



ECOLE SUPÉRIEURE DE RECHERCHE  
EN MATÉRIAUX ET EN INFOTRONIQUE

ITC312 INTRODUCTION AUX RÉSEAUX

---

**Rapport final des TP<sub>s</sub> de réseau**

---

*Soumis par :*  
Wilfried L. Bounsi  
Ulrich Fonkoue  
IT3A TD2 TP3

Année académique 2018/2019

# Table des matières

<b>1 TP1 : Installation et découverte</b>	<b>2</b>
1.1 Objectif du TP . . . . .	2
1.2 Installation et prise en main . . . . .	2
1.2.1 Installation du système . . . . .	2
1.2.2 Découverte du système . . . . .	2
1.3 Découverte de l'environnement réseau . . . . .	3
1.3.1 Interface réseau . . . . .	3
1.3.2 Etat du réseau et connexions actives . . . . .	5
1.3.3 Politique de routage . . . . .	8
1.3.4 Résolution des noms de domaine . . . . .	9
1.4 Manipulation avancées . . . . .	10
1.4.1 Etude d'une trame ethernet . . . . .	10
1.4.2 La commande traceroute . . . . .	13
<b>2 TP2 : Interconnexion des réseaux</b>	<b>15</b>
2.1 Objectif du TP . . . . .	15
2.2 Prerequisites . . . . .	15
2.2.1 Droits administrateur . . . . .	15
2.2.2 Définition du plan d'adressage . . . . .	16
2.3 Mise en place du réseau . . . . .	16
2.3.1 Connexion physique . . . . .	16
2.3.2 Configuration logique . . . . .	16
2.3.3 Connexion internet des postes clients . . . . .	18
2.3.4 Sauvegarde de la configuration . . . . .	20
<b>3 TP3 : Configuration de serveurs</b>	<b>22</b>
3.1 Objectif du TP . . . . .	22
3.2 Mise en place d'un serveur DHCP . . . . .	23
3.2.1 Comprendre le DHCP . . . . .	23
3.2.2 Définition des paramètres du serveur DHCP . . . . .	23
3.2.3 Mise en place du serveur DHCP . . . . .	24
3.3 Mise en place d'un serveur Web + SQL . . . . .	27
3.3.1 Installation des packages . . . . .	27
3.3.2 Mise en place du serveur Apache2 . . . . .	27
3.3.3 Mise en place du serveur MySQL . . . . .	28
<b>4 TP4 : Firewall</b>	<b>30</b>
4.1 Objectif du TP . . . . .	30
4.2 Exercice 1 . . . . .	30
4.3 Exercice 2 . . . . .	31
4.3.1 Vérification de la table de iptables . . . . .	31
4.4 Exercice 3 . . . . .	33

# Chapitre 1

## TP1 : Installation et découverte

### 1.1 Objectif du TP

Ce TP a pour but de nous initier à la connaissance et à la gestion de l'environnement réseau sous Linux. Nous commençerons par installer sur nos machines des systèmes Linux, puis nous verrons et utiliserons des commandes classiques d'administration réseau sur notre système. Nous analyserons enfin le trafic du réseau grâce à l'outil de capture de traffic wireshark.

### 1.2 Installation et prise en main

#### 1.2.1 Installation du système

La réalisation des TPs exige de disposer des droits administrateurs sur un système Linux. Nous utiliserons donc conjointement une clé live USB Linux, mais également un disque dur externe, remis par l'encadrant afin de pouvoir sauvegarder les modifications de la configuration et de conserver le fil de notre évolution tout au long des quatre séances.

C'est dans cette optique que nous avons donc installé sur le disque dur externe qui nous a été temporairement attribué, un système Linux Ubuntu 16.04 LTS à partir de la clé live.

#### 1.2.2 Découverte du système

##### Droits d'administrateurs

Pour lancer un terminal avec les droits d'administrateur (utilisateur root), on ouvre un terminal et on utilise l'une des commandes suivantes. Le terminal aura alors les droits root. Nous détaillons également les différences dans l'environnement selon la commande utilisée.

```
1 sudo su # connexion en tant que root
2 sudo -i # connexion en tant que root et changement de répertoire au home du root
3 sudo -s # connexion en tant que root et changement du shell pour celui en argument.
```

**NB :** On peut également juste faire un

```
1 sudo nom-commande
```

pour exécuter la commande `nom-commande` en tant que root sans changer ni le répertoire courant, ni le shell.

##### Gestionnaire de packages

Le gestionnaire de packages permet d'installer des outils complémentaires sur le système. Il peut être appelé dans sa version graphique, ou dans sa version console. En mode console, la recherche d'un package à partir de son nom ou d'un mot clé se fait avec la commande suivante :

```
1 apt-cache search mot-clé
```

L'installation d'un paquet requiert les droits administrateurs et s'effectue avec la commande suivante :

```
1 sudo apt-get install nom-paquet
```

Cette installation peut également se faire avec les outils graphiques présents dans la distribution de linux.

## 1.3 Découverte de l'environnement réseau

### 1.3.1 Interface réseau

1. Nous exécutons et notons le résultat des commandes suivantes :

— **ifconfig**

Cette commande affiche la liste des interfaces réseaux liées à l'ordinateur et leurs caractéristiques, protocole de couche 2, adresse IP, adresse Mac. Nous avons obtenu la sortie suivante :

```
root@interface:~# ifconfig
enp0s31f6 Link encap:Ethernet HWaddr 18:66:da:45:bc:df
      inet adr:172.24.43.47 Bcast:172.24.127.255 Masque:255.255.128.0
          adr inét: fe80::896b:e419:b42c:e9ac/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          Packets reçus:79329 erreurs:0 :0 overrunns:0 frame:0
          TX packets:33212 errors:0 dropped:0 overrunns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:88754037 (88.7 MB) Octets transmis:3444605 (3.4 MB)
          Interruption:16 Mémoire:f7f00000-f7f20000

enp3s2   Link encap:Ethernet HWaddr 00:0a:cd:2c:31:1c
          UP BROADCAST MULTICAST MTU:1500 Metric:1
          Packets reçus:0 erreurs:0 :0 overrunns:0 frame:0
          TX packets:0 errors:0 dropped:0 overrunns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:0 (0.0 B) Octets transmis:0 (0.0 B)

enp4s0   Link encap:Ethernet HWaddr 00:13:3b:10:66:8e
          UP BROADCAST MULTICAST MTU:1500 Metric:1
          Packets reçus:0 erreurs:0 :0 overrunns:0 frame:0
          TX packets:0 errors:0 dropped:0 overrunns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:0 (0.0 B) Octets transmis:0 (0.0 B)

lo      Link encap:Boucle locale
          inet adr:127.0.0.1 Masque:255.0.0.0
              adr inét: ::1/128 Scope:Hôte
              UP LOOPBACK RUNNING MTU:65536 Metric:1
              Packets reçus:2178 erreurs:0 :0 overrunns:0 frame:0
              TX packets:2178 errors:0 dropped:0 overrunns:0 carrier:0
              collisions:0 lg file transmission:1000
              Octets reçus:407204 (407.2 KB) Octets transmis:407204 (407.2 KB)

root@interface:~#
```

FIGURE 1.1 – résultatat ifconfig

— **ifconfig enp0s31f6**

Cette commande affiche les détails de l'interface réseau ayant pour nom logique enp0s31f6, entre autres :

- Protocole de couche 2 : Ethernet
- Adresse IP : 172.24.43.27
- Adresse Mac : 18 :66 :da :45 :bc :df

```
root@interface:~# ifconfig enp0s31f6
enp0s31f6 Link encap:Ethernet HWaddr 18:66:da:45:bc:df
      inet adr:172.24.43.47 Bcast:172.24.127.255 Masque:255.255.128.0
          adr inét: fe80::896b:e419:b42c:e9ac/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          Packets reçus:84960 erreurs:0 :0 overrunns:0 frame:0
          TX packets:34718 errors:0 dropped:0 overrunns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:89537351 (89.5 MB) Octets transmis:3900256 (3.9 MB)
          Interruption:16 Mémoire:f7f00000-f7f20000
```

FIGURE 1.2 – résultatat ifconfig enp0s31f6

## 2. Relevé des informations sur les ports ethernet

Interface	Adr Mac	Adr IP	Masque SR	Adr Réseau	Statut
enp0s31f6	18:66:de:45:bc:df	172.24.43.47	255.255.128.0	172.24.0.0	UP BROADCAST RUNNING MULTICAST
enp3s2	00:0a:cd:2c:31:1c	Néant	Néant	Néant	UP BROADCAST MULTICAST
enp4s2	00:13:3b:10:66:8e	Néant	Néant	Néant	UP BROADCAST MULTICAST

TABLE 1.1 – relevé des informations sur les ports

3. Détermination de la vitesse de transfert de chaque interface On utilise pour cela la commande **ethtool** et on obtient le tableau suivant :

Interface	Vitesse de transfert
enp0s31f6	100Mb/s
enp3s2	10Mb/s
enp4s0	10Mb/s

TABLE 1.2 – vitesse de transfert des interfaces

- 4. — L'interface qui correspond à la carte réseau connectée au commutateur de la salle de TP est l'interface **enp0s31f6** car c'est la seule disposant d'une adresse IP.
- En effectuant un **ET logique** entre l'adresse IP et le masque de sous réseau on obtient **172.24.0.0**.
- C'est bien l'adresse réseau des autres machines du groupe.
- 5. Exploration du manuel pour réaliser les opérations suivantes :
  - Changement de l'adresse IP :

```
1 sudo ifconfig enp0s31f6 172.24.47.43/17
```

  - Changement de l'adresse Mac :

```
1 sudo ifconfig enp0s31f6 hw ether 18:60:db:45:bc:df
```

  - Désactivation de l'interface :

```
1 sudo ifconfig enp0s31f6 down
```

  - Restauration de l'adresse Mac originale :

```
1 sudo ifconfig enp0s31f6 hw ether $(ethtool -P eth0 | awk '{print $3}')
```

### 1.3.2 Etat du réseau et connexions actives

La commande ping permet de "pinger" une machine, c'est à dire de lui demander si elle est active et reçoit bien les paquets. Si l'hôte est actif, il répond en général en renvoyant le paquet envoyé. Le temps de transfert entre l'envoie du paquet et la réception du message est affiché à l'écran. Notons ce pendant qu'un hôte peut avoir désactiver la fonction de réponse mais être quand-même actif.

1. Nous Utilisons la commande ping pour évaluer la connectivité des différentes machines de notre réseau.

Nous présentons tout d'abord ci-dessous, le résultat obtenu en faisant un ping de 10 paquets vers le serveur dns de google d'adresse IP 8.8.8.8

```
bounkoue@interface:~$ ping 8.8.8.8 -c 10
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=8.11 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=8.34 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=117 time=8.30 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=117 time=8.19 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=117 time=8.35 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=117 time=8.15 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=117 time=8.26 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=117 time=8.32 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=117 time=8.33 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=117 time=8.22 ms

--- 8.8.8.8 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9013ms
rtt min/avg/max/mdev = 8.112/8.260/8.350/0.081 ms
```

FIGURE 1.3 – ping vers google

La commande netstat est une commande importante qui permet de voir l'état des connexions actives sur la machine, que ces connexions soient gérées par des socket réseau ou par des sockets locales. Nous générerons un peu d'activité réseau et puis nous testons les différentes commandes suivantes et notons leur résultat :

#### — netstat

Affiche les connexions réseau, les tables de routage, les statistiques des interfaces, les connexions masquées, les messages netlink, et les membres multicast.

```
bounkoue@interface:~$ netstat
Connexions Internet actives (sans serveurs)
Proto Recv-Q Send-Q Adresse locale      Adresse distante      Etat
tcp     0      0 gr19-13.estrem-ad:42268 ws-in-f189.1e100.:https ESTABLISHED
tcp     0      0 gr19-13.estrem-ad:48848 par10s10-in-f227.:https TIME_WAIT
tcp     0      0 gr19-13.estrem-ad:55238 stackoverflow.com:https ESTABLISHED
tcp     0      0 gr19-13.estrem-ad:48850 par10s10-in-f227.:https ESTABLISHED
tcp     0      0 gr19-13.estrem-ad:47040 han02s11-in-f46.1:https ESTABLISHED
tcp     0      0 gr19-13.estrem-ad:42270 ws-in-f189.1e100.:https ESTABLISHED
Sockets du domaine UNIX actives (sans serveurs)
Proto RefCnt Flags       Type      State      I-Node    Chemin
unix   17     [ ]      DGRAM          15955   /run/systemd/journal/dev-log
unix   2     [ ]      DGRAM          15960   /run/systemd/journal/syslog
unix   3     [ ]      DGRAM          13727   /run/systemd/notify
unix   2     [ ]      DGRAM          24886   /run/user/1000/systemd/notify
unix   7     [ ]      DGRAM          14581   /run/systemd/journal/socket
unix   3     [ ]      STREAM CONNECTE  44601   @/tmp/.X11-unix/X0
unix   3     [ ]      STREAM CONNECTE  37078   @/tmp/dbus-bYdvfr0i6q
unix   3     [ ]      STREAM CONNECTE  23274
unix   3     [ ]      STREAM CONNECTE  25020
unix   3     [ ]      STREAM CONNECTE  24991   /var/run/dbus/system_bus_socket
unix   3     [ ]      STREAM CONNECTE  24931
unix   3     [ ]      STREAM CONNECTE  25819
unix   3     [ ]      STREAM CONNECTE  26677
unix   3     [ ]      STREAM CONNECTE  24128
unix   3     [ ]      STREAM CONNECTE  42196
unix   3     [ ]      STREAM CONNECTE  20116   @/tmp/dbus-40oXWGcyDm
unix   3     [ ]      STREAM CONNECTE  22247
unix   3     [ ]      STREAM CONNECTE  23064   @/tmp/ibus/dbus-W1y1P0KY
unix   3     [ ]      STREAM CONNECTE  20153   @/tmp/dbus-bYdvfr0i6q
unix   3     [ ]      STREAM CONNECTE  25018
unix   3     [ ]      STREAM CONNECTE  22358   /var/run/dbus/system_bus_socket
unix   3     [ ]      STREAM CONNECTE  21347   @/tmp/.X11-unix/X0
unix   2     [ ]      DGRAM          24956
unix   3     [ ]      STREAM CONNECTE  26669
unix   3     [ ]      STREAM CONNECTE  24112
unix   3     [ ]      STREAM CONNECTE  22340
unix   3     [ ]      STREAM CONNECTE  21111
unix   3     [ ]      DGRAM          1643
```

FIGURE 1.4 – résultat netstat

— netstat -n

Affiche les adresses en format numérique au lieu d'essayer de déterminer le nom symbolique d'hôte, de port ou d'utilisateur.

```
bounkoue@interface:~$ netstat -n
Connexions Internet actives (sans serveurs)
Proto Recv-Q Send-Q Adresse locale           Adresse distante         Etat
tcp     0      0 172.24.43.47:36208        172.217.18.225:443 ESTABLISHED
tcp     0      0 172.24.43.47:42268        173.194.76.189:443 ESTABLISHED
tcp     0      0 172.24.43.47:55238        198.252.206.25:443 ESTABLISHED
tcp     0      0 172.24.43.47:48850        172.217.18.227:443 ESTABLISHED
tcp     0      0 172.24.43.47:47040        172.217.19.46:443 ESTABLISHED
tcp     0      0 172.24.43.47:42270        173.194.76.189:443 ESTABLISHED
Sockets du domaine UNIX actives (sans serveurs)
Proto RefCnt Flags       Type            State          I-Node    Chemin
unix   17      [ ]      DGRAM           CONNECTE      15955    /run/systemd/journal/dev-log
unix   2      [ ]      DGRAM           CONNECTE      15960    /run/systemd/journal/syslog
unix   3      [ ]      DGRAM           CONNECTE      13727    /run/systemd/notify
unix   2      [ ]      DGRAM           CONNECTE      24806    /run/user/1000/systemd/notify
unix   8      [ ]      DGRAM           CONNECTE      14581    /run/systemd/journal/socket
unix   3      [ ]      STREAM          CONNECTE     44601    @/tmp/.X11-unix/X0
unix   3      [ ]      STREAM          CONNECTE     37078    @/tmp/dbus-bYdvfr0i6q
unix   3      [ ]      STREAM          CONNECTE     23274
unix   3      [ ]      STREAM          CONNECTE     25020
unix   3      [ ]      STREAM          CONNECTE     24991    /var/run/dbus/system_bus_socket
unix   3      [ ]      STREAM          CONNECTE     24931
unix   3      [ ]      STREAM          CONNECTE     25819
unix   3      [ ]      STREAM          CONNECTE     26677
unix   3      [ ]      STREAM          CONNECTE     24128
unix   3      [ ]      STREAM          CONNECTE     42196
unix   3      [ ]      STREAM          CONNECTE     20116    @/tmp/dbus-40oXWGcyDm
unix   3      [ ]      STREAM          CONNECTE     22247
unix   3      [ ]      STREAM          CONNECTE     23064    @/tmp/ibus/dbus-W1ytP0KY
unix   3      [ ]      STREAM          CONNECTE     20153    @/tmp/dbus-bYdvfr0i6q
unix   3      [ ]      STREAM          CONNECTE     25018
unix   3      [ ]      STREAM          CONNECTE     22358    /var/run/dbus/system_bus_socket
unix   3      [ ]      STREAM          CONNECTE     21347    @/tmp/.X11-unix/X0
unix   2      [ ]      DGRAM           CONNECTE      24956
unix   3      [ ]      STREAM          CONNECTE     26669
unix   3      [ ]      STREAM          CONNECTE     24112
unix   3      [ ]      STREAM          CONNECTE     22340
unix   3      [ ]      STREAM          CONNECTE     21111
unix   3      [ ]      DGRAM           CONNECTE      1643
unix   3      [ ]      STREAM          CONNECTE     21760
unix   3      [ ]      STREAM          CONNECTE     24330    @/tmp/.X11-unix/X0
unix   3      [ ]      STREAM          CONNECTE     24121    @/tmp/.X11-unix/X0
unix   3      [ ]      STREAM          CONNECTE     21441
unix   3      [ ]      STREAM          CONNECTE     21132    @/tmp/.X11-unix/X0
unix   3      [ ]      STREAM          CONNECTE     19120
unix   3      [ ]      STREAM          CONNECTE     27535
unix   3      [ ]      STREAM          CONNECTE     24403    @/tmp/.X11-unix/X0
unix   3      [ ]      STREAM          CONNECTE     25013
unix   3      [ ]      STREAM          CONNECTE     20128
unix   3      [ ]      STREAM          CONNECTE     25916
unix   3      [ ]      STREAM          CONNECTE     26640
unix   3      [ ]      STREAM          CONNECTE     19938    /run/systemd/journal/stdout
unix   3      [ ]      STREAM          CONNECTE     42265
unix   3      [ ]      STREAM          CONNECTE     21352
unix   3      [ ]      STREAM          CONNECTE     34025    /run/systemd/journal/stdout
unix   3      [ ]      STREAM          CONNECTE     37065    @/tmp/dbus-bYdvfr0i6q
unix   2      [ ]      STREAM          CONNECTE     42373
unix   3      [ ]      STREAM          CONNECTE     24021    /run/systemd/journal/stdout
unix   3      [ ]      STREAM          CONNECTE     24301
```

FIGURE 1.5 – résultat netstat -n

— netstat --inet --udp

Affiche l'état des connexions udp

```
bounkoue@interface:~$ netstat --inet --udp
Connexions Internet actives (sans serveurs)
Proto Recv-Q Send-Q Adresse locale           Adresse distante         Etat
```

FIGURE 1.6 – netstat -inet -udp

— netstat -r --inet -6

Affiche la table de routage IPv4 et IPv6

```
bounkoue@interface:~$ netstat -r --inet -6
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic   MSS Fenêtre irtt Iface
default         vl33-miraa-ltp  0.0.0.0        UG      0 0      0 enp0s31f6
link-local      *              255.255.0.0    U       0 0      0 enp0s31f6
172.24.0.0      *              255.255.128.0  U       0 0      0 enp0s31f6
dhcp1.rp.u-bour vl33-miraa-ltp 255.255.255.255 UGH      0 0      0 enp0s31f6
Table de routage IPv6 du noyau
Destination          Next Hop          Flag Met Ref Use If
fe80::/64             ::               U   256 0      0 enp0s31f6
::/0                  ::               !n  -1  1  2125 lo
::1/128               ::               Un  0  7  11 lo
fe80::896b:e419:b42c:e9ac/128 ::               Un  0  1  0 lo
ff00::/8               ::               U   256 6  236 enp0s31f6
::/0                  ::               !n  -1  1  2125 lo
```

FIGURE 1.7 – table de routage I

— netstat -a

Affiche toutes les sockets, y compris les sockets d'écoute des serveurs.

```
bounkoue@Interface:~$ netstat -a
Connexions Internet actives (serveurs et établies)
Proto Recv-Q Send-Q Adresse locale           Adresse distante      Etat
tcp     0      0 interface:domain        *.*                  LISTEN
tcp     0      0 *:ssh                 *.*                  LISTEN
tcp     0      0 localhost:ipp          *.*                  LISTEN
tcp     0      0 gr19-13.esirem-ad:36208 mrs08s02-in-f1.1e:https ESTABLISHED
tcp     0      0 gr19-13.esirem-ad:42268 ws-in-f189.1e100.:https ESTABLISHED
tcp     0      0 gr19-13.esirem-ad:55238 stackoverflow.com:https ESTABLISHED
tcp     0      0 gr19-13.esirem-ad:48850 mrs08s02-in-f3.1e:https TIME_WAIT
tcp     0      0 gr19-13.esirem-ad:47040 mrs08s03-in-f14.1:https ESTABLISHED
tcp     0      0 gr19-13.esirem-ad:42270 ws-in-f189.1e100.:https ESTABLISHED
tcp6    0      0 [::]:ssh              [::]:*                LISTEN
tcp6    0      0 ip6-localhost:ipp      [::]:*                LISTEN
udp     0      0 *:38745              *.*                  *
udp     0      0 interface:domain      *.*                  *
udp     0      0 *:bootpc             *.*                  *
udp     0      0 *:ipp                *.*                  *
udp     0      0 *:36395              *.*                  *
udp     0      0 *:mdns               *.*                  *
udp6    0      0 [::]:45037            [::]:*                *
udp6    0      0 [::]:mdns             [::]:*                *
raw6   0      0 [::]:ipv6-icmp       [::]:*                7
Sockets du domaine UNIX actives (serveurs et établies)
Proto RefCnt Flags     Type      State         I-Node  Chemin
unix  2      [ ACC ]  STREAM    LISTENING  25866  /run/user/1000/keyring/pkcs11
unix  2      [ ACC ]  STREAM    LISTENING  20794  /tmp/.X11-unix/X0
unix  2      [ ACC ]  STREAM    LISTENING  25869  /run/user/1000/keyring/ssh
unix  2      [ ACC ]  STREAM    LISTENING  25995  /tmp/.ICE-unix/1509
unix  2      [ ACC ]  STREAM    LISTENING  32725  /tmp/OSL_PIPE_1000_SingleOfficeIPC_e3739bb863b956b1245f33559a7ffc67
unix  2      [ ACC ]  STREAM    LISTENING  25994  @/tmp/.ICE-unix/1509
unix  2      [ ACC ]  STREAM    LISTENING  20793  @/tmp/.X11-unix/X0
unix  2      [ ACC ]  STREAM    LISTENING  48746  @bounkoue-com.canonical.Unity.Master.Scope.files.T17878992316125
unix  17     [ ]       DGRAM     LISTENING  15955  /run/systemd/journal/dev-log
unix  2      [ ]       DGRAM     LISTENING  15960  /run/systemd/journal/syslog
unix  2      [ ACC ]  STREAM    LISTENING  25853  @/tmp/dbus-bvdvfr0l6q
unix  2      [ ACC ]  STREAM    LISTENING  48745  @bounkoue-com.canonical.Unity.Master.Scope.applications.T17878988454434
unix  2      [ ACC ]  STREAM    LISTENING  14194  /var/run/avahi-daemon/socket
unix  2      [ ACC ]  STREAM    LISTENING  14197  /run/uuidd/request
unix  2      [ ACC ]  STREAM    LISTENING  24953  /run/user/1000/pulse/native
unix  2      [ ACC ]  STREAM    LISTENING  14200  /var/run/cups/cups.sock
unix  2      [ ACC ]  STREAM    LISTENING  14203  /run/acpid.socket
unix  2      [ ACC ]  STREAM    LISTENING  14205  /var/run/dbus/system_bus_socket
unix  2      [ ACC ]  STREAM    LISTENING  14208  /run/snapd.socket
unix  2      [ ACC ]  STREAM    LISTENING  14210  /run/snapd-snap.socket
unix  3      [ ]       DGRAM     LISTENING  13727  /run/systemd/notify
unix  2      [ ACC ]  STREAM    LISTENING  22162  @/com/ubuntu/upstart-session/1000/1281
unix  2      [ ACC ]  STREAM    LISTENING  13729  /run/systemd/private
unix  2      [ ACC ]  STREAM    LISTENING  50373  @bounkoue-com.canonical.Unity.Scope.scopes.T17882849821590
unix  2      [ ACC ]  STREAM    LISTENING  23068  /home/bounkoue/.gnupg/S.gpg-agent
unix  2      [ ACC ]  STREAM    LISTENING  50374  @bounkoue-com.canonical.Unity.Scope.applications.T17882853791782
unix  2      [ ACC ]  STREAM    LISTENING  47458  @bounkoue-com.canonical.Unity.Scope.files.T17879407367014
unix  2      [ ACC ]  STREAM    LISTENING  37329  /var/run/NetworkManager/private-dhcp
```

FIGURE 1.8 – résultat netstat -a

## 2. Utilité des options -a et -l de la commande

- Option -a : affiche toutes les sockets, y compris les sockets d'écoute des serveurs.
- Option -l : affiche uniquement les sockets d'écoute des serveurs.

3. On veut maintenant afficher la table des sockets en attente de connexions (état LISTENING) sans résolution des noms. La commande à utiliser est :

```
netstat -l
```

On obtient le résultat suivant :

```
bounkoue@Interface:~$ netstat -l
Connexions Internet actives (seulement serveurs)
Proto Recv-Q Send-Q Adresse locale      Adresse distante      Etat
tcp     0      0 interface:domain      *.*               LISTEN
tcp     0      0 *:ssh                *.*               LISTEN
tcp     0      0 localhost:ipp        *.*               LISTEN
tcp6    0      0 [::]:ssh             [::]:*            LISTEN
tcp6    0      0 ip6-localhost:ipp   [::]:*            LISTEN
udp     0      0 *:38745              *.*               LISTEN
udp     0      0 interface:domain    *.*               LISTEN
udp     0      0 *:bootpc             *.*               LISTEN
udp     0      0 *:ipp                *.*               LISTEN
udp     0      0 *:36395              *.*               LISTEN
udp     0      0 *:mdns               *.*               LISTEN
udp6    0      0 [::]:45037            [::]:*            LISTEN
udp6    0      0 [::]:mdns             [::]:*            LISTEN
raw6   0      0 [::]:ipv6-icmp       [::]:*            7
Sockets du domaine UNIX actives (seulement serveurs)
Proto RefCnt Flags     Type      State      I-Node  Chemin
unix  2      [ ACC ]  STREAM    LISTENING  25866  /run/user/1000/keyring/pkcs11
unix  2      [ ACC ]  STREAM    LISTENING  20794  /tmp/.X11-unix/X0
unix  2      [ ACC ]  STREAM    LISTENING  25869  /run/user/1000/keyring/ssh
unix  2      [ ACC ]  STREAM    LISTENING  25995  /tmp/.ICE-unix/1509
unix  2      [ ACC ]  STREAM    LISTENING  32725  /tmp/OSL_PIPE_1000_SingleOfficeIPC_e3739bb863b956b1245f33559a7ffc67
unix  2      [ ACC ]  STREAM    LISTENING  25994  @/tmp/.ICE-unix/1509
unix  2      [ ACC ]  STREAM    LISTENING  20793  @/tmp/.X11-unix/X0
unix  2      [ ACC ]  STREAM    LISTENING  48746  @bounkoue-com.canonical.Unity.Master.Scope.files.T17878992316125
unix  2      [ ACC ]  STREAM    LISTENING  25853  @/tmp/dbus-bvdvfroiq
unix  2      [ ACC ]  STREAM    LISTENING  48745  @bounkoue-com.canonical.Unity.Master.Scope.applications.T17878988454434
unix  2      [ ACC ]  STREAM    LISTENING  14194  /var/run/avahi-daemon/socket
unix  2      [ ACC ]  STREAM    LISTENING  14197  /run/uidd/request
unix  2      [ ACC ]  STREAM    LISTENING  24953  /run/user/1000/pulse/native
unix  2      [ ACC ]  STREAM    LISTENING  14200  /var/run/cups/cups.sock
unix  2      [ ACC ]  STREAM    LISTENING  14203  /run/acpid.socket
unix  2      [ ACC ]  STREAM    LISTENING  14205  /var/run/dbus/system_bus_socket
unix  2      [ ACC ]  STREAM    LISTENING  14208  /run/snapd.socket
unix  2      [ ACC ]  STREAM    LISTENING  14210  /run/snapd-snap.socket
unix  2      [ ACC ]  STREAM    LISTENING  22162  @/com/ubuntu/upstart-session/1000/1281
unix  2      [ ACC ]  STREAM    LISTENING  13729  /run/systemd/private
unix  2      [ ACC ]  STREAM    LISTENING  50373  @bounkoue-com.canonical.Unity.Scope.scopes.T17882849821590
unix  2      [ ACC ]  STREAM    LISTENING  23068  /home/bounkoue/.gnupg/S.gpg-agent
unix  2      [ ACC ]  STREAM    LISTENING  50374  @bounkoue-com.canonical.Unity.Scope.applications.T17882853791782
unix  2      [ ACC ]  STREAM    LISTENING  47458  @bounkoue-com.canonical.Unity.Scope.files.T17879407367014
unix  2      [ ACC ]  STREAM    LISTENING  37329  /var/run/NetworkManager/private-dhcp
unix  2      [ ACC ]  STREAM    LISTENING  25971  @/tmp/dbus-400XWGcyDm
unix  2      [ ACC ]  STREAM    LISTENING  25832  /run/user/1000/keyring/control
unix  2      [ ACC ]  STREAM    LISTENING  24808  /run/user/1000/systemd/private
unix  2      [ ACC ]  STREAM    LISTENING  14569  /run/systemd/fsck.progress
unix  2      [ ACC ]  SEQPACKET  LISTENING  14571  /run/udev/control
unix  2      [ ACC ]  STREAM    LISTENING  14578  /run/systemd/journal/stdout
unix  2      [ ACC ]  STREAM    LISTENING  24870  @/tmp/ibus/dbus-W1yiPOKY
bounkoue@Interface:~$
```

FIGURE 1.9 – table des sockets en attente de connexions : netstat -l

### 1.3.3 Politique de routage

La commande route permet de visualiser l'ensemble des politiques de routage de la machine sous forme d'une table de routage.

Affichage de la table de routage

```
bounkoue@Interface:~$ route
Table de routage IP du noyau
Destination     Passerelle      Genmask      Indic Metric Ref  Use Iface
default         vl33-miraa-ltp- 0.0.0.0      UG  100   0      0 enp0s31f6
link-local      *               255.255.0.0   U    1000  0      0 enp0s31f6
172.24.0.0      *               255.255.128.0  U    100   0      0 enp0s31f6
dhcp1.rp.u-bour vl33-miraa-ltp- 255.255.255.255 UGH  100   0      0 enp0s31f6
```

FIGURE 1.10 – table de routage

1. La ligne de la table de routage qui correspond au routage du traffic entre les machine de la salle de TP est la ligne 3

```
bounkoue@Interface:~$ route
Table de routage IP du noyau
Destination     Passerelle      Genmask      Indic Metric Ref  Use Iface
default         vl33-miraa-ltp- 0.0.0.0      UG  100   0      0 enp0s31f6
link-local      *               255.255.0.0   U    1000  0      0 enp0s31f6
172.24.0.0      *               255.255.128.0  U    100   0      0 enp0s31f6
dhcp1.rp.u-bour vl33-miraa-ltp- 255.255.255.255 UGH  100   0      0 enp0s31f6
```

FIGURE 1.11 – ligne correspondant au traffic de la salle de TP

2. Une passerelle sert à communiquer avec d'autres réseaux
3. — Si l'on souhaite envoyer un message à la machine d'adresse IP 80.80.80.80 c'est la première ligne de la table qui sera considérée

```
bounkoue@interface:~$ route
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref  Use Iface
default         vl33-miraa-ltp  0.0.0.0        UG    100    0      0 enp0s31f6
link-local      *               255.255.0.0   U     1000   0      0 enp0s31f6
172.24.0.0      255.255.128.0 U     100    0      0 enp0s31f6
dhcp1.rp.u-bour vl33-miraa-ltp  255.255.255.255 UGH   100    0      0 enp0s31f6
```

FIGURE 1.12 – ligne correspondant au traffic extérieur

- La prochaine machine à relayer le message sera alors **vl33-miraa-ltp-** et le message lui sera relayé via l'interface **enp0s31f6**
- 4. La passerelle par défaut sur notre système est la machine **vl33-miraa-ltp-**

### 1.3.4 Résolution des noms de domaine

Afin de faciliter l'utilisation des réseaux, les machines peuvent être associées à un nom qui est généralement plus facile à retenir qu'une adresse IP. La résolution d'un nom est la conversion d'un nom sous forme d'adresse (ex : yahoo.fr) en une adresse IP (ex : 87.248.120.148). Cette résolution est opérée par un serveur distant, à qui l'on envoie une demande de résolution, et qui répond par l'adresse IP correspondante. Pour visualiser brièvement ces mécanismes, on utilisera la commande nslookup.

1. Nous exécutons la commande nslookup en lui passant www.yahoo.fr en paramètre :

```
bounkoue@interface:~$ nslookup www.yahoo.fr
Server:          127.0.1.1
Address:         127.0.1.1#53

Non-authoritative answer:
www.yahoo.fr      canonical name = rc.yahoo.com.
rc.yahoo.com      canonical name = src.g03.yahoodns.net.
Name:             src.g03.yahoodns.net
Address:          212.82.100.150
```

FIGURE 1.13 – nslookup exécuté sur www.yahoo.fr

2. Des résultats précédents on note que :
  - L'adresse IP du serveur DNS est **127.0.1.1**
  - L'adresse IP de la machine www.yahoo.fr est **212.82.100.150**
3. Le fichier /etc/resolv.conf contient l'adresse des différents serveurs DNS disponibles sur le réseau

```
bounkoue@interface:~$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#      DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 127.0.1.1
search ensgt.u-bourgogne.fr
```

FIGURE 1.14 – contenu du fichier /etc/resolv.conf

4. Le fichier /etc/hosts quant à lui, contient un mapping entre des noms de machine et leur adresse IP

```
bounkoue@interface:~$ cat /etc/hosts
127.0.0.1      localhost
127.0.1.1      interface

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

FIGURE 1.15 – contenu du fichier /etc/hosts

5. Nous ajoutons 2 lignes (correspondances) à notre fichier /etc/hosts

```
bounkoue@interface:~$ cat /etc/hosts
127.0.0.1      localhost
127.0.1.1      interface

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
#Nouvelles lignes
172.24.43.41  quentin
172.24.43.42  jules
```

FIGURE 1.16 – ajout de deux lignes au fichier /etc/hosts

Puis nous effectuons un ping vers ces deux nouvelles machines répertoriées avec leur nom respectif

```
bounkoue@interface:~$ ping quentin
PING quentin (172.24.43.41) 56(84) bytes of data.
64 bytes from quentin (172.24.43.41): icmp_seq=1 ttl=64 time=0.480 ms
64 bytes from quentin (172.24.43.41): icmp_seq=2 ttl=64 time=0.657 ms
64 bytes from quentin (172.24.43.41): icmp_seq=3 ttl=64 time=0.655 ms
64 bytes from quentin (172.24.43.41): icmp_seq=4 ttl=64 time=0.659 ms
64 bytes from quentin (172.24.43.41): icmp_seq=5 ttl=64 time=0.424 ms
64 bytes from quentin (172.24.43.41): icmp_seq=6 ttl=64 time=0.448 ms
64 bytes from quentin (172.24.43.41): icmp_seq=7 ttl=64 time=0.445 ms
64 bytes from quentin (172.24.43.41): icmp_seq=8 ttl=64 time=0.653 ms
64 bytes from quentin (172.24.43.41): icmp_seq=9 ttl=64 time=0.661 ms
64 bytes from quentin (172.24.43.41): icmp_seq=10 ttl=64 time=0.639 ms
64 bytes from quentin (172.24.43.41): icmp_seq=11 ttl=64 time=0.656 ms
64 bytes from quentin (172.24.43.41): icmp_seq=12 ttl=64 time=0.653 ms
64 bytes from quentin (172.24.43.41): icmp_seq=13 ttl=64 time=0.660 ms
64 bytes from quentin (172.24.43.41): icmp_seq=14 ttl=64 time=0.599 ms
^C
--- quentin ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13300ms
rtt min/avg/max/mdev = 0.424/0.592/0.661/0.092 ms
bounkoue@interface:~$ ping jules
PING jules (172.24.43.42) 56(84) bytes of data.
64 bytes from jules (172.24.43.42): icmp_seq=1 ttl=64 time=0.941 ms
64 bytes from jules (172.24.43.42): icmp_seq=2 ttl=64 time=0.686 ms
64 bytes from jules (172.24.43.42): icmp_seq=3 ttl=64 time=0.682 ms
64 bytes from jules (172.24.43.42): icmp_seq=4 ttl=64 time=0.685 ms
64 bytes from jules (172.24.43.42): icmp_seq=5 ttl=64 time=0.678 ms
64 bytes from jules (172.24.43.42): icmp_seq=6 ttl=64 time=0.591 ms
64 bytes from jules (172.24.43.42): icmp_seq=7 ttl=64 time=0.725 ms
64 bytes from jules (172.24.43.42): icmp_seq=8 ttl=64 time=0.686 ms
64 bytes from jules (172.24.43.42): icmp_seq=9 ttl=64 time=0.677 ms
64 bytes from jules (172.24.43.42): icmp_seq=10 ttl=64 time=0.686 ms
64 bytes from jules (172.24.43.42): icmp_seq=11 ttl=64 time=0.565 ms
64 bytes from jules (172.24.43.42): icmp_seq=12 ttl=64 time=0.366 ms
64 bytes from jules (172.24.43.42): icmp_seq=13 ttl=64 time=0.615 ms
```

FIGURE 1.17 – ping avec les deux nouveaux noms de machines

## 1.4 Manipulation avancées

### 1.4.1 Etude d'une trame ethernet

1. Nous effectuons tour à tour un ping :

— Vers les machines de la salle

```
bounkoue@interface:~$ ping quentin
PING quentin (172.24.43.41) 56(84) bytes of data.
64 bytes from quentin (172.24.43.41): icmp_seq=1 ttl=64 time=0.480 ms
64 bytes from quentin (172.24.43.41): icmp_seq=2 ttl=64 time=0.657 ms
64 bytes from quentin (172.24.43.41): icmp_seq=3 ttl=64 time=0.655 ms
64 bytes from quentin (172.24.43.41): icmp_seq=4 ttl=64 time=0.659 ms
64 bytes from quentin (172.24.43.41): icmp_seq=5 ttl=64 time=0.424 ms
64 bytes from quentin (172.24.43.41): icmp_seq=6 ttl=64 time=0.448 ms
64 bytes from quentin (172.24.43.41): icmp_seq=7 ttl=64 time=0.445 ms
64 bytes from quentin (172.24.43.41): icmp_seq=8 ttl=64 time=0.653 ms
64 bytes from quentin (172.24.43.41): icmp_seq=9 ttl=64 time=0.661 ms
64 bytes from quentin (172.24.43.41): icmp_seq=10 ttl=64 time=0.639 ms
64 bytes from quentin (172.24.43.41): icmp_seq=11 ttl=64 time=0.656 ms
64 bytes from quentin (172.24.43.41): icmp_seq=12 ttl=64 time=0.653 ms
64 bytes from quentin (172.24.43.41): icmp_seq=13 ttl=64 time=0.660 ms
64 bytes from quentin (172.24.43.41): icmp_seq=14 ttl=64 time=0.599 ms
^C
--- quentin ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13300ms
rtt min/avg/max/mdev = 0.424/0.592/0.661/0.092 ms
bounkoue@interface:~$ ping jules
PING jules (172.24.43.42) 56(84) bytes of data.
64 bytes from jules (172.24.43.42): icmp_seq=1 ttl=64 time=0.941 ms
64 bytes from jules (172.24.43.42): icmp_seq=2 ttl=64 time=0.686 ms
64 bytes from jules (172.24.43.42): icmp_seq=3 ttl=64 time=0.682 ms
64 bytes from jules (172.24.43.42): icmp_seq=4 ttl=64 time=0.685 ms
64 bytes from jules (172.24.43.42): icmp_seq=5 ttl=64 time=0.678 ms
64 bytes from jules (172.24.43.42): icmp_seq=6 ttl=64 time=0.591 ms
64 bytes from jules (172.24.43.42): icmp_seq=7 ttl=64 time=0.725 ms
64 bytes from jules (172.24.43.42): icmp_seq=8 ttl=64 time=0.686 ms
64 bytes from jules (172.24.43.42): icmp_seq=9 ttl=64 time=0.677 ms
64 bytes from jules (172.24.43.42): icmp_seq=10 ttl=64 time=0.686 ms
64 bytes from jules (172.24.43.42): icmp_seq=11 ttl=64 time=0.565 ms
64 bytes from jules (172.24.43.42): icmp_seq=12 ttl=64 time=0.366 ms
64 bytes from jules (172.24.43.42): icmp_seq=13 ttl=64 time=0.615 ms
```

FIGURE 1.18 – ping vers les machines de la salle

— Vers www.yahoo.fr

```
bounkoue@interface:~$ ping www.yahoo.fr
PING src.g03.yahoodns.net (212.82.100.150) 56(84) bytes of data.
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=1 ttl=50 time=26.3 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=2 ttl=50 time=26.3 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=3 ttl=50 time=26.4 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=4 ttl=50 time=26.4 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=5 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=6 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=7 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=8 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=9 ttl=50 time=26.4 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=10 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=11 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=12 ttl=50 time=26.4 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=13 ttl=50 time=26.4 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=14 ttl=50 time=221 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=15 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=16 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=17 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=18 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=19 ttl=50 time=26.5 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=20 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=21 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=22 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=23 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=24 ttl=50 time=26.1 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=25 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=26 ttl=50 time=26.3 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=27 ttl=50 time=26.3 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=28 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=29 ttl=50 time=26.3 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=30 ttl=50 time=26.2 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=31 ttl=50 time=26.3 ms
64 bytes from w2.src.vip.ir2.yahoo.com (212.82.100.150): icmp_seq=32 ttl=50 time=26.2 ms
```

FIGURE 1.19 – ping vers www.yahoo.fr

— Comportement de la commande

On remarque que la commande s'exécute à l'infini et que le temps de réponse est beaucoup plus grand lorsqu'on adresse le ping vers www.yahoo.fr. Par contre lorsque nous faisons un ping vers une machine du réseau les temps de réponses sont faibles et très voisins.

— Le champ ttl correspond au nombre maximum de sauts avant que le paquet ne soit droppé

## 2. Exploration wireshark et capture paquets avec filtre ip

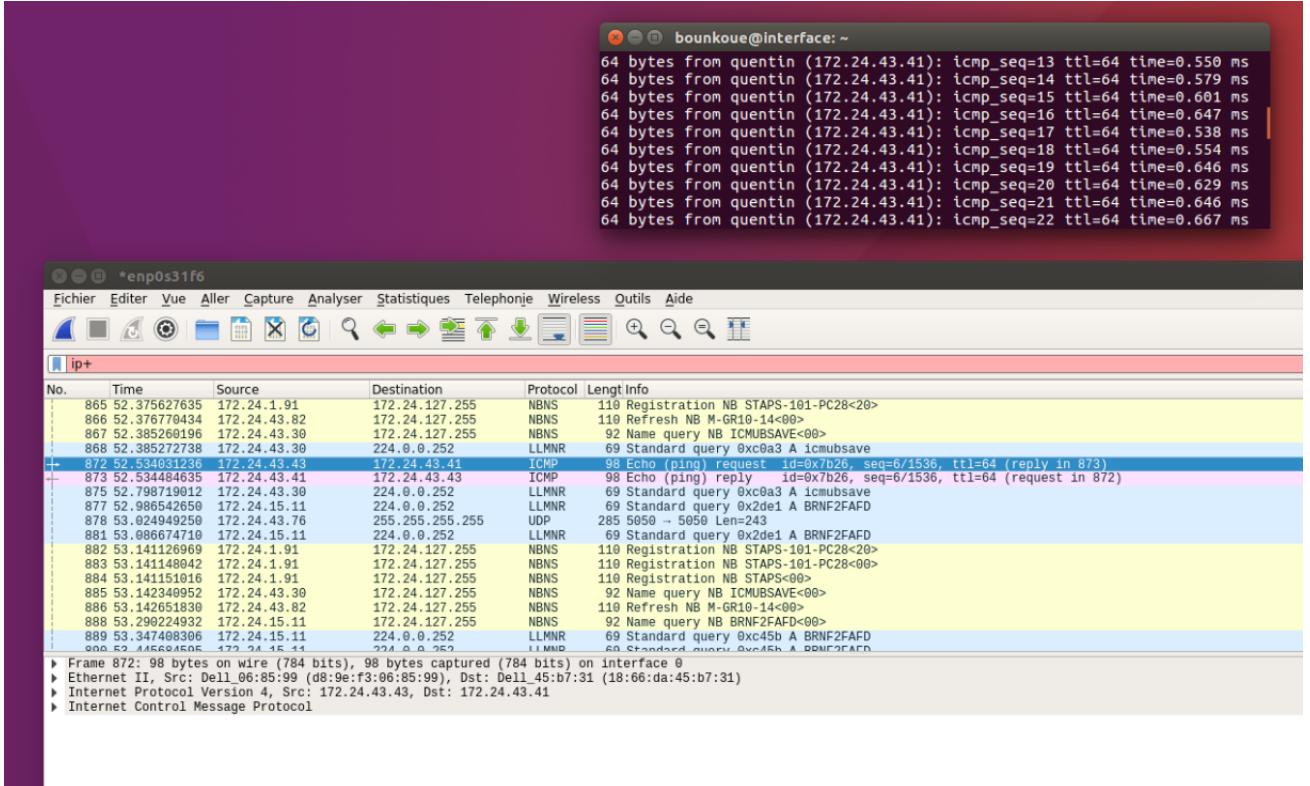


FIGURE 1.20 – capture des paquets issues du ping avec wireshark

## 3. Regardons de plus près une trame capturée par wireshark issue d'une requête ping vers une machine de notre réseau. On peut en tirer les informations suivantes :

```

▼ Frame 4/: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
  ▶ Interface id: 0 (enp0s31f6)
    Encapsulation type: Ethernet (1)
    Arrival Time: Jan 9, 2019 14:34:30.441292270 CET
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1547040870.441292270 seconds
    [Time delta from previous captured frame: 0.097809735 seconds]
    [Time delta from previous displayed frame: 0.097809735 seconds]
    [Time since reference or first frame: 1.446586193 seconds]
    Frame Number: 47
    Frame Length: 98 bytes (784 bits)
    Capture Length: 98 bytes (784 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:icmp:data]
    [Coloring Rule Name: ICMP]
    [Coloring Rule String: icmp || icmpv6]
  ▶ Ethernet II, Src: Dell_06:85:99 (d8:9e:f3:06:85:99), Dst: Dell_45:b7:31 (18:66:da:45:b7:31)
    ▶ Internet Protocol Version 4, Src: 172.24.43.43, Dst: 172.24.43.41
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
      Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 84
      Identification: 0x954d (38221)
      Flags: 0x4000, Don't fragment
      Time to live: 64
      Protocol: ICMP (1)
      Header checksum: 0xf6d6 [validation disabled]
      [Header checksum status: Unverified]
      Source: 172.24.43.43
      Destination: 172.24.43.41
    ▶ Internet Control Message Protocol
  
```

FIGURE 1.21 – détails trame capturée

- Protocol utilisé : **ICMP (Internet Control Message Protocol)**
- Numéro de type de paquet : **8**
- L'adresse IP du destinataire se situe dans le champ **Destination** de la couche réseau
- La valeur du TTL est **64**
- Taille des en têtes pour chaque niveau d'encapsulation
  - taille de l'entête de la couche réseau : **20 octets**. Elle se lit directement dans wireshark
  - taille de l'en tête de la couche application : **64 - 48 = 16 octects**. Elle est obtenue en déduisant la taille du paquet sans header à la taille des données applicatives.

## 4. Lorsqu'on effectue l'opération de visualisation des paquets en faisant varier le ttl, il se passe que pour de trop petites valeurs du ttl, la machine de destination ne reçoit plus la requête. On remarque

alors dans wireshark que le type du message de retour est **11** et le message est "**Time to live exceeded**". D'ailleurs on peut également le voir au terminal.

```
bounkoue@Interface:~$ ping 8.8.8.8 -t 2
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.254.0.17 icmp_seq=1 Time to live exceeded
From 10.254.0.17 icmp_seq=2 Time to live exceeded
From 10.254.0.17 icmp_seq=3 Time to live exceeded
From 10.254.0.17 icmp_seq=4 Time to live exceeded
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3049ms

bounkoue@Interface:~$ ping 8.8.8.8 -t 3
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 192.168.144.12 icmp_seq=1 Time to live exceeded
From 192.168.144.12 icmp_seq=2 Time to live exceeded
From 192.168.144.12 icmp_seq=3 Time to live exceeded
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2041ms
```

FIGURE 1.22 – visualisation d'un ttl épuisé au terminal

L'adresse de la machine qui répond dans ce cas est alors celle d'un routeur intermédiaire variant selon la valeur du TTL utilisé. Il s'agit du routeur auquel le TTL arrive à 0.

Conclusion sur l'importance du TTL : Le TTL détermine le nombre de sauts qu'un paquet peut subir avant d'être détruit. Il doit alors être suffisamment grand pour que le paquet puisse attendre la destination avant que sa valeur devienne nulle. Le TTL permet notamment d'éviter le congestionnement en cas de rupture de lien par exemple dans le réseau.

5. Pour déterminer l'ensemble des adresses IP des routeurs qui se situent entre notre machine et un hôte dont l'adresse IP est donnée, il suffit d'initialiser le TTL à 1, de faire un ping et de l'incrémenter et répéter l'envoi d'un ping jusqu'à ce que l'hôte de destination réponde. On note alors l'ensemble des adresses IP des machines nous ayant envoyé le message "Time to live exceeded", ces machines sont les routeurs intermédiaires se trouvant entre nous et la destination.

#### 1.4.2 La commande traceroute

Traceroute (ou tracert sous Windows) est un programme utilitaire qui permet de suivre les chemins qu'un paquet de données (paquet IP) va prendre pour aller de la machine locale à une autre machine connectée au réseau IP. Il a été conçu au sein du Laboratoire national Lawrence-Berkeley.

1. Nous effectuons maintenant un traceroute vers l'hôte **bonifacio.fr** et notons la liste des routeurs empruntés.

```
bounkoue@Interface:~$ traceroute bonifacio.fr
traceroute to bonifacio.fr (149.202.69.160), 30 hops max, 60 byte packets
 1  vl33-miraa-ltp-1a-agr01.net.u-bourgogne.fr (172.24.0.1)  11.275 ms  11.274 ms  11.258 ms
 2  vl3204-dijon-core01.net.u-bourgogne.fr (10.254.0.17)  0.243 ms  0.537 ms  0.524 ms
 3  vl39-hogan2.net.u-bourgogne.fr (192.168.144.12)  0.454 ms  0.469 ms  0.449 ms
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

FIGURE 1.23 – résultats traceroute vers bonifacio.fr

Nous remarquons que certains routeurs sont d'identité masqué et représenté par trois étoiles plutôt que par leur adresse IP. Cela est dû au fait qu'ils sont configurés pour ne pas répondre aux requêtes

ICMP, donc ils se contentent tout simplement soit de les dropper soit de les rediriger, et dans le cas où le TTL devient nul, ils les droppent.

- Enfin nous effectuons un ping vers un hôte `yahoo.fr` et notons l'adresse IP du premier routeur emprunté.

```
bounkoue@interface:~$ traceroute yahoo.fr
traceroute to yahoo.fr (98.136.103.24), 30 hops max, 60 byte packets
1 vl33-miraa-ltp-1a-agr01.net.u-bourgogne.fr (172.24.0.1) 12.106 ms 12.139 ms 12.131 ms
2 vl3204-dijon-core01.net.u-bourgogne.fr (10.254.0.17) 0.338 ms 0.351 ms 0.608 ms
3 vl39-hogan2.net.u-bourgogne.fr (192.168.144.12) 0.331 ms 0.328 ms 0.313 ms
4 * * *
5 * * *
6 * * *
7 * * *
8 * * *
9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *
```

FIGURE 1.24 – résultats traceroute vers yahoo.fr

Regardons maintenant notre table de routage.

Destination	Passerelle	Genmask	Indic	Metric	Ref	Use	Iface
default	vl33-miraa-ltp-	0.0.0.0	UG	100	0	0	enp0s31f6
link-local	*	255.255.0.0	U	1000	0	0	enp0s31f6
172.24.0.0	*	255.255.128.0	U	100	0	0	enp0s31f6
dhcp1.rp.u-bour	vl33-miraa-ltp-	255.255.255.255	UGH	100	0	0	enp0s31f6

FIGURE 1.25 – table de routage

On constate que l'adresse IP relevée précédemment est identique à celle de notre passerelle par défaut (même nom de domaine donc même adresse par résolution). Comme on pouvait s'y attendre car en effet, notre passerelle par défaut est le premier routeur par lequel transite un message issue de notre machine et donc il est normal qu'il soit celui qui figure sur la première ligne des résultats affichés par traceroute car la commande ne fait rien d'autre que des pings en incrémentant successivement le TTL à partir de 1. Notre passerelle par défaut se trouvant exactement à 1 saut de nous, elle apparaît donc en première position.

# Chapitre 2

## TP2 : Interconnexion des réseaux

### 2.1 Objectif du TP

Mise en place d'une architecture réseau interconnectant différents sous réseaux, chacun d'entre eux étant administré par un groupe de TP.

- Chaque groupe possède son propre réseau, constitué d'un routeur et de deux machines clientes.
- Chaque routeur est également connecté au réseau de l'université, qui lui donne accès à une connexion internet. Ces routeurs sont sur le même commutateur de la salle de TP, ce qui permet une connexion directe entre chaque paire de routeur.
- Un autre réseau (192.168.255.0) permet la connexion entre ces routeurs. Ce réseau est commun à tous les groupes. Le but est que la communication entre les différentes machines de la salle transite via les routeurs au moyen de ce réseau, et non au moyen du réseau de l'université. La figure suivante présente l'architecture du réseau à mettre en place :

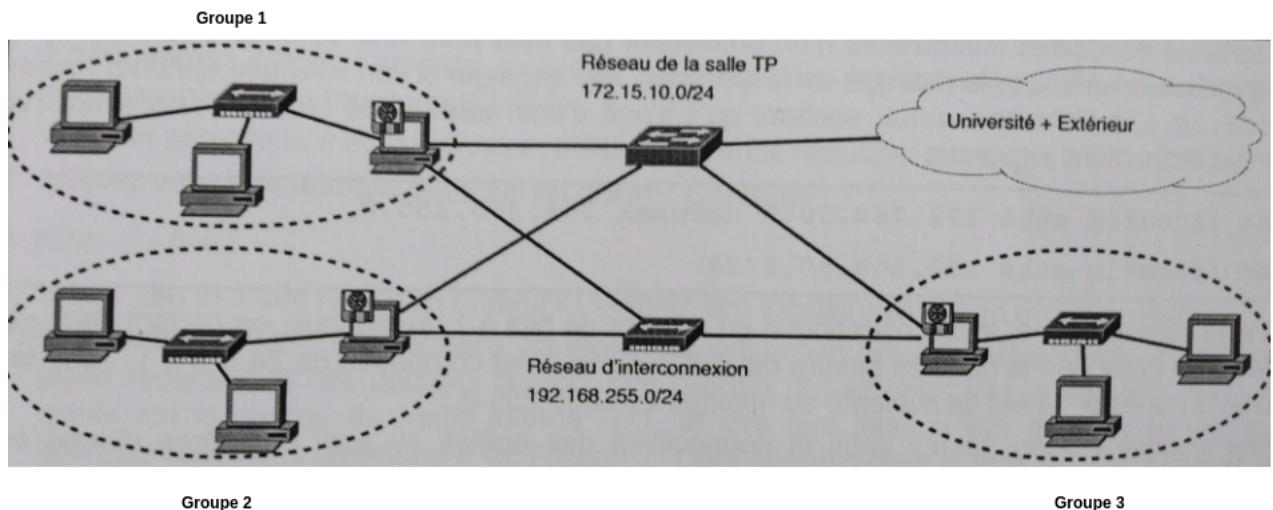


FIGURE 2.1 – architecture du réseau à créer

#### Adresse de chaque groupe

- Groupe 1 : 192.168.10.0/24 (le nôtre)
- Groupe 2 : 192.168.20.0/24
- Groupe 3 : 192.168.30.0/24

### 2.2 Prérequis

#### 2.2.1 Droits administrateur

Nous possédons déjà les droits d'administrateur sur notre machine.

## 2.2.2 Définition du plan d'adressage

1. Adresse réseau : 192.168.10.0/24
2. — Masque associé : 255.255.255.0
  - Adresse de diffusion : 192.168.10.255
3. Adresse du routeur : 192.168.10.255  
†Nous précisons que pour ce TP c'est notre machine fait office de routeur
4. Les deux autres réseaux du routeur sont :
  - (a) Le réseau de la salle de TP
    - Adresse du réseau : 172.24.0.0/17
    - Masque du réseau : 255.255.128.0
    - Adresse de diffusion : 172.24.127.255
    - Adresse de notre routeur sur ce réseau : 172.24.43.43
  - (b) Le réseau d'interconnexion
    - Adresse du réseau : 192.168.255.0/24
    - Masque du réseau : 255.255.255.0
    - Adresse de diffusion : 192.168.255.255
    - Adresse de notre routeur sur ce réseau : 192.168.255.1

## 2.3 Mise en place du réseau

### 2.3.1 Connexion physique

Nous branchons notre machine sur le port 1 du switch du réseau local.

### 2.3.2 Configuration logique

#### Mise en place des adresses et réseaux

- Attribution de l'adresse IP et du masque à l'interface **enp3s2** connectée au sous réseau

```
bounkoue@interface:~$ sudo ifconfig enp3s2 192.168.10.1/24
bounkoue@interface:~$ ifconfig
enp0s31f6 Link encap:Ethernet HWaddr d8:9e:f3:06:85:99
      inet adr:172.24.43.43 Bcast:172.24.127.255 Masque:255.255.128.0
        adr inet6: fe80::cf1c:135c:b274:9244/64 Scope:lien
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          Packets reçus:153797 erreurs:0 :0 overruns:0 frame:0
          TX packets:72613 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:133154970 (133.1 MB) Octets transmis:10553780 (10.5 MB)
          Interruption:16 Mémoire:f7f00000-f7f20000

enp3s2 Link encap:Ethernet HWaddr 00:0a:cd:f2:ec:bd
      inet adr:192.168.10.1 Bcast:192.168.10.255 Masque:255.255.255.0
        adr inet6: fe80::20a:cdff:fe2f:ecbd/64 Scope:lien
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          Packets reçus:93 erreurs:0 :0 overruns:0 frame:0
          TX packets:144 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:20570 (20.5 KB) Octets transmis:26107 (26.1 KB)

enp4s0 Link encap:Ethernet HWaddr 00:13:3b:0f:c0:21
      UP BROADCAST MULTICAST MTU:1500 Metric:1
      Packets reçus:0 erreurs:0 :0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 lg file transmission:1000
      Octets reçus:0 (0.0 B) Octets transmis:0 (0.0 B)

lo Link encap:Boucle locale
      inet adr:127.0.0.1 Masque:255.0.0.0
        adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING MTU:65536 Metric:1
          Packets reçus:9502 erreurs:0 :0 overruns:0 frame:0
          TX packets:9502 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:1846547 (1.8 MB) Octets transmis:1846547 (1.8 MB)
```

FIGURE 2.2 – attribution de l'adresse IP et du masque

- Test de connectivité dans le sous réseau On observe des réponses à nos requêtes ping ce qui assure de la bonne connectivité dans le sous réseau.

```
bounkoue@interface:~$ ping 192.168.10.2
PING 192.168.10.2 (192.168.10.2) 56(84) bytes of data.
64 bytes from 192.168.10.2: icmp_seq=1 ttl=64 time=0.279 ms
64 bytes from 192.168.10.2: icmp_seq=2 ttl=64 time=0.287 ms
64 bytes from 192.168.10.2: icmp_seq=3 ttl=64 time=0.390 ms
64 bytes from 192.168.10.2: icmp_seq=4 ttl=64 time=0.286 ms
64 bytes from 192.168.10.2: icmp_seq=5 ttl=64 time=0.306 ms
64 bytes from 192.168.10.2: icmp_seq=6 ttl=64 time=0.369 ms
64 bytes from 192.168.10.2: icmp_seq=7 ttl=64 time=0.287 ms
64 bytes from 192.168.10.2: icmp_seq=8 ttl=64 time=0.283 ms
64 bytes from 192.168.10.2: icmp_seq=9 ttl=64 time=0.283 ms
64 bytes from 192.168.10.2: icmp_seq=10 ttl=64 time=0.295 ms
64 bytes from 192.168.10.2: icmp_seq=11 ttl=64 time=0.285 ms
64 bytes from 192.168.10.2: icmp_seq=12 ttl=64 time=0.208 ms
64 bytes from 192.168.10.2: icmp_seq=13 ttl=64 time=0.395 ms
64 bytes from 192.168.10.2: icmp_seq=14 ttl=64 time=0.389 ms
64 bytes from 192.168.10.2: icmp_seq=15 ttl=64 time=0.390 ms
64 bytes from 192.168.10.2: icmp_seq=16 ttl=64 time=0.342 ms
64 bytes from 192.168.10.2: icmp_seq=17 ttl=64 time=0.275 ms
```

FIGURE 2.3 – ping dans le sous réseau

### Politique de routage

A ce stade, la connectivité des entités au sein du même sous réseau est assurée. Il reste à établir la communication entre entités de réseaux différents. Pour cela nous définissons tout d'abord les politiques de routage pour chacune des entités du réseau.

#### 1. — Architecture du sous réseau

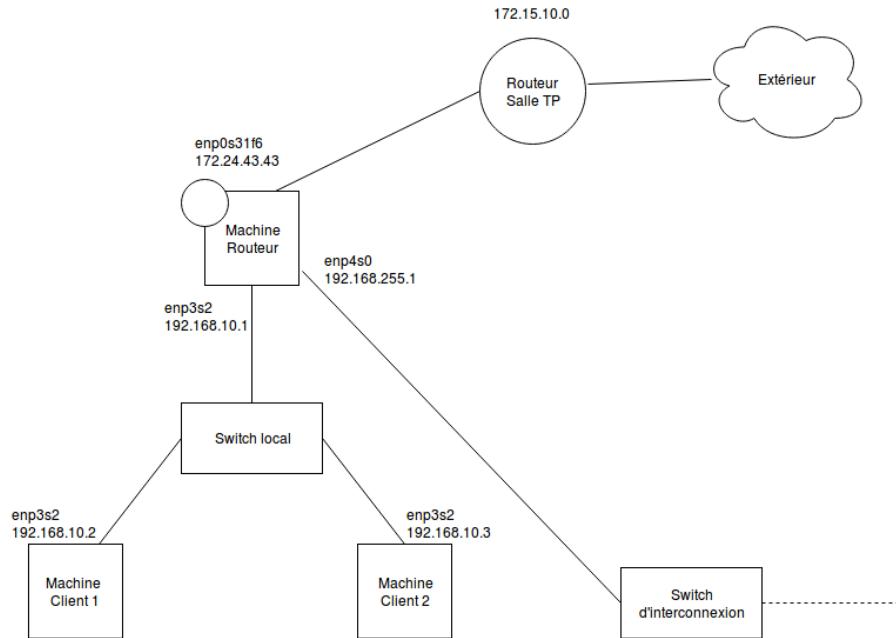


FIGURE 2.4 – architecture du sous réseau

#### — Table de routage des machines clientes

Destination	Passerelle	Masque	Interface
192.168.20.0	192.168.10.1	255.255.255.0	enp3s2
192.168.30.0	192.168.10.1	255.255.255.0	enp3s2

TABLE 2.1 – table de routage

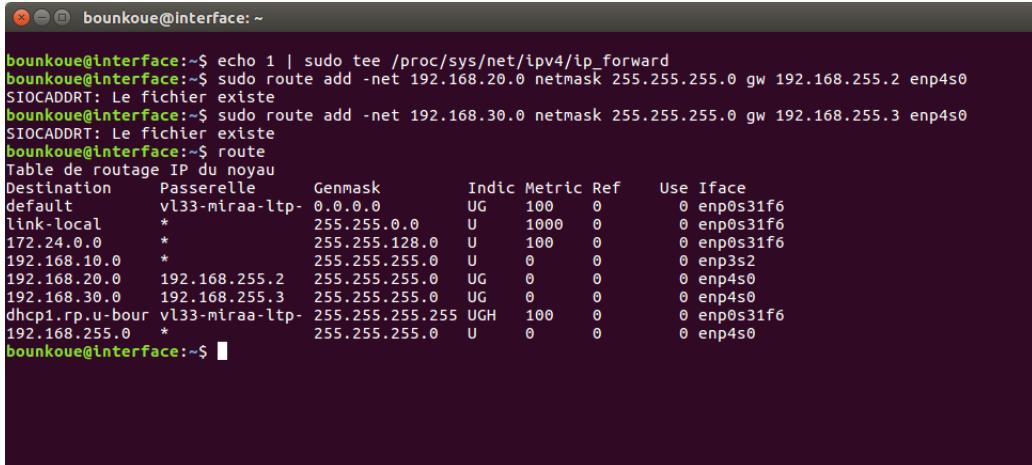
2. Nous devons communiquer l'adresse 192.168.255.1 aux autres routeurs pour que ces derniers puissent accéder à notre propre sous réseau.
3. — Les machines du sous réseau 192.168.20.0 (i.e du groupe 2) sont accessibles via la passerelle 192.168.255.2

- Celles du sous réseau 192.168.30.0 (i.e du groupe 3) sont accessibles via la passerelle 192.168.255.3
- Table de routage de notre routeur (en l'occurrence notre machine)

Destination	Passerelle	Masque	Interface
192.168.20.0	192.168.10.2	255.255.255.0	enp4s0
192.168.30.0	192.168.10.3	255.255.255.0	enp4s0

TABLE 2.2 – insert caption

### Intégration des tables de routage



```

bounkoue@interface:~$ echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
bounkoue@interface:~$ sudo route add -net 192.168.20.0 netmask 255.255.255.0 gw 192.168.255.2 enp4s0
SIOCADDRT: Le fichier existe
bounkoue@interface:~$ sudo route add -net 192.168.30.0 netmask 255.255.255.0 gw 192.168.255.3 enp4s0
SIOCADDRT: Le fichier existe
bounkoue@interface:~$ route
Table de routage IP du noyau
Destination     Passerelle      Genmask       Indic Metric Ref   Use Iface
default         vl33-miraa-ltp- 0.0.0.0    UG    100    0        0 enp0s31f6
link-local      *              255.255.0.0  U     1000   0        0 enp0s31f6
172.24.0.0      *              255.255.128.0 U     100    0        0 enp0s31f6
192.168.10.0    *              255.255.255.0 U     0       0        0 enp3s2
192.168.20.0    192.168.255.2 255.255.255.0 UG    0       0        0 enp4s0
192.168.30.0    192.168.255.3 255.255.255.0 UG    0       0        0 enp4s0
dhcpc1.rp.u-bour vl33-miraa-ltp- 255.255.255.255 UGH   100    0        0 enp0s31f6
192.168.255.0   *              255.255.255.0 U     0       0        0 enp4s0
bounkoue@interface:~$ 

```

FIGURE 2.5 – table de routage du routeur

### 2.3.3 Connexion internet des postes clients

1. Lorsqu'un client ping 8.8.8.8, l'adresse IP du poste ayant émis le message ICMP est celle du routeur sur le réseau de l'université.
2. La machine 8.8.8.8 est donc censée répondre au routeur.
3. Nous utilisons tous des adresses privées toutefois, aucune résolution NAT n'a été configurée pour les machines du sous réseau au niveau du routeur ce qui explique les machines du sous réseau aient perdu l'accès à internet.

### Activation du NAT

1. Il sert à convertir les adresses privées des machines de sont sous réseau en adresse publics

## 2. Commande d'activation du NAT

```
bounkoue@interface: ~
bounkoue@interface:~$ sudo iptables -t nat -A POSTROUTING -s 192.168.10.0/24 -o
enp0s31f6 -j MASQUERADE
bounkoue@interface:~$
```

FIGURE 2.6 – activation du NAT

3. A partir de la machine cliente d'adresse 192.168.10.2 on envoie une requête ping vers le serveur de google d'adresse 8.8.8.8. On effectue avec wireshark une capture sur le sous réseau et on observe sur que le serveur lui répond ce qui nous assure de la connectivité entre les machines du sous réseau et l'extérieur

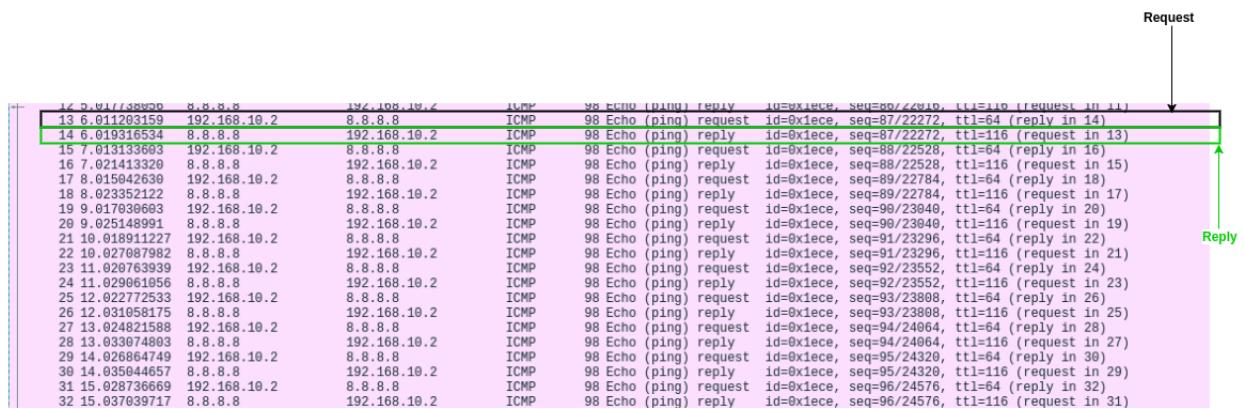


FIGURE 2.7 – test de la connectivité vers l'exterieur - capture sur le sous réseau

Si l'on examine la même communication en faisant une capture sur le réseau de l'université, on se rend compte qu'en réalité, le serveur de google communique avec notre routeur.

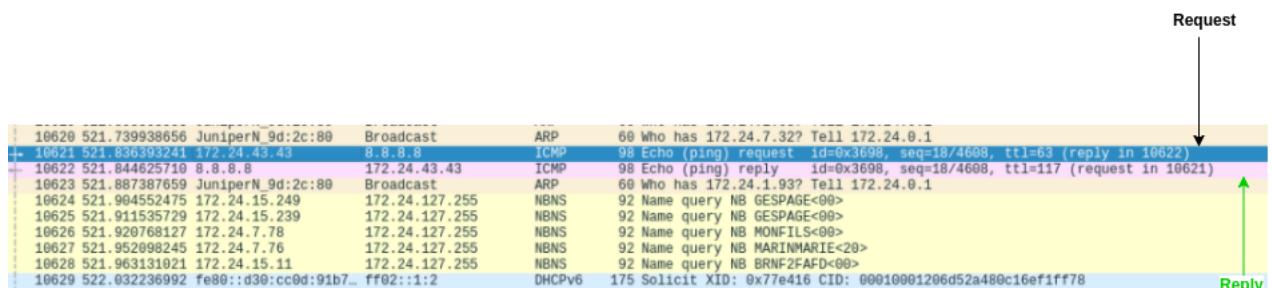


FIGURE 2.8 – test de la connectivité vers l'extérieur - capture sur le réseau de l'université

## Mise en place du nom de domaine

### Rappels du TP1

- Le fichier du système contenant la liste des serveurs DNS à utiliser pour convertir un nom de domaine en adresse IP est `/etc/resolv.conf`
- Nous confirmons la présence d'un serveur de nom de domaine en consultant le fichier précédent.

```
bounkoue@interface:~$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#      DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 127.0.1.1
search ensgt.u-bourgogne.fr
```

FIGURE 2.9 – contenu du fichier `/etc/resolv.conf`

- Test de l'effectivité de la résolution des noms

```
bounkoue@interface:~$ ping www.google.com
PING www.google.com (172.217.21.68) 56(84) bytes of data.
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=1 ttl=51 time=8.23 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=2 ttl=51 time=8.30 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=3 ttl=51 time=8.24 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=4 ttl=51 time=8.39 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=5 ttl=51 time=8.33 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=6 ttl=51 time=8.49 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=7 ttl=51 time=8.38 ms
64 bytes from fra07s31-in-f68.1e100.net (172.217.21.68): icmp_seq=8 ttl=51 time=8.31 ms
```

FIGURE 2.10 – test de la résolution de nom de domaine

### 2.3.4 Sauvegarde de la configuration

Les manipulations jusqu'à présent réalisées pour la plupart éphémères (excepté la mise en place du nom de domaine) : tout redémarrage ou désactivation / réactivation de l'interface efface l'adresse iP assignée. Pour sauvegarder la configuration sur nous allons éditer quelques fichiers utilisés au démarrage des services.

## Sauvegarde du mode routeur

Tout d'abord nous allons sauvegarder le fait que le poste de travail agit en mode routeur, en décommentant la ligne suivante dans le fichier `/etc/sysctl.conf`

```
1 net.ipv4.ip_forward=1
```

```
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.

#
#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

#####
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPV6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
# Enabling this option disables Stateless Address Autoconfiguration
# based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1

#####
# Additional settings - these settings can improve the network

```

Texte brut ▾ Largeur des tabulations : 8 ▾ Lig 28, Col 1 ▾ INS

FIGURE 2.11 – ligne décommentée

## Sauvegarde des paramètres des interfaces et politiques de routage

On rédige pour cela le script suivant qui sera exécuté au lancement de la machine :

```
1 auto enp0s31f6
2 iface enp0s31f6 inet static
3 address 172.24.43.43
4 netmask 255.255.128.0
5
6 auto enp3s2
7 iface enp3s2 inet static
8 address 192.168.10.1
9 netmask 255.255.255.0
10
11 auto enp4s0
12 iface enp4s0 inet static
13 address 192.168.255.1
14 netmask 255.255.255.0
15 up route add -net 192.168.20.0 netmask 255.255.0.0 gw 192.168.255.2
16 up route add -net 192.168.30.0 netmask 255.255.0.0 gw 192.168.255.3
```

Listing 2.1 – script de sauvegarde des paramètres d'interfaces et politiques de routage

# Chapitre 3

## TP3 : Configuration de serveurs

### 3.1 Objectif du TP

L'objectif de ce TP est de configurer différents services sur le poste routeur, afin de répondre à deux besoins : d'une part la nécessité de pouvoir configurer automatiquement les machines clientes qui se connectent au réseau privé de chaque groupe, et d'autre part la fourniture d'une solution d'hébergement de pages internet.

Nous reprenons les configurations sous réseau du TP2 :

- Adressse sous réseau : 192.168.10.0
- Masque : 255.255.255.0
- Notre adresse (routeur) : 192.168.10.1
- Adresse Client 1 : 192.168.10.2
- Adresse Client 2 : 192.168.10.3

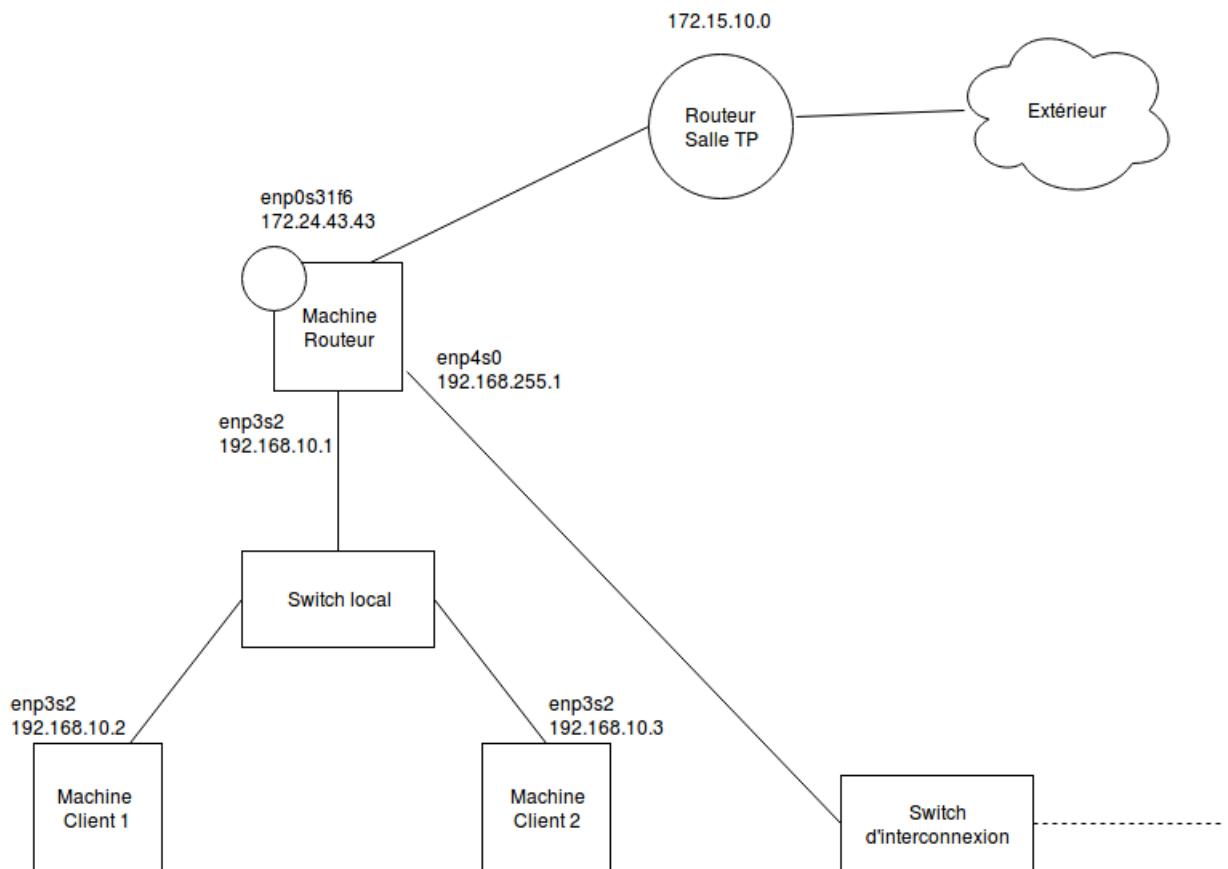


FIGURE 3.1 – Architecture sous réseau

## 3.2 Mise en place d'un serveur DHCP

### 3.2.1 Comprendre le DHCP

#### DHCP

DHCP signifie **Dynamic Host Configuration Protocol**. Le protocole DHCP Permet à un hôte d'obtenir une adresse IP sans que l'administrateur ait à définir un profil pour chaque équipement. Avec le protocole DHCP il suffit qu'une plage d'adresses IP soit définie sur un serveur DHCP. Lorsque les ordinateurs se connectent, ils communiquent avec le serveur DHCP et demandent une adresse. Le Serveur DHCP choisit une adresse et l'affecte à l'ordinateur hôte.

#### Fonctionnement

Lorsqu'un client DHCP initialise un accès à un réseau , le processus d'obtention du bail IP se déroule en 4 phases :

1. Le client émet un message de demande de bail IP (DHCPDISCOVER) qui est envoyé sous forme d'une diffusion sur le réseau avec adresse IP source 0.0.0.0 et adresse IP destination 255.255.255.255 et adresse MAC.
2. Les serveurs DHCP répondent en proposant une adresse IP avec une durée de bail et l'adresse IP du serveur DHCP (DHCoffer)
3. Le client sélectionne la première adresse IP (s'il y a plusieurs serveurs DHCP) reçue et envoie une demande d'utilisation de cette adresse au serveur DHCP (DHCPREQUEST). Son message envoyé par diffusion comporte l'identification du serveur sélectionné qui est informé que son offre a été retenue ; tous les autres serveurs DHCP retirent leur offre et les adresses proposée redeviennent disponibles.
4. Le serveur DHCP accuse réception de la demande et accorde l'adresse en bail (DHCPACK), les autres serveurs retirent leur proposition. Enfin le client utilise l'adresse pour se connecter au réseau.

### 3.2.2 Définition des paramètres du serveur DHCP

#### Emplacement relatif du serveur DHCP

Notre machine abrite le routeur du réseau local, donc le serveur DHCP doit logiquement se situer sur notre machine, car le routeur doit posséder une adresse IP fixe à tout moment.

#### Machines devant être configurées

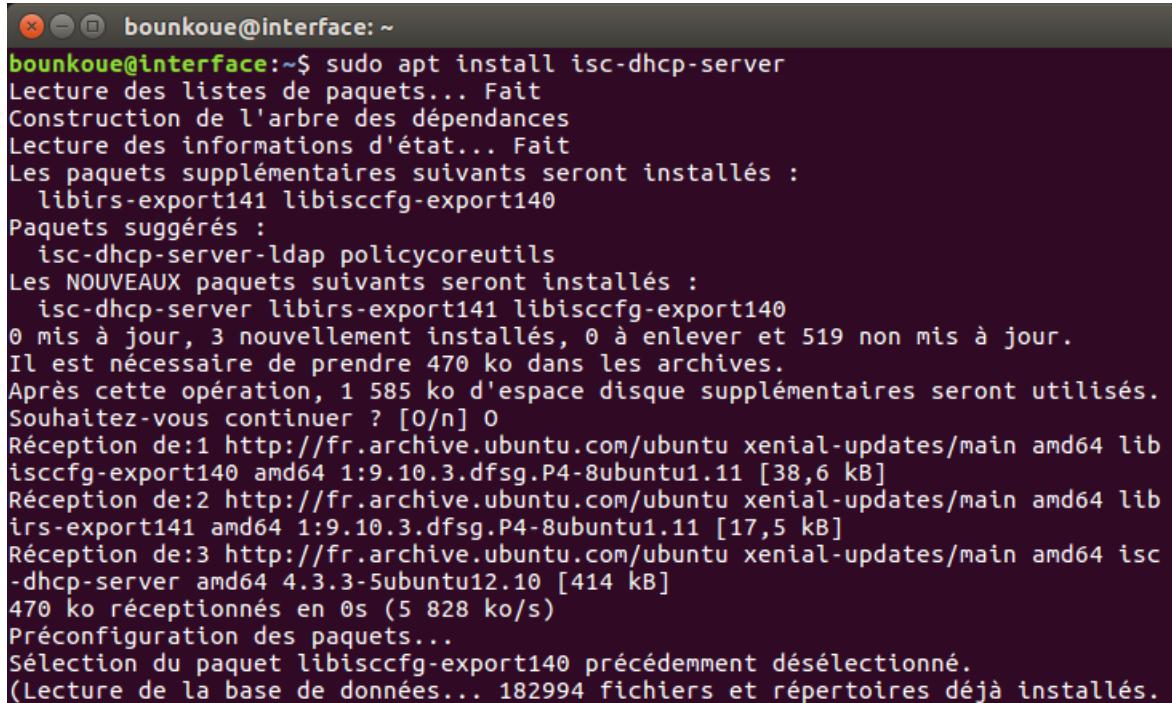
Les machines qui doivent être configurées par le serveur sont les machines clients du réseau local soit les machines d'adresses 192.168.10.2 et 192.168.10.3. Ces machines sont accessibles à travers l'interface enp3s2

#### Paramètres à communiquer aux clients

- Adresse Broadcast : 192.168.10.255
- Adresse passerelle : 192.168.10.1
- Masque de sous réseau : 255.255.255.0
- Adresse des serveurs DNS disponibles : 193.50.50.2 193.50.50.6
- Les adresses IP utilisables par les machines clientes sont : 192.168.10.2 à 192.168.10.254

### 3.2.3 Mise en place du serveur DHCP

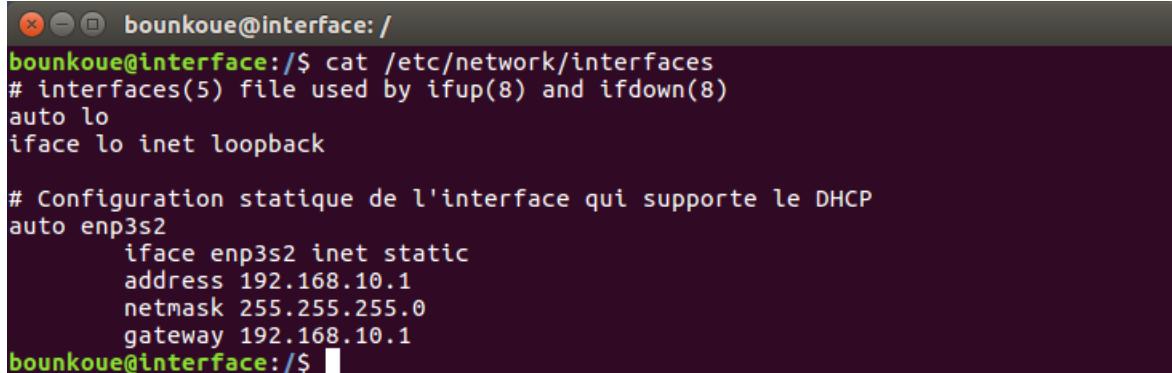
Installation du package isc-dhcp-server



```
bounkoue@interface: ~
bounkoue@interface:~$ sudo apt install isc-dhcp-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  libirs-export141 libiscfg-export140
Paquets suggérés :
  isc-dhcp-server-ldap policycoreutils
Les NOUVEAUX paquets suivants seront installés :
  isc-dhcp-server libirs-export141 libiscfg-export140
0 mis à jour, 3 nouvellement installés, 0 à enlever et 519 non mis à jour.
Il est nécessaire de prendre 470 ko dans les archives.
Après cette opération, 1 585 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [0/n] 0
Réception de:1 http://fr.archive.ubuntu.com/ubuntu xenial-updates/main amd64 lib
iscfg-export140 amd64 1:9.10.3.dfsg.P4-8ubuntu1.11 [38,6 kB]
Réception de:2 http://fr.archive.ubuntu.com/ubuntu xenial-updates/main amd64 lib
irs-export141 amd64 1:9.10.3.dfsg.P4-8ubuntu1.11 [17,5 kB]
Réception de:3 http://fr.archive.ubuntu.com/ubuntu xenial-updates/main amd64 isc
-dhcp-server amd64 4.3.3-5ubuntu12.10 [414 kB]
470 ko réceptionnés en 0s (5 828 ko/s)
Préconfiguration des paquets...
Sélection du paquet libiscfg-export140 précédemment désélectionné.
(Lecture de la base de données... 182994 fichiers et répertoires déjà installés.
```

FIGURE 3.2 – Installation du serveur dhcp

Vérification de la configuration statique de l'interface à utiliser



```
bounkoue@interface: /
bounkoue@interface:/$ cat /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

# Configuration statique de l'interface qui supporte le DHCP
auto enp3s2
iface enp3s2 inet static
  address 192.168.10.1
  netmask 255.255.255.0
  gateway 192.168.10.1
bounkoue@interface:/$
```

FIGURE 3.3 – Vérification de l'interface

Nous avons édité le fichier `/etc/network/interfaces`, pour configurer de manière statique l'interface à utiliser par le serveur DHCP.

#### Indication du nom de l'interface à utiliser par le serveur DHCP

Nous nous sommes rendu dans le fichier `/etc/dhcp/isc-dhcp-server` pour y préciser l'interface à utiliser par le serveur DHCP. Nous avons précisé l'interface `enp3s2` car c'est l'interface de connexion avec notre sous réseau.

```

isc-dhcp-server
/etc/default

# Defaults for isc-dhcp-server initscript
# sourced by /etc/init.d/isc-dhcp-server
# installed at /etc/default/isc-dhcp-server by the maintainer scripts

#
# This is a POSIX shell fragment
#

# Path to dhcpcd's config file (default: /etc/dhcp/dhcpcd.conf).
#DHCPD_CONF=/etc/dhcp/dhcpcd.conf

# Path to dhcpcd's PID file (default: /var/run/dhcpcd.pid).
#DHCPD_PID=/var/run/dhcpcd.pid

# Additional options to start dhcpcd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpcd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACES="enp3s2"

```

FIGURE 3.4 – Indication interface

### Configuration du serveur DHCP

Nous nous sommes rendu dans le fichier **etc/dhcp/dhcpcd.conf** et nous y avons ajouté les lignes suivantes :

```

##### Configuration sous réseau #####
subnet 192.168.10.0 netmask 255.255.255.0 {
    range 192.168.10.2 192.168.10.254;
    option broadcast-address 192.168.10.255;
    option routers 192.168.10.1;
    option domain-name-servers 193.50.50.2, 193.50.50.6;
}

```

FIGURE 3.5 – Configuration serveur DHCP

### Test configuration client

On test la connexion client et tout va bien.

```
bounkoue@interface:~$ clear
bounkoue@interface:~$ sudo dhclient
[sudo] Mot de passe de bounkoue :
RTNETLINK answers: File exists
bounkoue@interface:~$
```

FIGURE 3.6 – Test configuration client

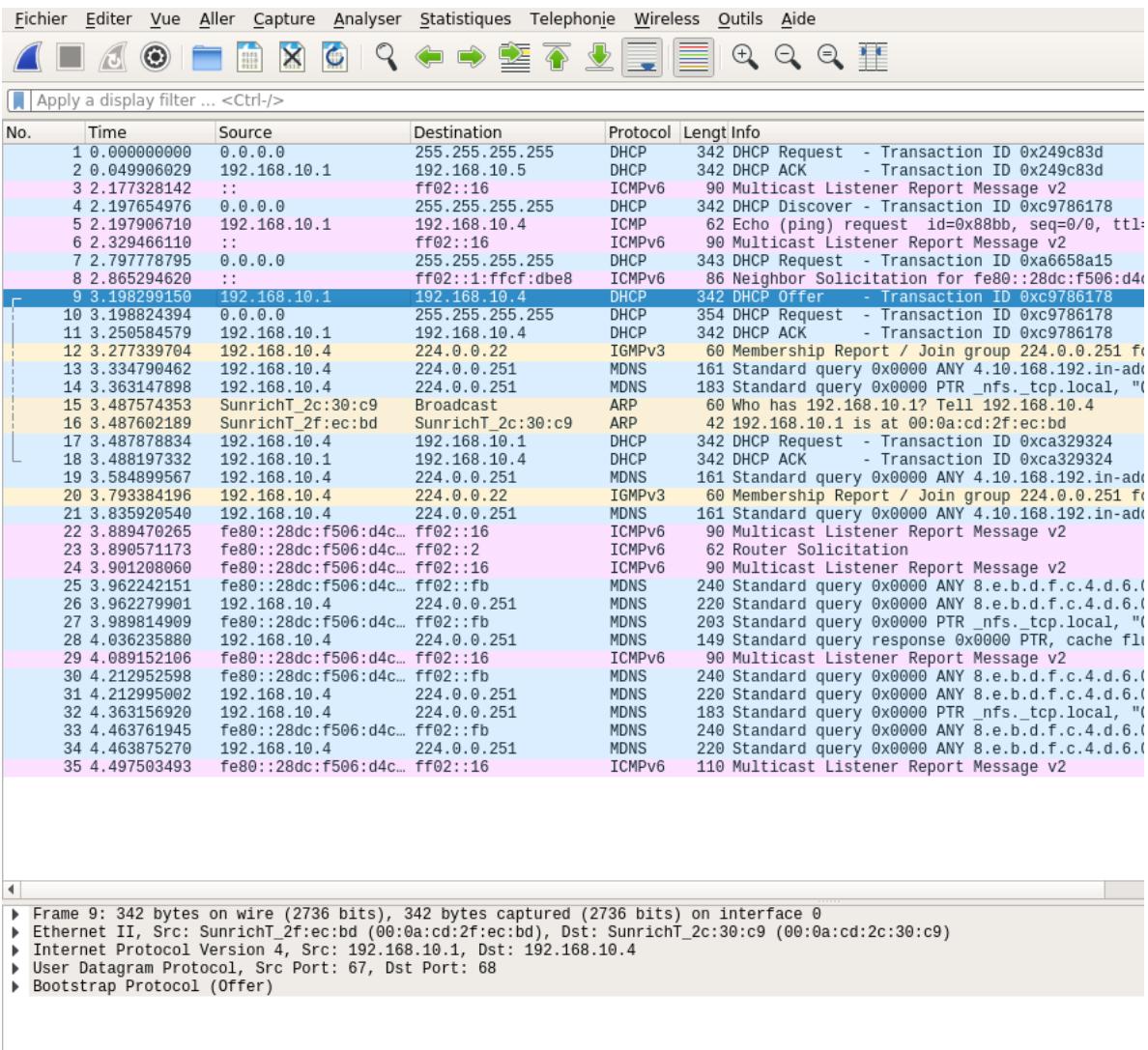


FIGURE 3.7 – visualisation fonctionnement DHCP

### 3.3 Mise en place d'un serveur Web + SQL

#### 3.3.1 Installation des packages

- apache2 pour le serveur web
- mysql-server pour le serveur de bd
- php7

#### 3.3.2 Mise en place du serveur Apache2

##### Configuration

On garde les paramètres par défaut lors de l'installation. On ne modifie pas les fichiers `/etc/apache2/apache2.conf` et `/etc/apache2/ports.conf`

##### Lancement du serveur Apache

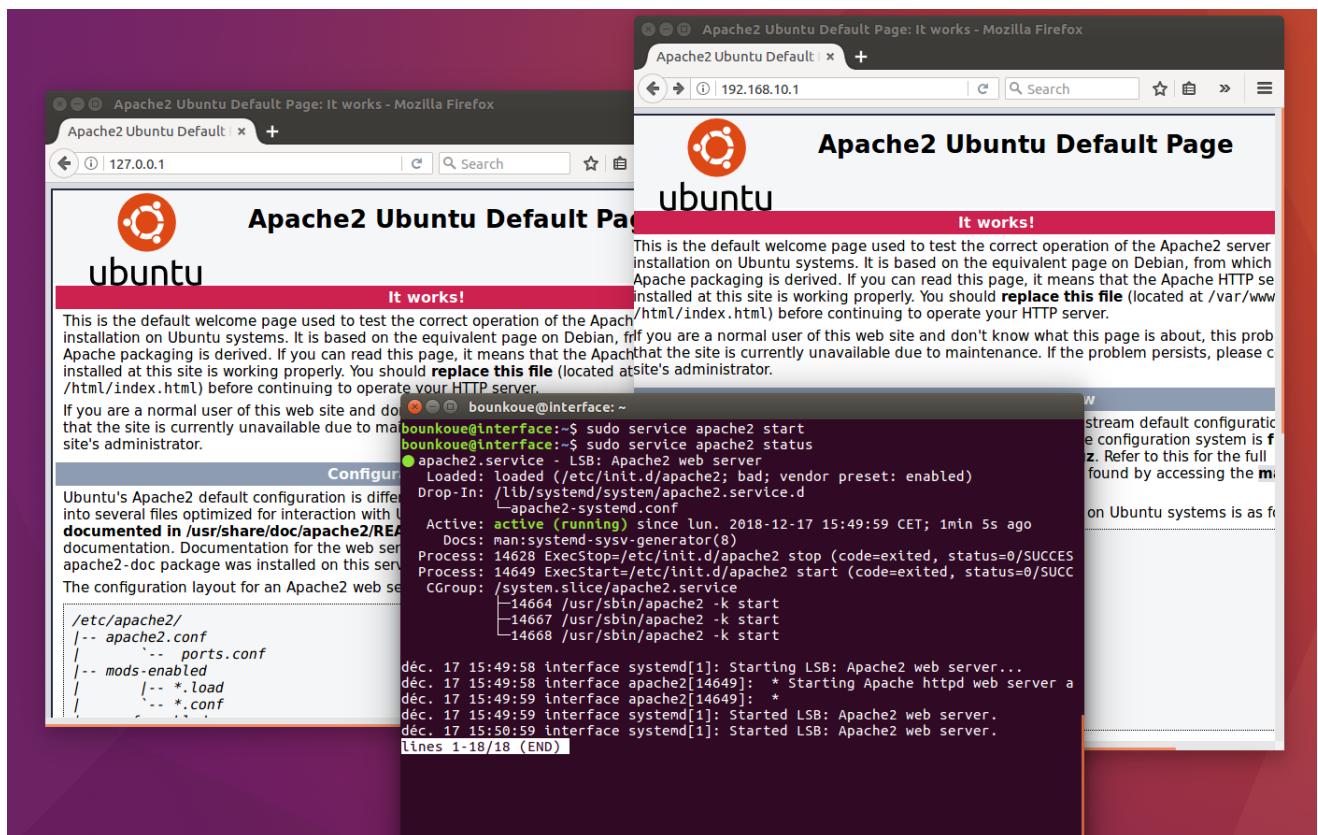


FIGURE 3.8 – Lancement du serveur apache

## Edition du site internet

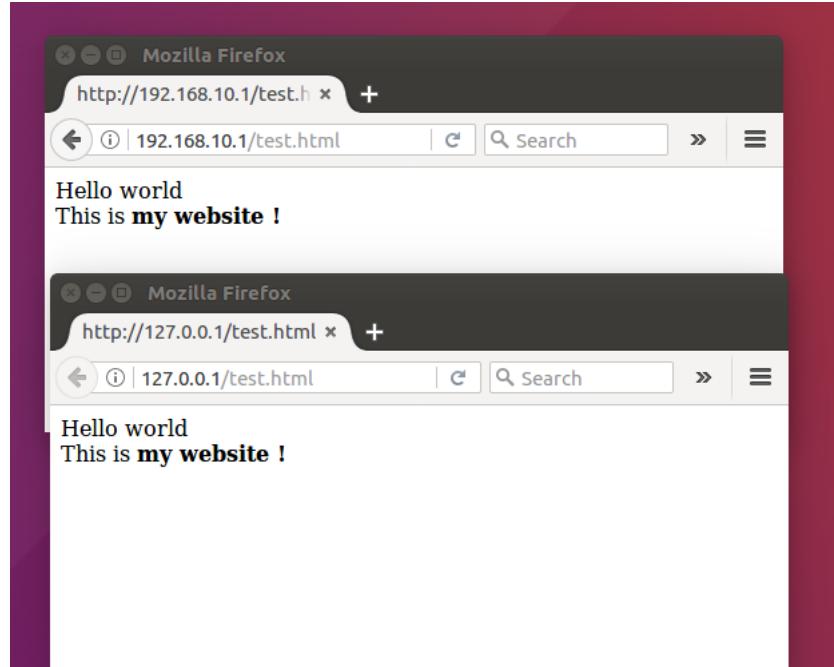


FIGURE 3.9 – Visualisation site internet

### 3.3.3 Mise en place du serveur MySQL

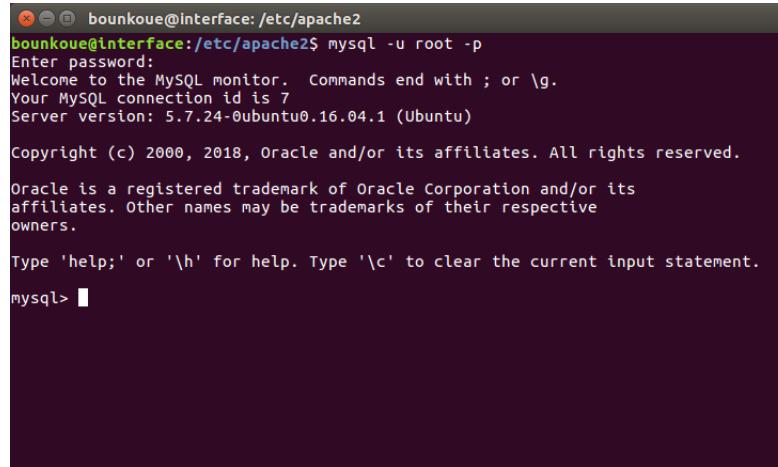
#### Lancement du serveur MySQL

```
bounkoue@interface:/etc/apache2$ service mysql status
● mysql.service - MySQL Community Server
  Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: en
  Active: active (running) since lun. 2018-12-17 15:01:28 CET; 57min ago
    Main PID: 11577 (mysqld)
      CGroup: /system.slice/mysql.service
              └─11577 /usr/sbin/mysqld

déc. 17 15:01:27 interface systemd[1]: Starting MySQL Community Server...
déc. 17 15:01:28 interface systemd[1]: Started MySQL Community Server.
lines 1-9/9 (END)
```

FIGURE 3.10 – Lancement du serveur mysql

## Connexion au serveur MySQL



The screenshot shows a terminal window with a dark background and light-colored text. At the top, it displays the user's name and the command used to connect:

```
bounkoue@interface:/etc/apache2$ mysql -u root -p
```

It then prompts for a password and provides information about the MySQL monitor and connection details:

```
Enter password:  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 7  
Server version: 5.7.24-0ubuntu0.16.04.1 (Ubuntu)  
  
Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.
```

Finally, it shows the MySQL prompt:

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> █
```

FIGURE 3.11 – Connexion au serveur mysql

# Chapitre 4

## TP4 : Firewall

### 4.1 Objectif du TP

Dans ce dernier TP nous nous intéressons à la mise en place de pare-feu pour machines clientes et pour routeurs sous linux sans passer par une interface graphique. Cela est possible grâce à l'outil `iptables` que nous explorerons et dont nous découvrirons les différentes fonctionnalités tout au long du TP.

### 4.2 Exercice 1

Iptables permet le filtrage de paquets avec la table filter (chaîne INPUT, OUTPUT, FORWARD), permet la translation d'adresse avec la table NAT (chaîne PREROUTING, OUTPUT POSTROUTING) et permet la modification de paquets avec la table mangle.

1. Définition de la politique par défaut d'une chaîne iptables (INPUT, OUTPUT, FORWARD) La syntaxe est la suivante :

```
1 iptables -P <chaine> -j <action>
```

où <chaine> peut être INPUT, OUTPUT ou FORWARD et <action> peut-être ACCEPT ou DROP

Par exemple, ci-dessous est présenté comment est définie une politique par défaut à DROP :

```
1 iptables -P INPUT -j DROP
2 iptables -P OUTPUT -j DROP
3 iptables -P FORWARD -j DROP
```

2. Proposition d'une politique par défaut pour les chaînes INPUT, OUTPUT, FORWARD

— Machine de bureau

```
1 iptables -P INPUT -j ACCEPT
2 iptables -P OUTPUT -j ACCEPT
3 iptables -P FORWARD -j DROP
```

— Routeur filtrant

```
1 iptables -P INPUT -j DROP
2 iptables -P OUTPUT -j DROP
3 iptables -P FORWARD -j ACCEPT
```

3. Autoriser les paquets ayant pour port source le port 80 à entrer sur une machine

```
1 iptables -A INPUT -p tcp --sport 80 -j ACCEPT
```

4. Autoriser les paquets ayant pour port source le port 80 à traverser sur une machine

```
1 iptables -A FORWARD -p tcp --sport 80 -j ACCEPT
```

5. Autoriser les paquets ayant pour port source le port 53 et ayant pour adresse source 193.50.50.2 à entrer sur une machine

```
1 iptables -A INPUT -p tcp -s 193.50.50.2 --sport 53 -j ACCEPT
```

6. Autoriser les paquets ayant pour port source le port 53 et ayant pour adresse source 193.50.50.2 à traverser sur une machine  

```
1 iptables -A FORWARD -p tcp -s 193.50.50.2 --sport 53 -j ACCEPT
```
7. Autoriser les paquets ayant pour port source le port 53 et ayant pour adresse source 193.50.50.2 à traverser sur une machine  

```
1 iptables -A FORWARD -p tcp -s 193.50.50.2 --sport 53 -j ACCEPT
```
8. N'autoriser en entré QUE les paquets relatifs à une connexion déjà existante  

```
1 iptables -A INPUT -m conntrack --state RELATED,ESTABLISHED -j ACCEPT
```
9. N'autoriser en traversé QUE les paquets relatifs à une connexion déjà existante  

```
1 iptables -A FORWARD -m conntrack --state RELATED,ESTABLISHED -j ACCEPT
```
10. L'outil **netstat** peut être utilisé pour connaître les ports qui sont utilisé pendant un temps donné.

## 4.3 Exercice 2

En utilisant la politique par défaut à DROP, Nous présentons ci-dessous les commandes permettant de :

### 4.3.1 Vérification de la table de iptables

```
1 sudo iptables -P OUTPUT DROP
2 sudo iptables -P FORWARD DROP
3 sudo iptables -L
```

```
bounkoue@interface: ~
bounkoue@interface:~$ sudo iptables -P OUTPUT DROP
bounkoue@interface:~$ sudo iptables -P FORWARD DROP
bounkoue@interface:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
          prot opt source               destination
ACCEPT    tcp  --  anywhere             anywhere            tcp spt:http
Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT    tcp  --  anywhere             anywhere            tcp spt:http
Chain OUTPUT (policy DROP)
target     prot opt source               destination
bounkoue@interface:~$
```

FIGURE 4.1 – configuration initiale

1. Pour visualiser les pages web des autres serveurs depuis notre serveur (en utilisant l'adresse IP)

```
1 iptables -A INPUT -s 192.168.255.0 -p all -j ACCEPT
2 iptables -A OUTPUT -d 192.168.255.0 -p all -j ACCEPT
```

2. Pour visualiser les pages web des autres serveurs depuis notre client (en utilisant l'adresse IP)

```
1 iptables -A FORWARD -s 192.168.255.0 -p all -j ACCEPT
2 iptables -A FORWARD -d 192.168.255.0 -p all -j ACCEPT
```

3. Pour visualiser la page d'accueil de google depuis notre serveur (en utilisant l'adresse IP)

```
1 adr_google=$(nslookup www.google.com | tail -n2 | cut -d: -f2)
2 iptables -A INPUT -s $adr_google -p all -j ACCEPT
3 iptables -A OUTPUT -d $adr_google -p all -j ACCEPT
```

4. Pour visualiser la page d'accueil de google depuis notre client (en utilisant l'adresse IP)

```
1 adr_google=$(nslookup www.google.com | tail -n2 | cut -d: -f2)
2 iptables -A FORWARD -s $adr_google -p all -j ACCEPT
3 iptables -A FORWARD -d $adr_google -p all -j ACCEPT
```

5. Pour visualiser la page d'accueil de google depuis notre serveur (en utilisant le nom www.google.fr)

```

1 iptables -A INPUT -p tcp --sport 53 -j ACCEPT
2 iptables -A OUTPUT -p udp --sport 53 -j ACCEPT
3 iptables -A INPUT -p tcp --dport 53 -j ACCEPT
4 iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
5 #Puis on autorise www.google.fr
6 iptables -A INPUT -s www.google.fr -p all -j ACCEPT
7 iptables -A OUTPUT -d www.google.fr -p all -j ACCEPT

```

6. Pour visualiser la page d'accueil de google depuis notre client (en utilisant le nom www.google.fr)

```

1 iptables -A FORWARD -p tcp --sport 53 -j ACCEPT
2 iptables -A FORWARD -p udp --sport 53 -j ACCEPT
3 iptables -A FORWARD -p tcp --dport 53 -j ACCEPT
4 iptables -A FORWARD -p udp --dport 53 -j ACCEPT
5 #Puis on autorise www.google.fr
6 iptables -A FORWARD -s www.google.fr -p all -j ACCEPT
7 iptables -A FORWARD -d www.google.fr -p all -j ACCEPT

```

7. Script pour stocker les règles de filtrage

```

1#!/bin/bash
2iptables-restore < /etc/iptables.test.rules
3
4iptables -A INPUT -s 192.168.255.0 -p all -j ACCEPT
5iptables -A OUTPUT -d 192.168.255.0 -p all -j ACCEPT
6
7iptables -A FORWARD -s 192.168.255.0 -p all -j ACCEPT
8iptables -A FORWARD -d 192.168.255.0 -p all -j ACCEPT
9
10adr_google=$(nslookup www.google.com | tail -n2 | cut -d: -f2)
11
12iptables -A FORWARD -s $adr_google -p all -j ACCEPT
13iptables -A FORWARD -d $adr_google -p all -j ACCEPT
14
15iptables -A FORWARD -s $adr_google -p all -d 192.168.10.1 -j ACCEPT
16iptables -A FORWARD -d $adr_google -p all -s 192.168.10.1 -j ACCEPT
17
18iptables -A INPUT -p tcp --sport 53 -j ACCEPT
19iptables -A OUTPUT -p udp --sport 53 -j ACCEPT
20iptables -A INPUT -p tcp --dport 53 -j ACCEPT
21iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
22
23#Puis on autorise www.google.fr pour serveur
24iptables -A INPUT -s www.google.fr -p all -j ACCEPT
25iptables -A OUTPUT -d www.google.fr -p all -j ACCEPT
26iptables -A FORWARD -p tcp --dport 53 -j ACCEPT
27iptables -A FORWARD -p udp --dport 53 -j ACCEPT
28
29#Puis on autorise www.google.fr pour client
30iptables -A FORWARD -s www.google.fr -p all -j ACCEPT
31iptables -A FORWARD -d www.google.fr -p all -j ACCEPT

```

Listing 4.1 – script règles de filtrage iptables.sh

Ensuite on doit déplacer le script iptables.sh dans /etc/network/if-pre-up.d/ et puis le rendre executable.

```

1 sudo mv /emplacement/du/script/iptables.sh /etc/network/if-pre-up.d/
2 sudo chmod +x /etc/network/if-pre-up.d/iptables.sh

```

8. Script pour effacer les règles de filtrages et autoriser tout le traffic

```

1#!/bin/bash
2iptables-restore < /etc/iptables.test.rules
3
4sudo iptables -F
5sudo iptables -X
6
7iptables -A INPUT -j ACCEPT
8iptables -A OUTPUT -j ACCEPT
9iptables -A FORWARD -j ACCEPT

```

Listing 4.2 – script effacer règles de filtrage et tout autoriser iptables\_allow.sh

Ensuite on doit déplacer le script `iptables_allow.sh` dans `/etc/network/if-pre-up.d/` et puis le rendre executable.

```
1 sudo mv /emplacement/du/script/iptables_allow.sh /etc/network/if-pre-up.d/
2 sudo chmod +x /etc/network/if-pre-up.d/iptables_allow.sh
```

9. Le fichier `/etc/services` contient une table de correspondance entre les numéros de ports et les programmes. Il permet l'identification d'une application à partir de son numéro de port. Nous présentons un extrait de son contenu ci-dessous

```
#=====
# The remaining port numbers are not as allocated by IANA.
#=====

# Kerberos (Project Athena/MIT) services
# Note that these are for Kerberos v4, and are unofficial. Sites running
# v4 should uncomment these and comment out the v5 entries above.
#
kerberos4    750/udp      kerberos-iv kdc # Kerberos (server)
kerberos4    750/tcp       kerberos-iv kdc
kerberos-master 751/udp   kerberos_master # Kerberos authentication
kerberos-master 751/tcp
passwd-server 752/udp     passwd_server # Kerberos passwd server
krb-prop      754/tcp      krb_prop krb5_prop hprop # Kerberos slave propagation
krbupdate     760/tcp      kreg        # Kerberos registration
swat          901/tcp      # swat
kpop          1109/tcp     # Pop with Kerberos
knethd       2053/tcp     # Kerberos de-multiplexor
```

FIGURE 4.2 – contenu fichier services

## 4.4 Exercice 3

1. Il est possible avec `iptables` de limiter (par heure par exemple) le nombre de paquets arrivant sur un port donné. Il suffit d'utiliser le module supplémentaire `limit`. Pour utiliser un module d'option de concordance, il faut charger le module en l'appelant par son nom à l'aide de l'option `-m`. comme ceci : `-m <module-name>`.

Ainsi pour limiter à 10 par exemple le nombre de paquets arrivant à un port `port` on utilise la commande

```
1 iptables -A INPUT -m limit --limit 10/hour -p <port> -j ACCEPT
```

2. Il est bien possible aussi avec `iptables` de limiter (par heure par exemple) le nombre de paquets arrivant sur port `<port>` donné en provenance de trois sources d'adresses X.X.X.X, Y.Y.Y.Y, Z.Z.Z.Z connues à l'avance.

```
1 iptables -A INPUT -m limit --limit 10/hour -s X.X.X.X, Y.Y.Y.Y, Z.Z.Z.Z -p <port> -j ACCEPT
```

3. Toutefois il ne semble pas possible avec `iptables` de limiter (par heure par exemple) le nombre de paquets arrivant sur port `<port>` donné en provenance de X sources d'adresses inconnues à l'avance.

4. Le temps ne nous a malheureusement pas permis d'appliquer ces règles en salle de TP sur une machine cliente.

5. Expliquons chacune des lignes de règles

6. Définition **port knocking** :

Le toilage à la porte, ou port-knocking, est une méthode permettant de modifier le comportement d'un pare-feu (firewall) en temps réel en provoquant l'ouverture de ports permettant la communication, grâce au lancement préalable d'une suite de connexions sur des ports distincts dans le bon ordre, à l'instar d'un code frappé à une porte.

Cette technique est notamment utilisée pour protéger l'accès au port 22 dédié au Secure shell (SSH), elle ne nécessite pas beaucoup de ressources et reste facile à mettre en œuvre.

7. Ces règles ouvriront le port 22 pendant 5 secondes après qu'on ait frappé les ports 100, 200, 300 et 400 avec des paquets TCP.

# Table des figures

1.1	résultat ifconfig . . . . .	3
1.2	résultat ifconfig enp0s31f6 . . . . .	3
1.3	ping vers google . . . . .	5
1.4	resultat netstat . . . . .	5
1.5	resultat netstat -n . . . . .	6
1.6	netstat -inet -udp . . . . .	6
1.7	table de routage I . . . . .	6
1.8	resultat netstat -a . . . . .	7
1.9	table des sockets en attente de connexions : netstat -l . . . . .	8
1.10	table de routage . . . . .	8
1.11	ligne correspondant au traffic de la salle de TP . . . . .	8
1.12	ligne correspondant au traffic exterieur . . . . .	9
1.13	nslookup executé sur www.yahoo.fr . . . . .	9
1.14	contenu du fichier /etc/resolv.conf . . . . .	9
1.15	contenu du fichier /etc/hosts . . . . .	9
1.16	ajout de deux lignes au fichier /etc/hosts . . . . .	10
1.17	ping avec les deux nouveaux noms de machines . . . . .	10
1.18	ping vers les machines de la salle . . . . .	11
1.19	ping vers www.yahoo.fr . . . . .	11
1.20	capture des paquets issues du ping avec wireshark . . . . .	12
1.21	détails trame capturée . . . . .	12
1.22	visualisation d'un ttl épuisé au terminal . . . . .	13
1.23	résultats traceroute vers bonifacio.fr . . . . .	13
1.24	résultats traceroute vers yahoo.fr . . . . .	14
1.25	table de routage . . . . .	14
2.1	architecture du réseau à créer . . . . .	15
2.2	attribution de l'adresse IP et du masque . . . . .	16
2.3	ping dans le sous réseau . . . . .	17
2.4	architecture du sous réseau . . . . .	17
2.5	table de routage du routeur . . . . .	18
2.6	activation du NAT . . . . .	19
2.7	test de la connectivité vers l'exterieur - capture sur le sous réseau . . . . .	19
2.8	test de la connectivité vers l'extérieure - capture sur le réseau de l'université . . . . .	19
2.9	contenu du fichier /etc/resolv.conf . . . . .	20
2.10	test de la résolution de nom de domaine . . . . .	20
2.11	ligne décommentée . . . . .	21
3.1	Architecture sous réseau . . . . .	22
3.2	Installation du serveur dhcp . . . . .	24
3.3	Vérification de l'interface . . . . .	24
3.4	Indication interface . . . . .	25
3.5	Configuration serveur DHCP . . . . .	25
3.6	Test configuration client . . . . .	26
3.7	visualisation fonctionnement DHCP . . . . .	26
3.8	Lancement du serveur apache . . . . .	27
3.9	Visualisation site internet . . . . .	28
3.10	Lancement du serveur mysql . . . . .	28
3.11	Connexion au serveur mysql . . . . .	29

4.1	configuration initiale . . . . .	31
4.2	contenu fichier services . . . . .	33

# Liste des tableaux

1.1	relevé des informations sur les ports . . . . .	4
1.2	vitesse de transfert des interfaces . . . . .	4
2.1	table de routage . . . . .	17
2.2	insert caption . . . . .	18