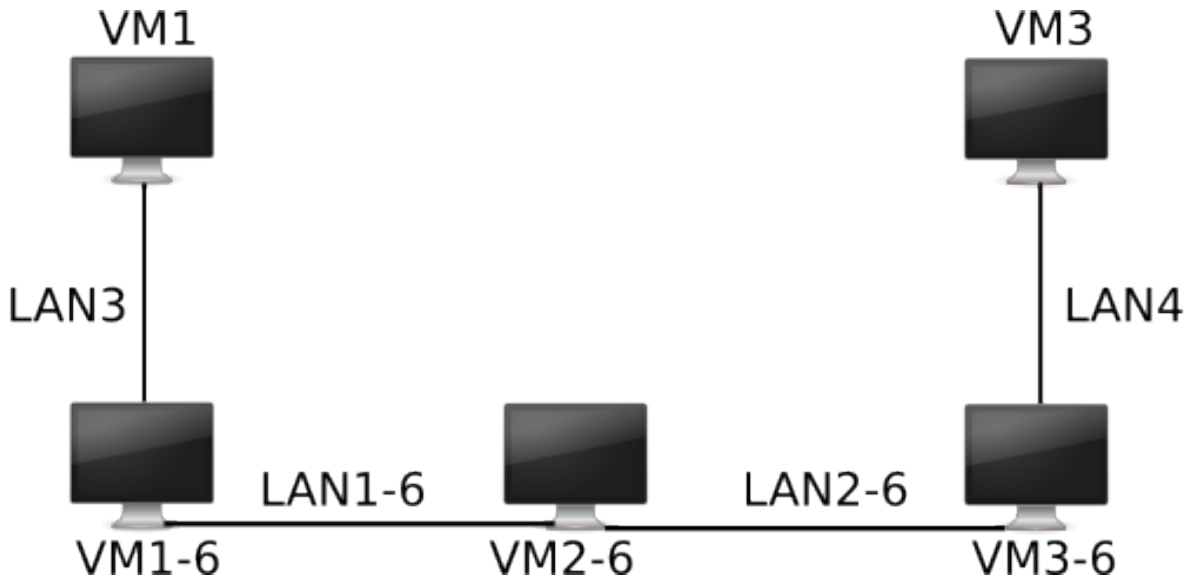
Brice Maussang
Nouman Mokadader
Jeremy Topalian

Compte rendu du projet réseaux 2018/19

1 Situation Initiale

Parc :



Plan d'adressage :

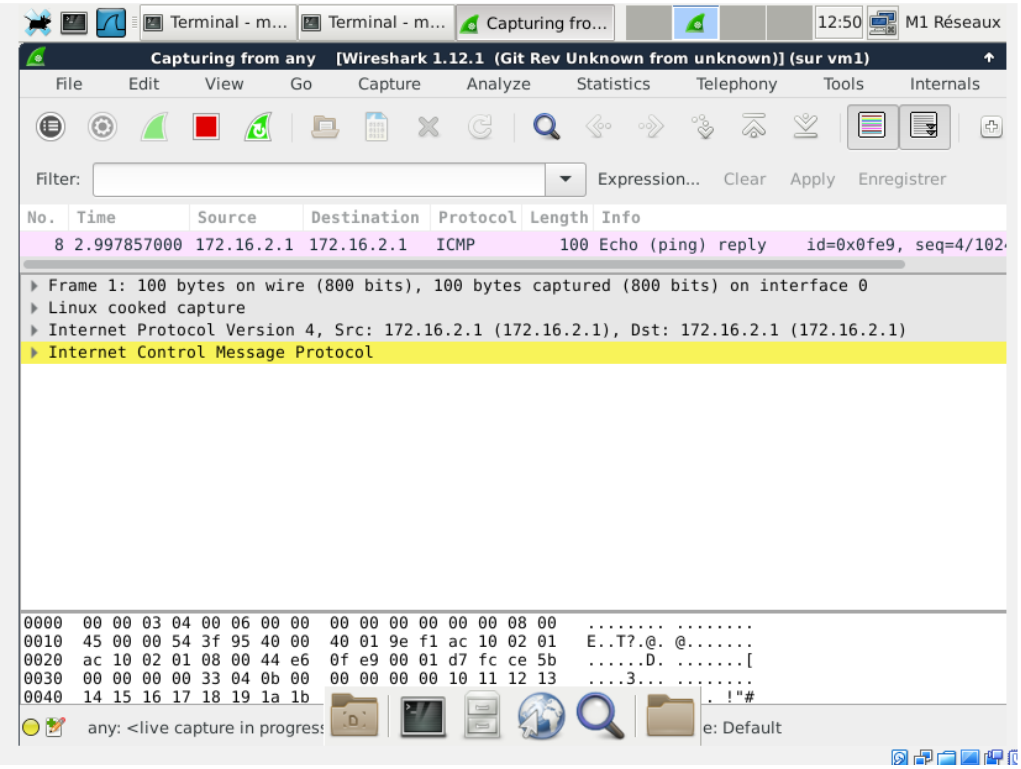
	LAN1	LAN2	LAN3	LAN4	LAN1-6	LAN2-6
réseau	172.16.2.128/28	172.16.2.160/28	172.16.2.144/28	172.16.2.176/28	fc00:1234:1::/64	fc00:1234:2::/64
VM1	172.16.2.131		172.16.2.151			
VM2	172.16.2.132	172.16.2.162				
VM3		172.16.2.163		172.16.2.183		
VM1-6			172.16.2.156		*auto*	
VM2-6					fc00:1234:1::26	fc00:1234:2::26
VM3-6				172.16.2.186		*auto*

2 L'interface virtuel TUN

2.1 Configuration de l'interface

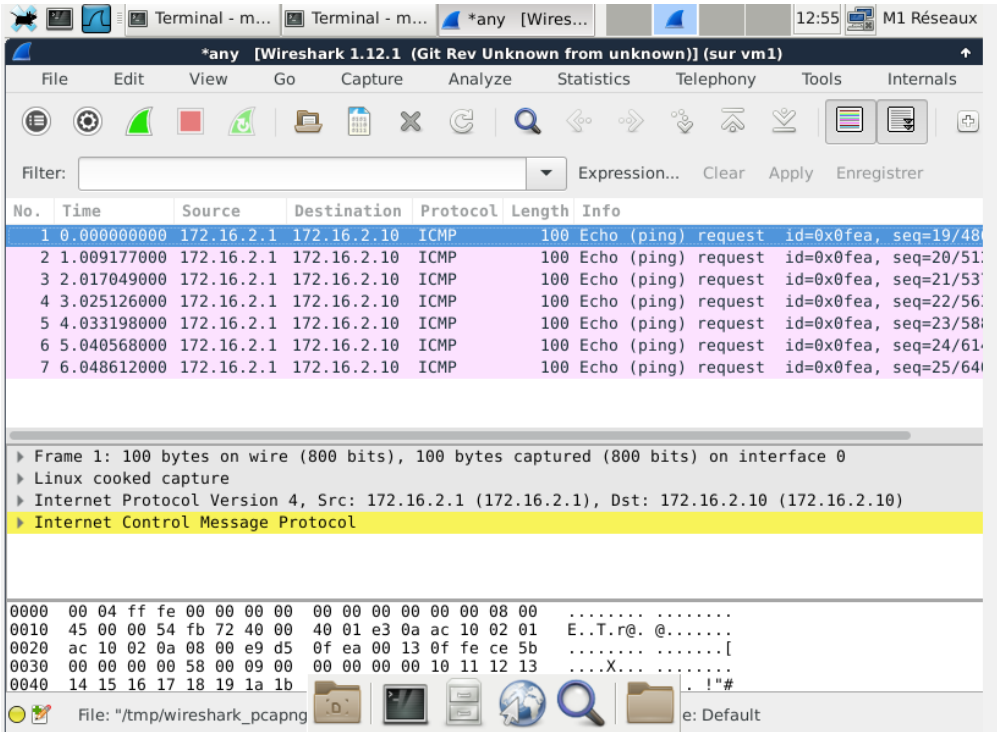
L'interface *tun0* est configuré avec l'adresse **172.16.2.1/28**. VM2 ayant disparu, il faut **retirer la route LAN1 de VM1**.

Émission d'un *ping* vers 172.16.2.1 :

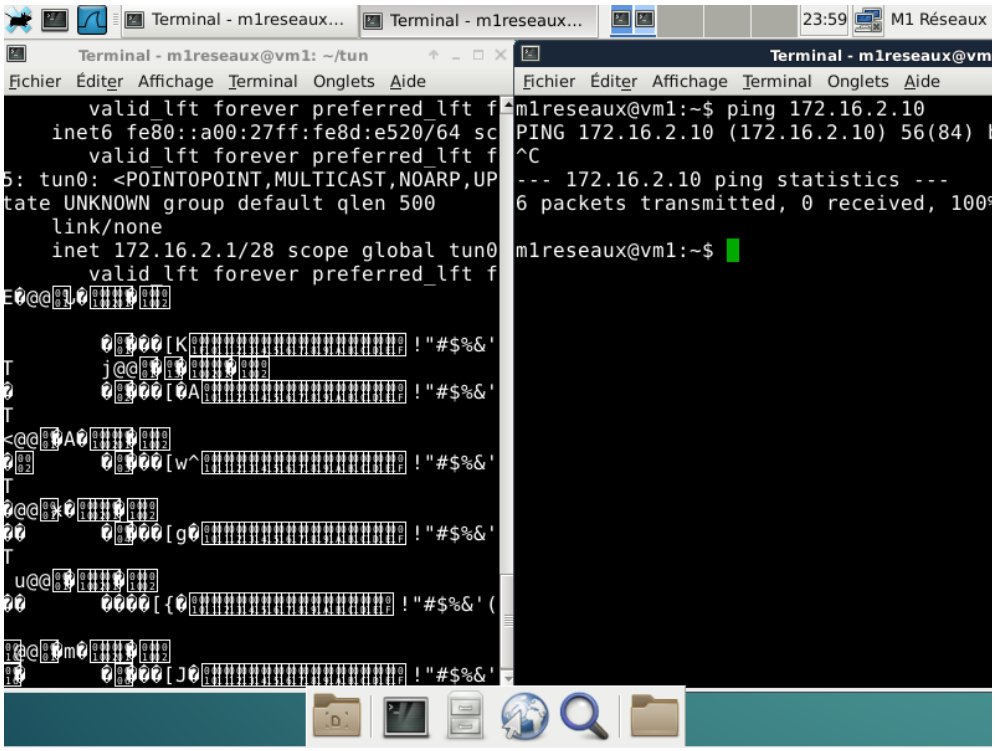


On constate que le *ping* fonctionne normalement, on émet et reçoit depuis 172.16.2.1.

Émission d'un *ping* vers 172.16.2.10 depuis 172.16.2.1 :



Le *ping* ne fonctionne pas, aucune réponse n'est renvoyée. Cependant la requête s'affiche dans le descripteur de fichier de *tun0*.

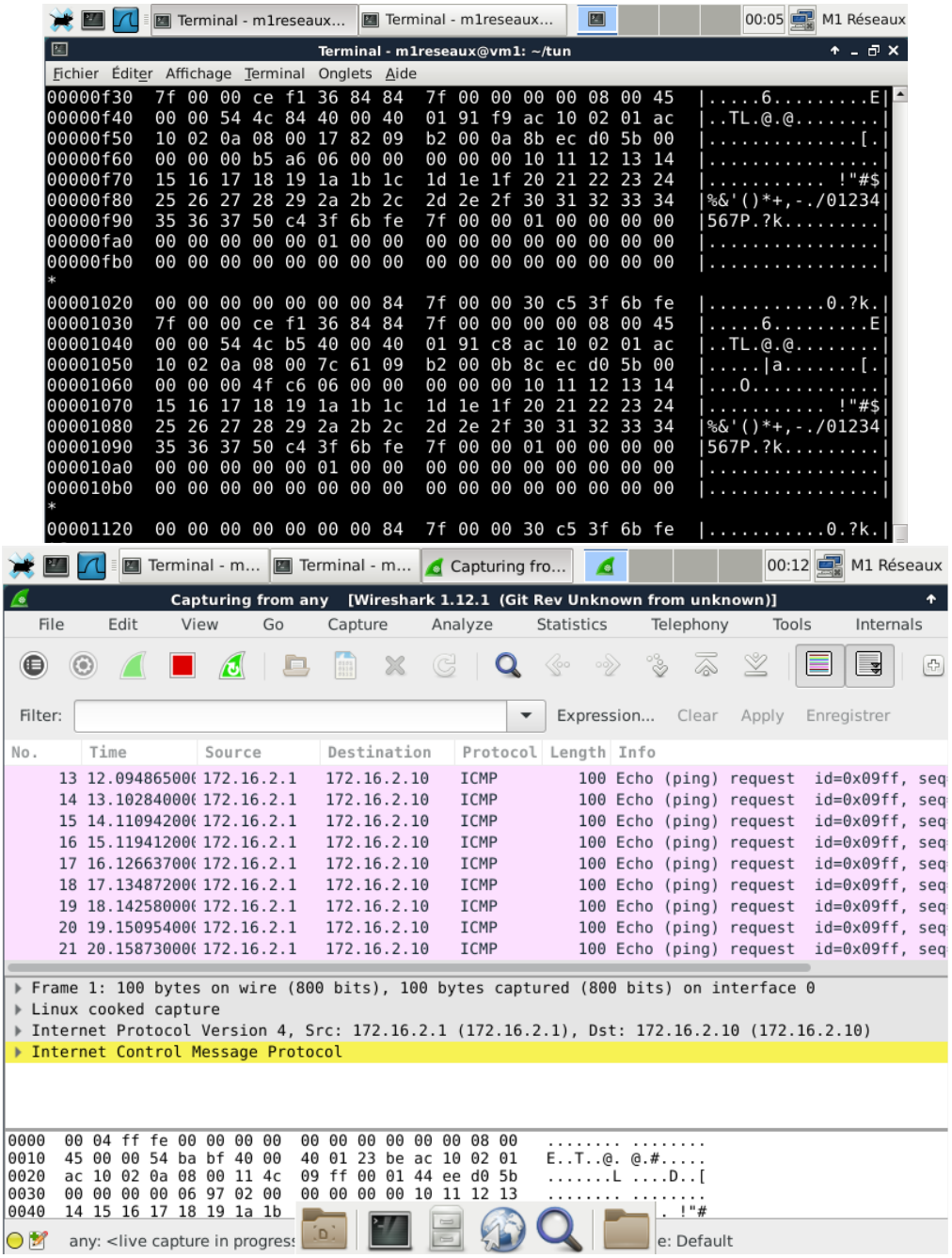


Le réseau *tun0* ne répond pas aux *ping* autres que 172.16.2.1 (son adresse donnée en configuration) mais il reçoit bien ces requêtes sur son descripteur de fichier.

2.2 Récupération des paquets

Après modification de la bibliothèque *iftun*, *ping* 172.16.2.1 n'affiche pas de nouvelles informations sur le descripteur de fichier.

Un ping 172.16.2.10 nous donne ce résultat :



D'après la documentation, si le flag IFF_NO_PI est activé on ne donne plus d'information sur le paquet dans l'en-tête. Si le flag n'est pas présent alors on a comme format de trame :
2 octets pour les flags et 2 octets pour le protocole dans l'en-tête de la trame.

3 Un tunnel simple pour Ipv4

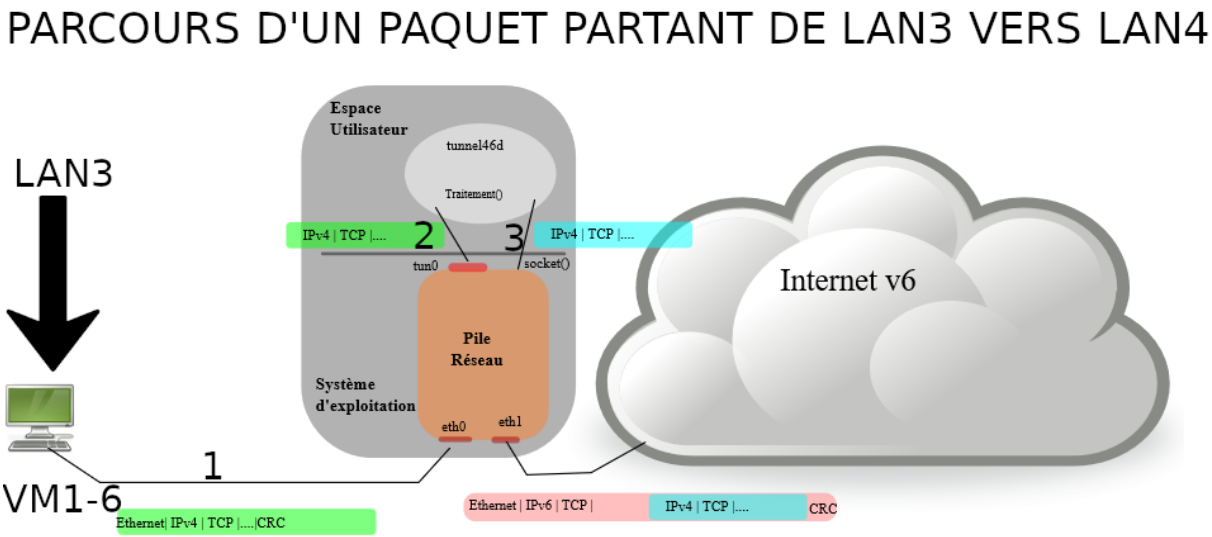
3.1 Redirection du trafic sortant

Après création de la bibliothèque *extremite*. Lorsque nous écrivons dans le descripteur du tunnel depuis VM1-6, le trafic est reçu puis écrit dans le tunnel de VM3-6.
Pour tester la connectivité, nous pouvons vérifier le contenu des trames envoyé par le tunnel à l'envoi et à la réception à l'aide de Wireshark.

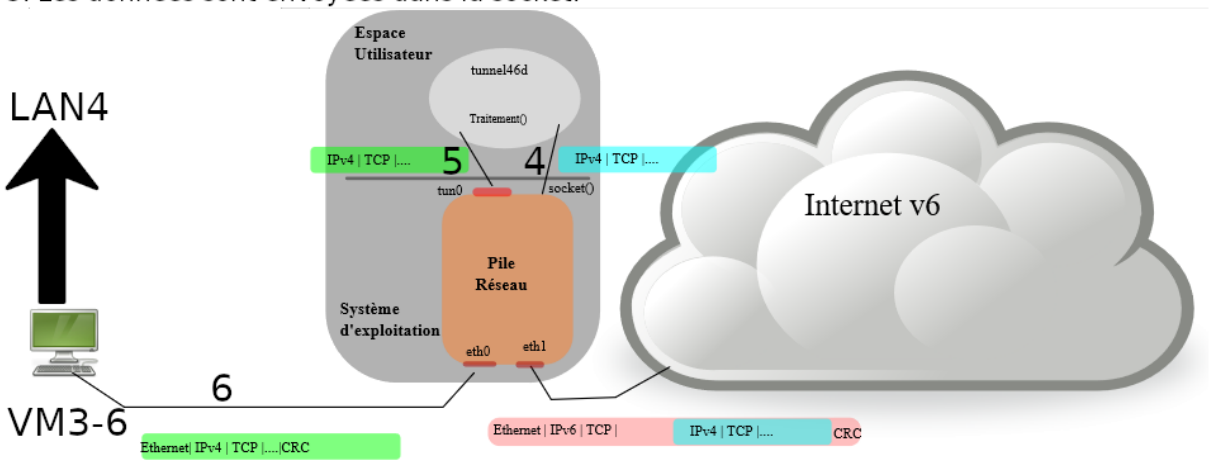
3.2 Intégration Finale du Tunnel

La bibliothèque *extremite* autorise désormais le flux bidirectionnel. Les communications sont asynchrones puisque VM1 et VM3 peuvent s'envoyer des *ping* mutuellement et simultanément.

3.3 Mise en place du tunnel entre VM1-6 et VM3-6 : Schémas



1. Envoi d'un paquet vers LAN4 via la route 172.16.2.10 (tun0).
2. Le client lit sur le tunnel tun0 les données à envoyer.
3. Les données sont envoyées dans la socket.



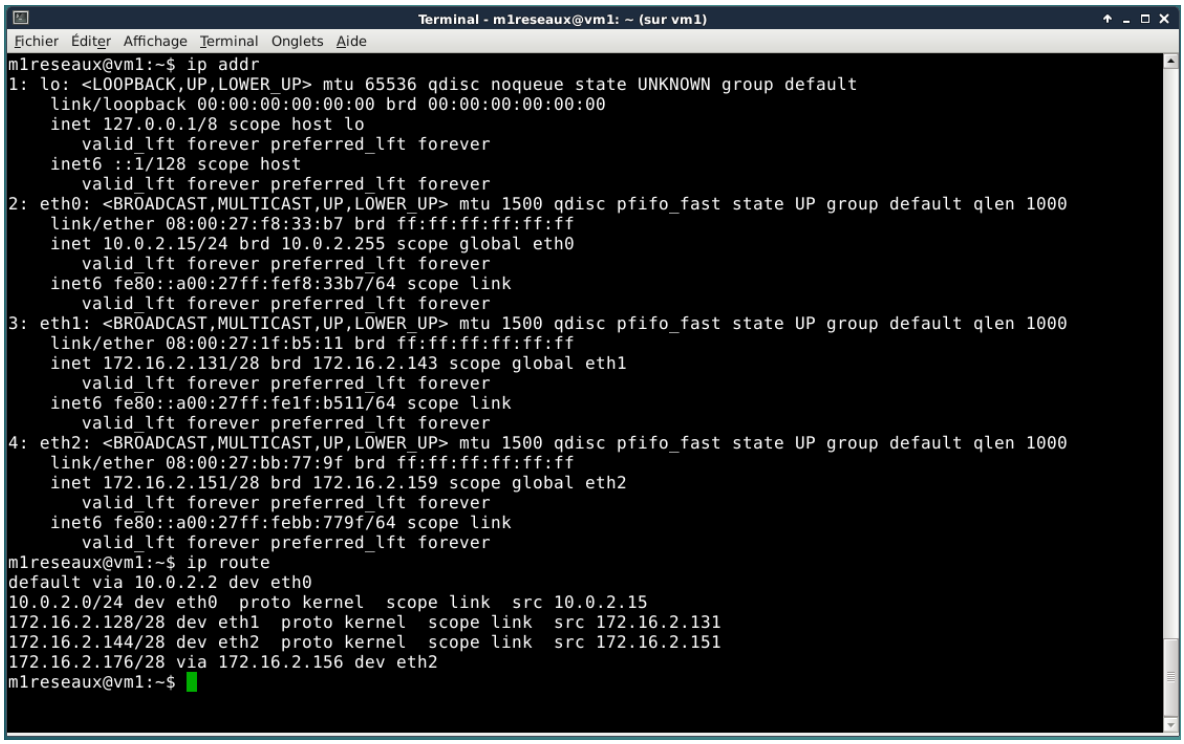
4. Le serveur lit depuis la socket les données reçues.
5. Les données sont écrites dans tun0.
6. Le paquet reçu est transmis vers LAN4.

4 Validation fonctionnelle

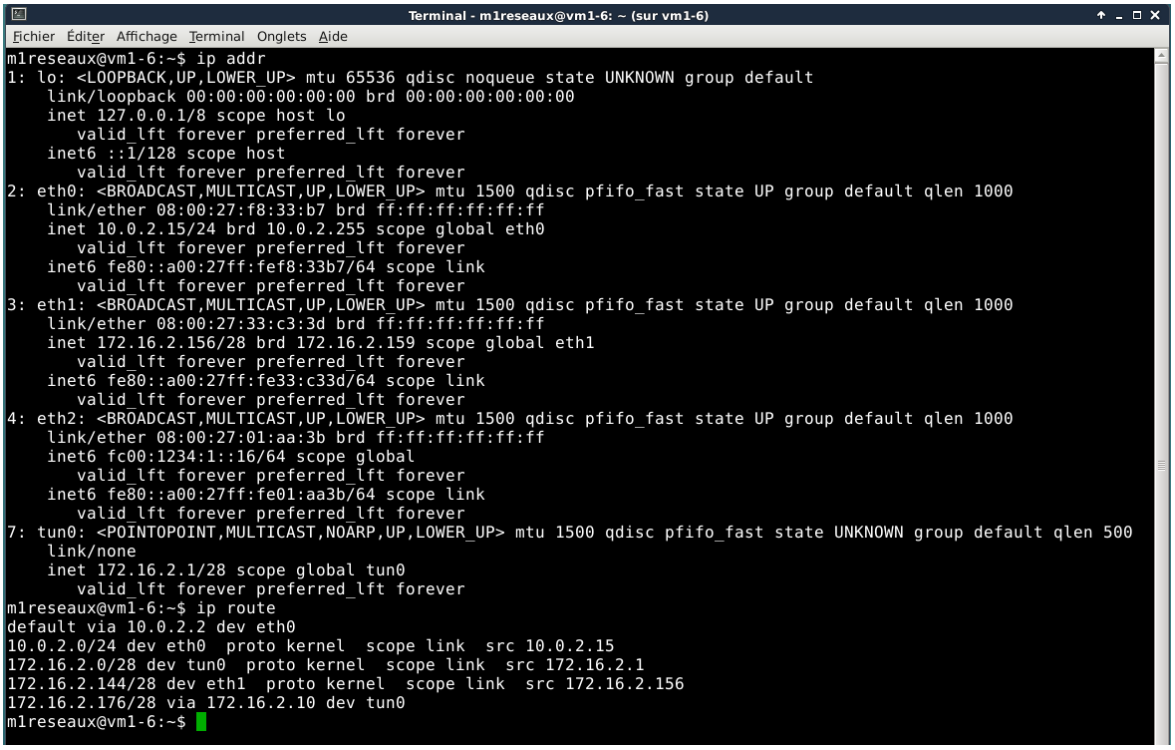
4.1 Configuration

Exécution de *ip addr* et *ip route* sur :

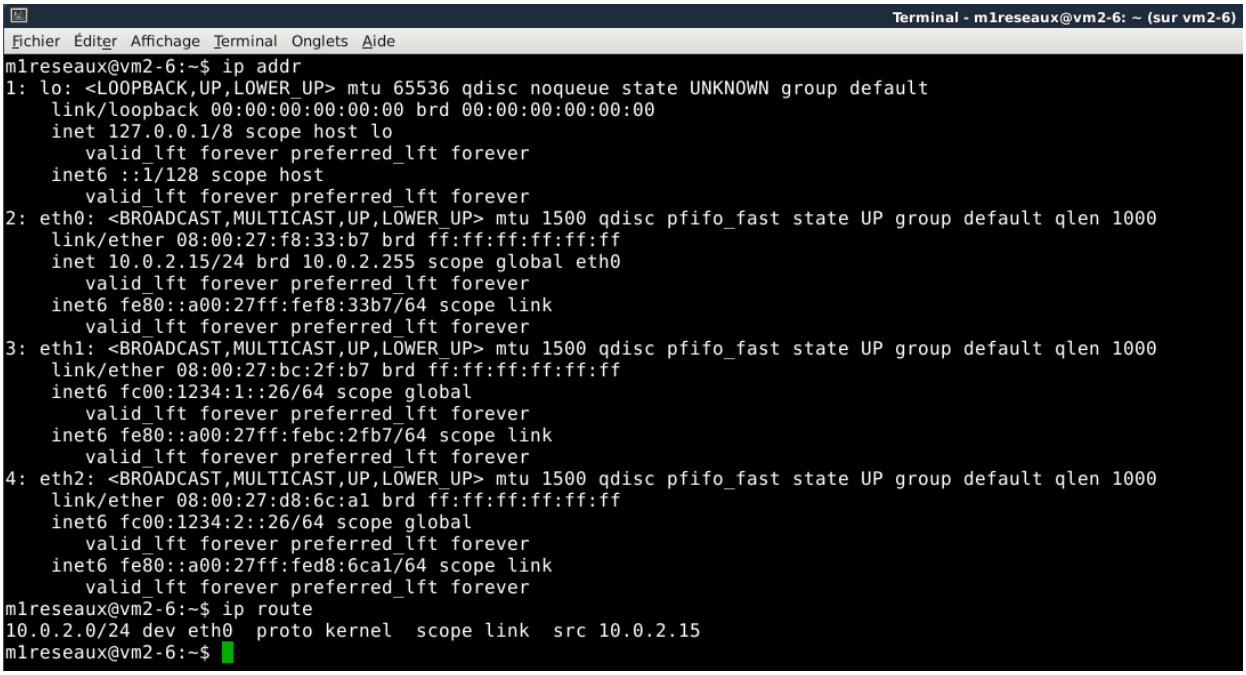
- VM1 :



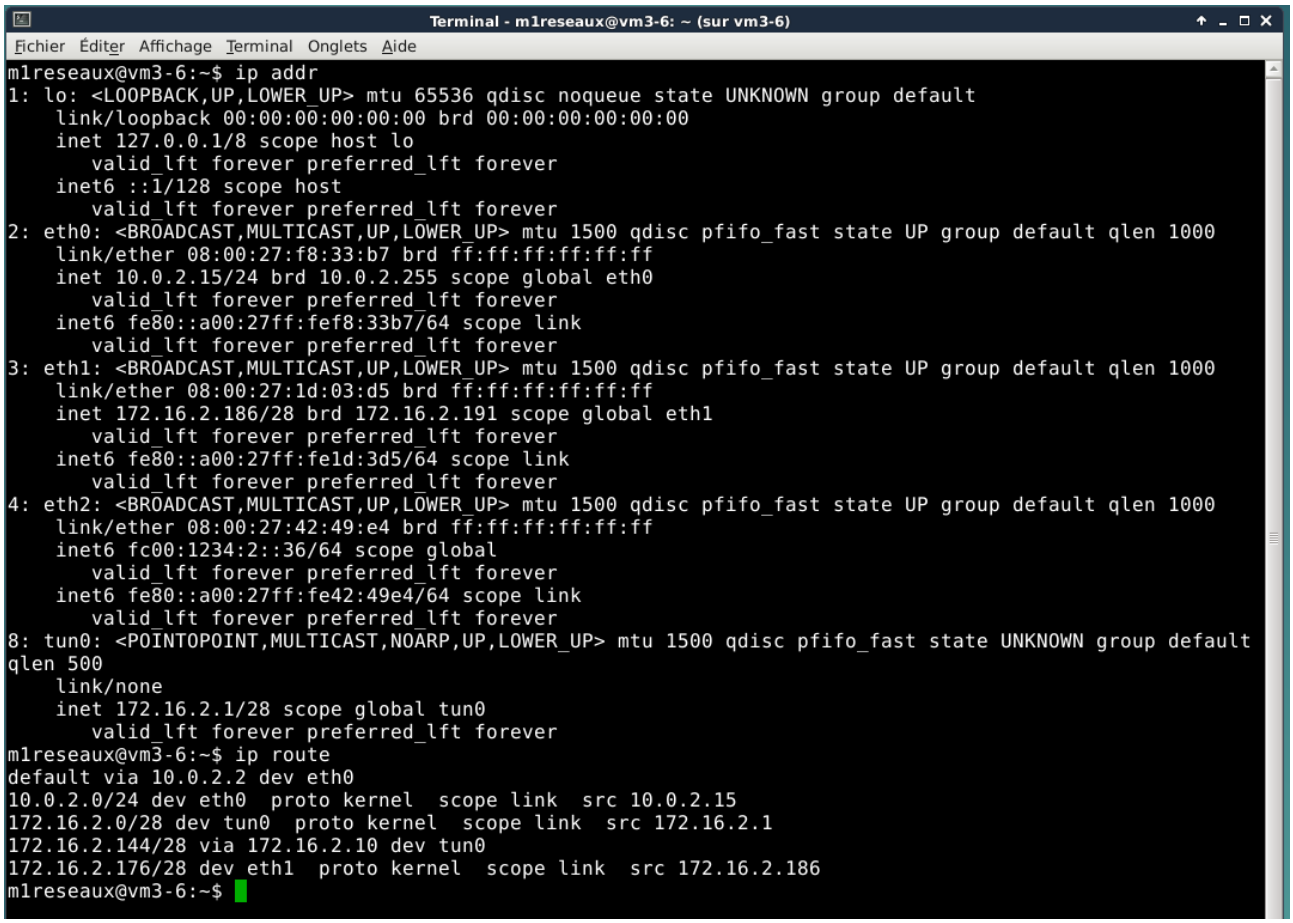
- VM1-6 :



- VM2-6 :

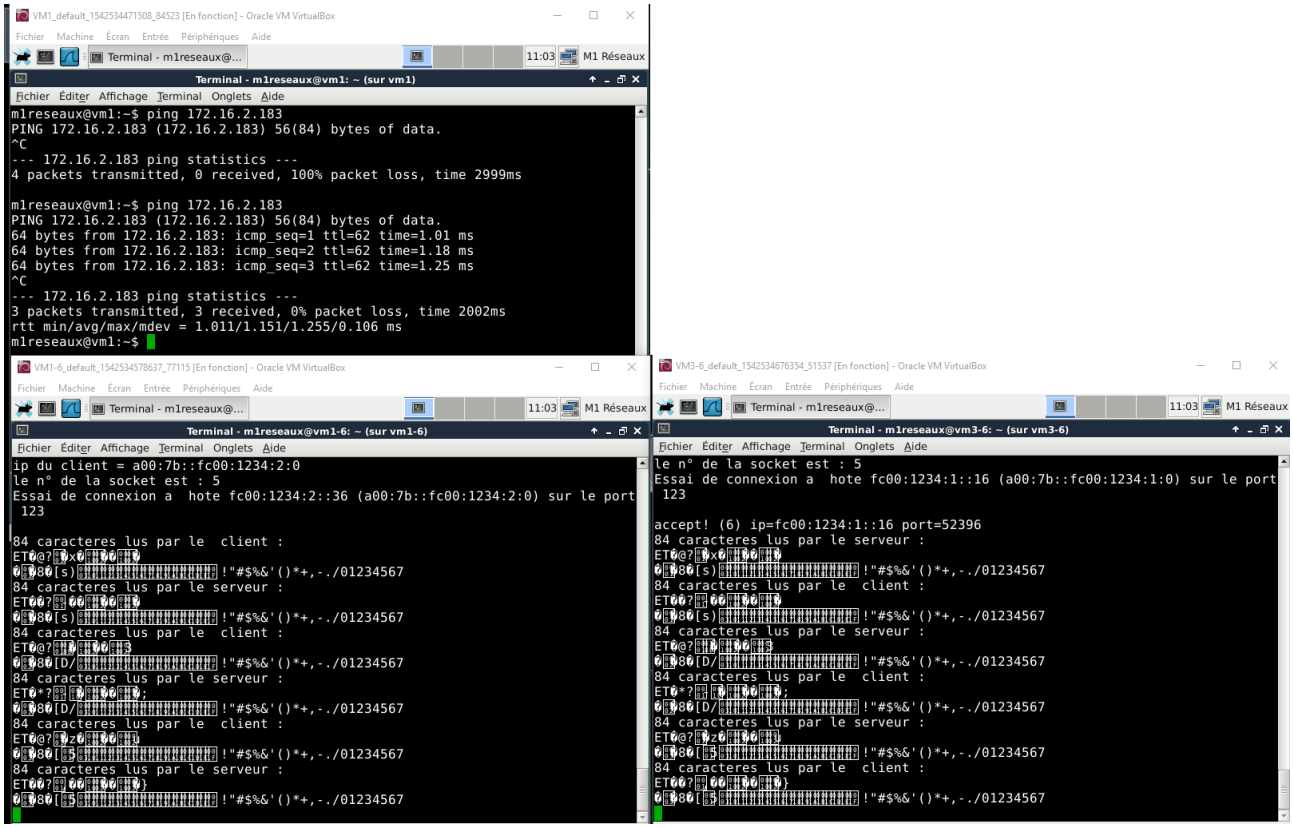

Terminal - m1reseaux@vm2-6: ~ (sur vm2-6)
Fichier Éditer Affichage Terminal Onglets Aide
m1reseaux@vm2-6:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 08:00:27:f8:33:b7 brd ff:ff:ff:ff:ff:ff
inet 10.0.2.15/24 brd 10.0.2.255 scope global eth0
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fef8:33b7/64 scope link
valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 08:00:27:bc:2f:b7 brd ff:ff:ff:ff:ff:ff
inet6 fc00:1234:1::26/64 scope global
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:febc:2fb7/64 scope link
valid_lft forever preferred_lft forever
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 08:00:27:d8:6c:a1 brd ff:ff:ff:ff:ff:ff
inet6 fc00:1234:2::26/64 scope global
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fed8:6ca1/64 scope link
valid_lft forever preferred_lft forever
m1reseaux@vm2-6:~$ ip route
10.0.2.0/24 dev eth0 proto kernel scope link src 10.0.2.15
m1reseaux@vm2-6:~$

- VM3-6 :


Terminal - m1reseaux@vm3-6: ~ (sur vm3-6)
Fichier Éditer Affichage Terminal Onglets Aide
m1reseaux@vm3-6:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 08:00:27:f8:33:b7 brd ff:ff:ff:ff:ff:ff
inet 10.0.2.15/24 brd 10.0.2.255 scope global eth0
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fef8:33b7/64 scope link
valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 08:00:27:1d:03:d5 brd ff:ff:ff:ff:ff:ff
inet 172.16.2.186/28 brd 172.16.2.191 scope global eth1
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fe1d:3d5/64 scope link
valid_lft forever preferred_lft forever
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 08:00:27:42:49:e4 brd ff:ff:ff:ff:ff:ff
inet6 fc00:1234:2::36/64 scope global
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fe42:49e4/64 scope link
valid_lft forever preferred_lft forever
8: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group default
qlen 500
link/none
inet 172.16.2.1/28 scope global tun0
valid_lft forever preferred_lft forever
m1reseaux@vm3-6:~$ ip route
default via 10.0.2.2 dev eth0
10.0.2.0/24 dev eth0 proto kernel scope link src 10.0.2.15
172.16.2.0/28 dev tun0 proto kernel scope link src 172.16.2.1
172.16.2.144/28 via 172.16.2.10 dev tun0
172.16.2.176/28 dev eth1 proto kernel scope link src 172.16.2.186
m1reseaux@vm3-6:~$

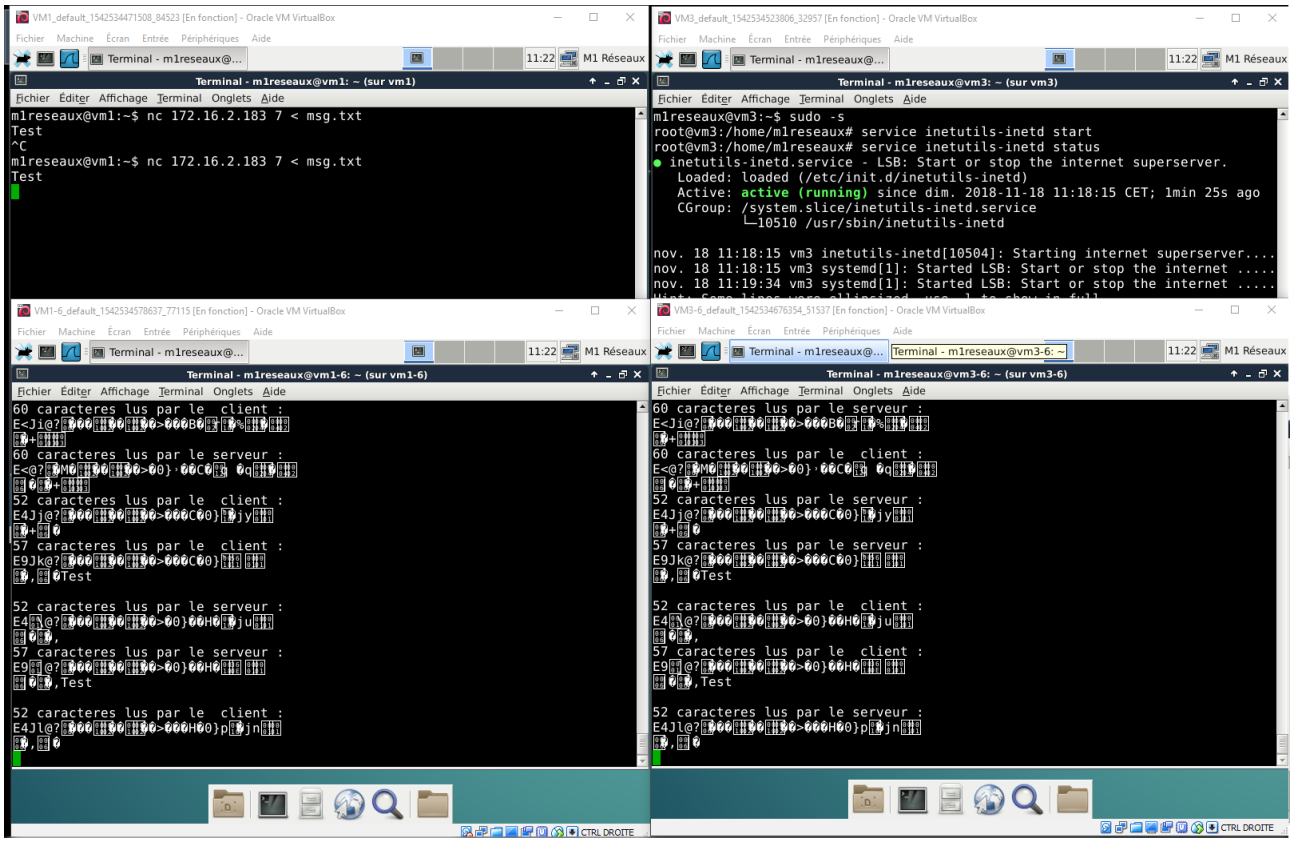
4.2 Couche 3

Exécution de *ping* 172.16.2.183 sur VM1 :



4.3 Couche 4

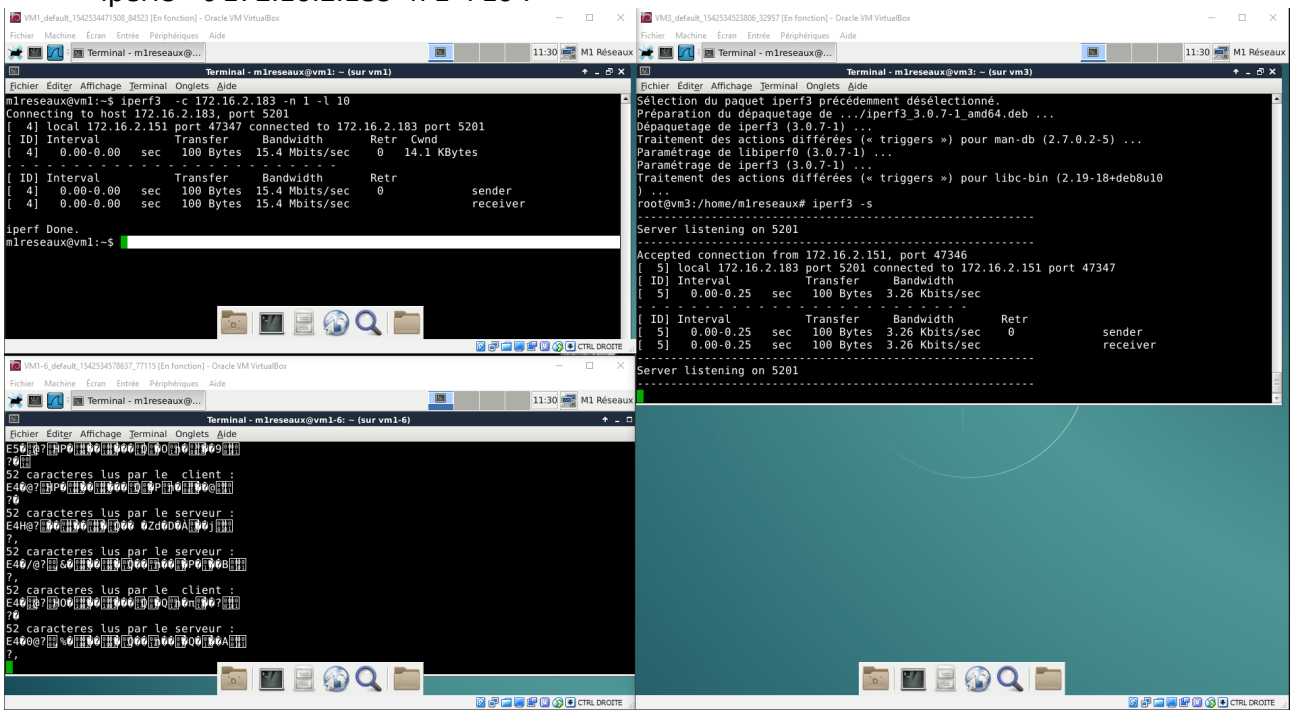
Création depuis VM1 du fichier *msg.txt* et envoie sur le service *echo* de VM3 :



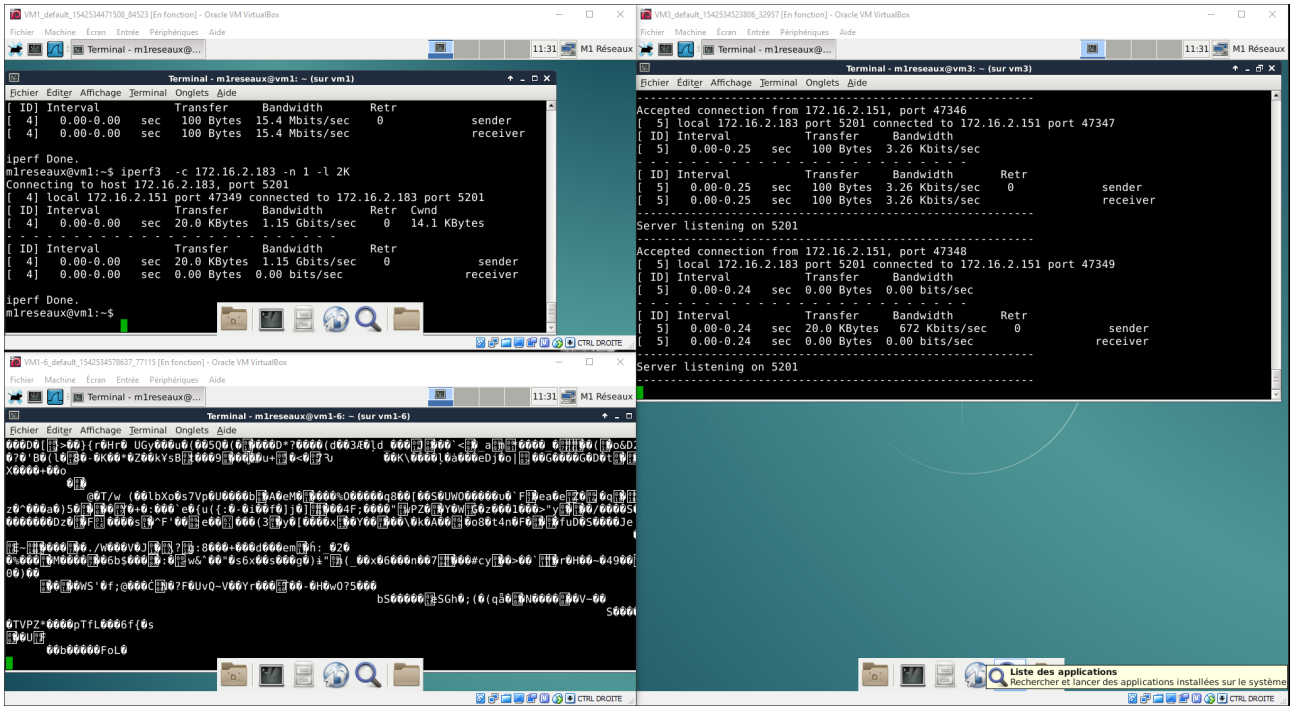
4.4 Couche 4 : bande passante

Voici les tests de performances effectués pour différents tampons :

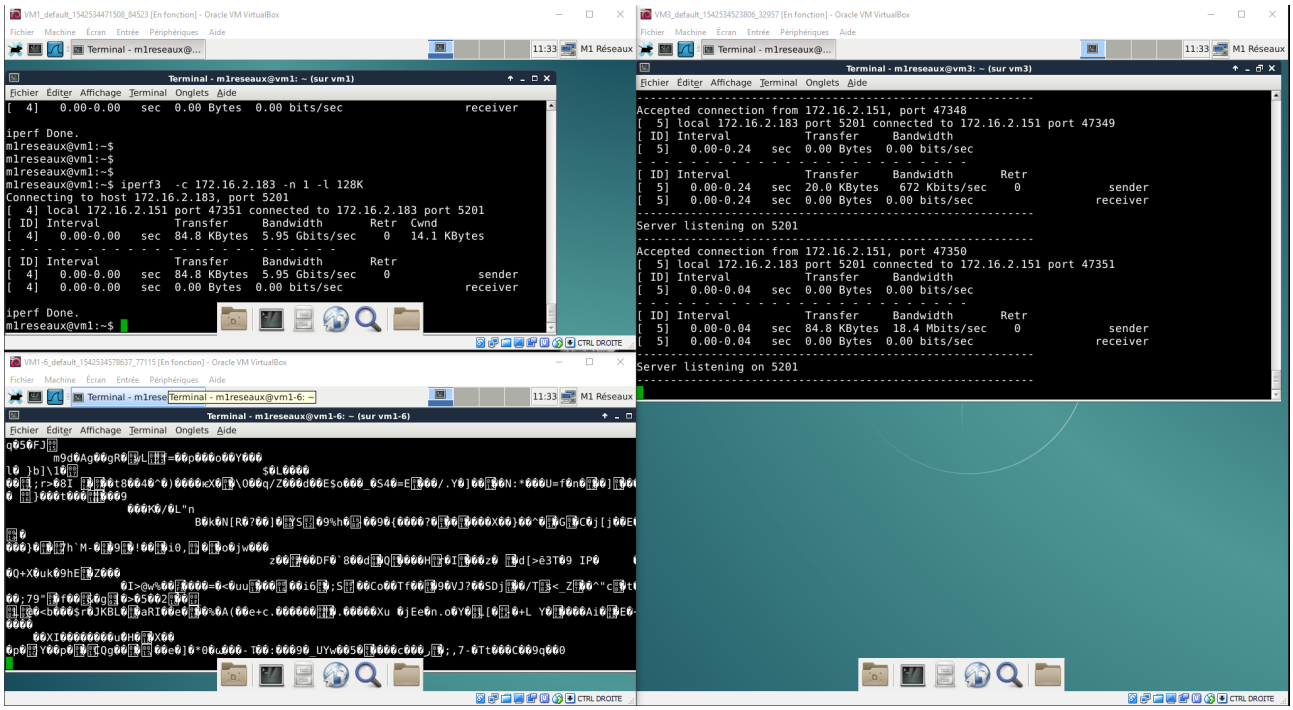
- iperf3 -c 172.16.2.183 -n 1 -l 10 :



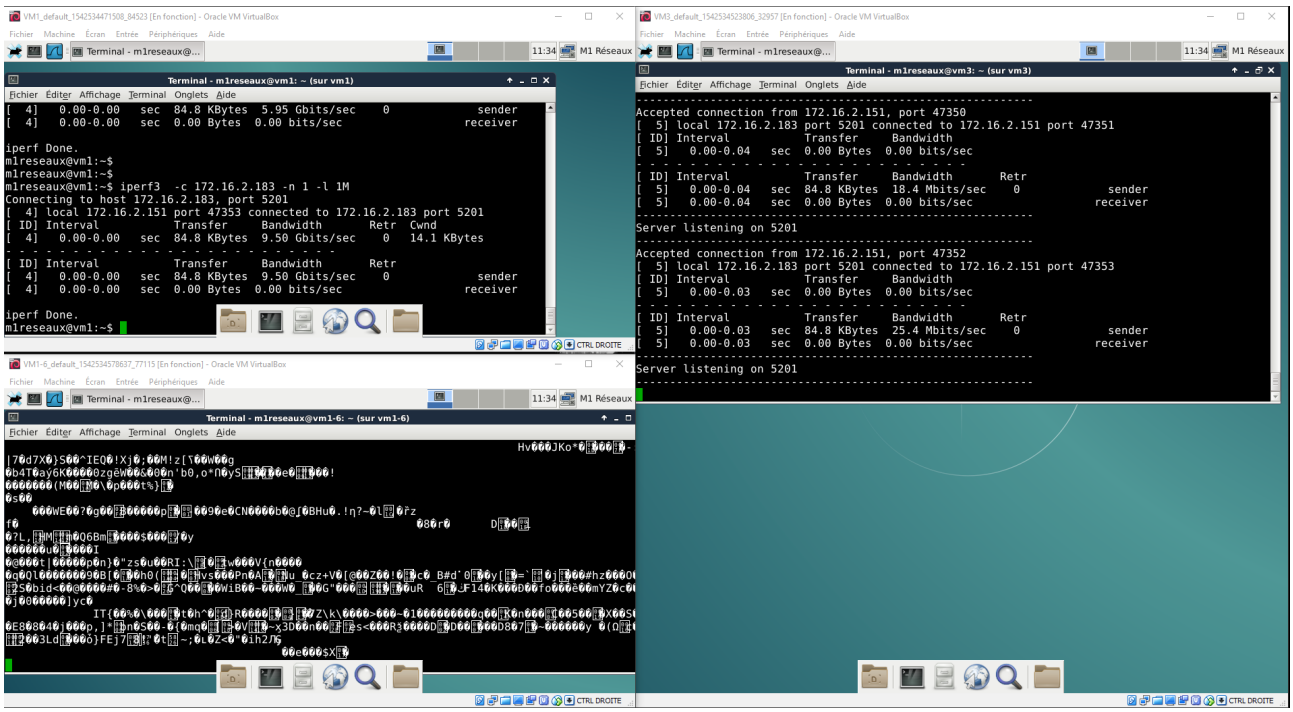
- iperf3 -c 172.16.2.183 -n 1 -l 2K :



- iperf3 -c 172.16.2.183 -n 1 -l 128K :



- iperf3 -c 172.16.2.183 -n 1 -l 1M :



5 Améliorations

5.1 Configuration *salt*

En lançant notre programme *extremite* sur les deux VMs (VM1-6 et VM3-6) via *salt* il est possible de mettre en place une gestion de tunnel. Les difficultés peuvent être le manque de portabilité du programme. En effet, la création des routes dépend du parc et donc ne peut pas être automatisé de la même manière.