
Administration Linux & Services – Compte Rendu

Table des matières

Administration Linux & Services – Compte Rendu.....	1
Introduction	2
Avant de commencer	2
Organisation et mise en place des équipements	3
a) Organisation	3
b) Mise en place des équipements.....	4
Configuration du réseau.....	5
a) Configurations statiques du réseau	5
b) Mise en place des VLANs.....	6
c) Mise en place du serveur DHCP et du DHCP Relay	7
d) Mise en place du serveur DNS	11
e) Test du serveur DNS.....	15
Tests de connectivité	16
2. Vérification de l'adressage.....	16
f) Connectivité interne.....	18
g) Connectivité vers l'extérieur	20
Configuration du Pare-feu	22
a) Configuration de base de la machine FIREWALL.....	22
b) Durcissement du filtrage au sein du réseau.....	26
Configuration de la gestion des utilisateurs	28
a) Mise en place des groupes et des utilisateurs sur les machines m1, m2, m3, m4	28
b) Mise en place du serveur NFS	30
c) Mise en place des dossiers partagés sur les machines m1, m2, m3, m4	31
Configuration du serveur Web.....	33
a) Vérification de communication entre les VM et la machine linux hôte.....	33
b) Installation des services	33
c) Start up des services	35
Exploitation	39
a) Problème initial sur la mise en place de cette partie.....	39
b) Mise en place	39
Pour aller plus loin... ..	40
a) Scripting	40
b) Amélioration du firewall, filtrage webgateway.....	41
c) Configuration du DDNS	41
d) Lancement automatique des services avec rc.local	42

Introduction

Ce projet a été réalisé en groupe de 4 sur Marionnet. L'objectif du projet est de réaliser une infrastructure d'entreprise mêlant plusieurs éléments communément utilisés. Pour mieux comprendre nous allons étudier le schéma logique Marionnet de ce projet, puis rentrer plus en détails dans chacune des configurations (VLAN, DHCP, DNS, FIREWALL, NFS, SERVEUR WEB, EXPLOITATION).

Avant de commencer

Notre projet est trouvable entièrement à l'adresse GitLab ci-dessous avec le readme associé, les scripts et le fichier marionnet :

https://gitlab.com/antoine_ansari/projet_admin_linux_ansari_vovard_douailly_tancev

Il est également téléchargeable via le lien swisstransfer suivant (valable jusqu'au 11/05/2021) :

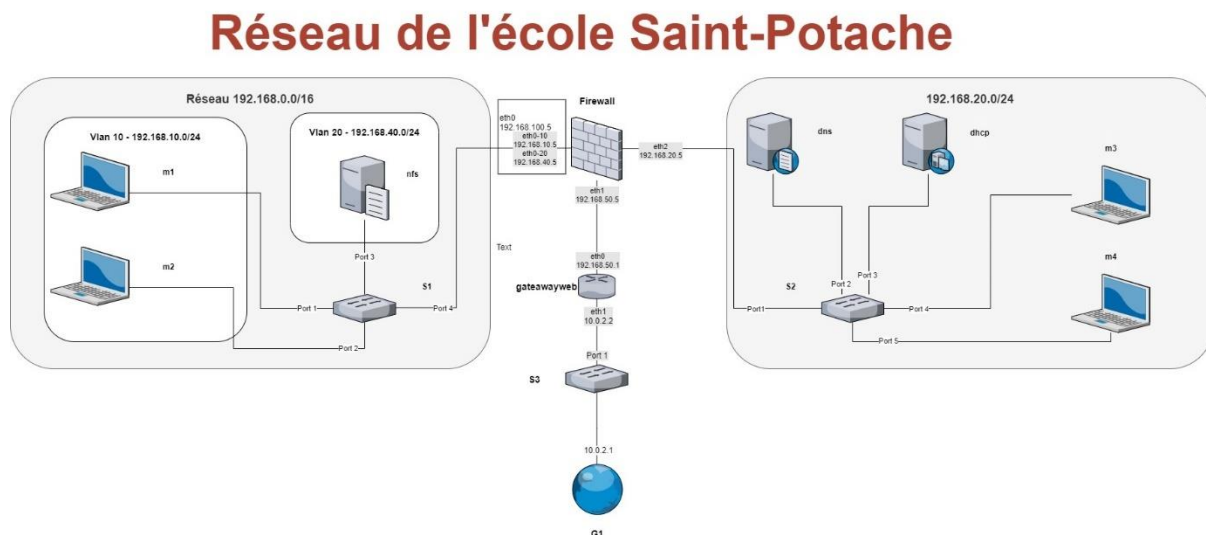
<https://www.swisstransfer.com/d/d92e5e60-a83a-4729-9ead-e6d01d123b52>

Nous avons établi une procédure de lancement pour exécuter notre réseau marionnet à retrouver dans le fichier readme.md en annexe.

Organisation et mise en place des équipements

a) Organisation

Pour commencer le projet, nous avons décidé de réaliser un schéma logique avec draw.io. Le schéma ci-dessous correspond ainsi à l'infrastructure du réseau de l'école Saint-Potache que nous avons mis en place.



Afin de nous organiser nous avons décidé de choisir comme support OneNote. Nous avons pu partager nos notes et nos avancements en trois sections chacune scindées en sous sections.



Dans **infos de bases** on retrouve :

- **Schéma logique** pour mieux nous repérer.
- **Tables IPs**, afin d'avoir un suivi des adresses attribués, mais aussi des changements qui sont fait au cours du projet.
- **Rôle des machines**, dans laquelle on trouve une définition de chaque machine afin de voir quel rôle celle-ci aura dans l'architecture.
- **Avancement** du projet sur lequel on trouve une to-do liste afin de pouvoir avoir un suivi complet.

Dans **Screens et avancement** :

- **Configuration VLAN**, pour les lignes de commandes des VLAN
- **Network** (VLAN, DHCP, RELAY, IP),
- **Services** (DNS, WEB, Backups)
- **Gestion** des utilisateurs
- **Configuration WEB**
- **Exploitation**, pour la sauvegarde sur le NFS de la base de donnée
- **Problèmes rencontrés**
- **Sources**, pour ce qui est des codes sources utilisés (pour le sql, html, etc...)

Dans **README** :

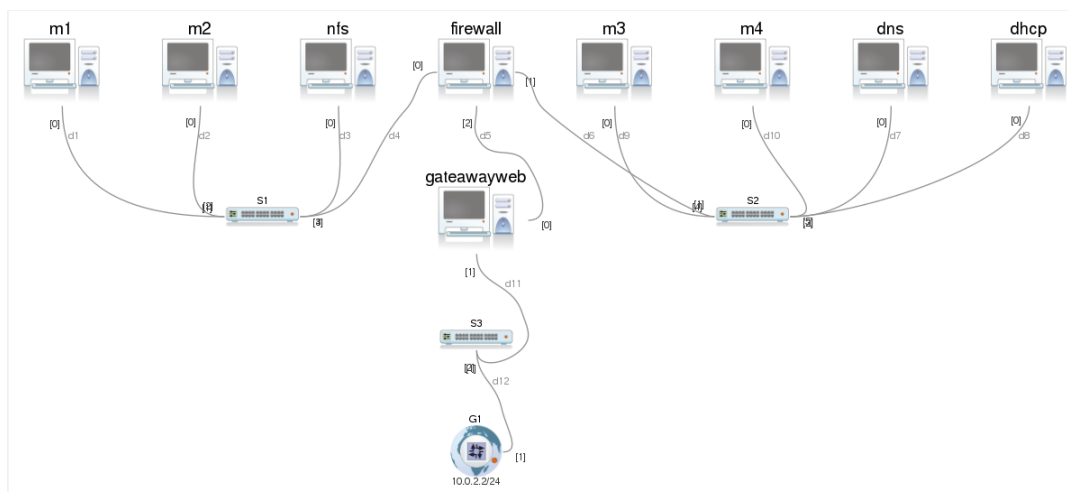
- **Booting order**, pour que tout le fonctionnement soit pérenne
- **Shut down order**

Mises en place d'un OneNote de suivis :

- Suivi de l'avancée
- Répartition des tâches
- Documentation des commandes et screenshots
- Documentation des problèmes rencontrés

b) Mise en place des équipements

Une fois le schéma logique mis en place, on a pu établir le modèle final avec les postes, les switches et la Gateway mise en place sur Marionnet.



Configuration du réseau

a) Configurations statiques du réseau

Pour commencer la configuration du réseau, on procède à l'attribution des IPs statiques avec leurs masques de sous-réseaux ainsi que les Gateways des appareils comme défini dans les spécifications techniques :

Machine	@IP	@Gateway	Port
m1 (eth0)	-	192.168.10.5/24	Port 1 (S1)
m2 (eth0)	-	192.168.10.5/24	Port 2 (S1)
nfs (eth0)	-	192.168.40.5/24	Port 3 (S1)
firewall (eth0.10)	192.168.10.5/24	-	Port 4 (S1)
firewall (eth0.20)	192.168.40.5/24	-	Port 4 (S1)
firewall (eth1)	192.168.20.5/24	-	Port 1 (S2)
firewall (eth2)	192.168.50.5/24	-	-
gatewayweb (eth0)	192.168.50.1/24	-	-
gatewayweb (eth1)	(DHCP via G1)10.0.2.2/24	-	-
dns (eth0)	-	192.168.20.5/24	-
dhcp (eth0)	-	192.168.20.5/24	-
m3 (eth0)	-	192.168.20.5/24	-
m4 (eth0)	-	192.168.20.5/24	-

Pour ça, il suffit de modifier les paramètres des interfaces directement dans les fichiers de configuration associés /etc/network/interfaces. Exemple dans la capture ci-dessous :

```

GNU nano 2.2.4      File: /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 192.168.100.5
    netmask 255.255.255.0

auto eth2
iface eth2 inet static
    address 192.168.20.5
    netmask 255.255.255.0

auto eth1
iface eth1 inet static
    address 192.168.50.5
    netmask 255.255.255.0
    gateway 192.168.50.1
  
```

Fichier de configuration des interfaces réseaux sur FIREWALL

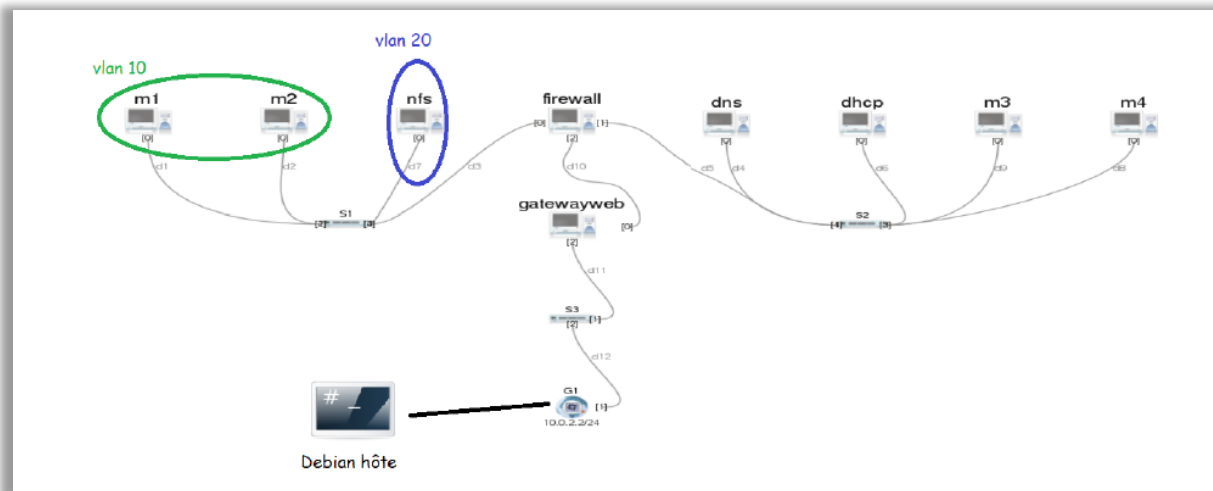
Une fois les configurations effectuées, on a procédé au forwarding des paquets sur le Firewall, en effet dans notre réseau, notre pare-feu doit agir comme un routeur et être capable de « forwarder » les paquets d'un réseau à l'autre. Pour ça, il suffit de modifier le fichier d'autorisation du forwarding des paquets : mettre la valeur à 1 dans le fichier /proc/sys/net/ipv4/ip_forward.

```

FIREWALL:~# cat /proc/sys/net/ipv4/ip_forward
1
FIREWALL:~#
  
```

b) Mise en place des VLANs

Rappel des VLAN à mettre en place dans notre réseau :



Il y a deux VLANs à mettre en place dans notre configuration pour cloisonner davantage le réseau et améliorer la sécurité.

Pour ça, il faut configurer notre switch S1 branché aux postes appartenant aux VLANs à créer et au pare-feu/routeur.

Par manque de possibilité d'automatiser cette mise en place au démarrage dû au système d'exploitation utilisé par le switch, cette étape est nécessaire à chaque redémarrage contrairement aux autres configurations ([voir explication rc.local](#)).

Dans cette étape, on crée les VLANs avec la commande **vlan/create**. On ajoute des « sous-ports » sur les interfaces réseaux du switch avec la commande **vlan/addport**. Enfin, la commande **port/setvlan** associe un appareil branché sur un port X au VLAN Y.

```
S1 terminal
VDE switch V.2.3.2
(C) Virtual Square Team (coord. R. Davoli) 2005,2006,2007 - GPLv2

vde$ vlan/create 10
1000 Success

vde$ vlan/addport 10 4
1000 Success

vde$ port/setvlan 1 10
1000 Success

vde$ port/setvlan 2 10
1000 Success

vde$ vlan/create 20
1000 Success

vde$ vlan/addport 20 4
1000 Success

vde$ port/setvlan 3 20
1000 Success
```

```
vde$ vlan/print
0000 DATA END WITH '.'
VLAN 0000
-- Port 0004 tagged=0 active=1 status=Forwarding
VLAN 0010
-- Port 0001 tagged=0 active=1 status=Forwarding
-- Port 0002 tagged=0 active=1 status=Forwarding
-- Port 0004 tagged=1 active=1 status=Forwarding
VLAN 0020
-- Port 0003 tagged=0 active=1 status=Forwarding
-- Port 0004 tagged=1 active=1 status=Forwarding
+
1000 Success
vde$
```

Affichage des VLANs mis en place

Problème rencontré : lors de la configuration des liens en TRUNK sur le firewall, la VM avait tendance à crasher.

Dans firewall, on doit ensuite créer et activer les interfaces TRUNK créées, pour ça on peut utiliser les commandes **vconfig** et **ifconfig**. Dans la capture suivante, on a mis en place ces commandes dans un fichier « /etc/rc.local » lancé automatiquement au démarrage de la machine ([voir section dédiée](#)) :

```
FIREWALL
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

vconfig add eth0 10
vconfig add eth0 20

ifconfig eth0.20 up
ifconfig eth0.20 192.168.40.5 netmask 255.255.255.0
ifconfig eth0.10 up
ifconfig eth0.10 192.168.10.5 netmask 255.255.255.0

service isc-dhcp-server start

exit 0
FIREWALL:~#
```

c) Mise en place du serveur DHCP et du DHCP Relay

Pour assigner dynamiquement des adresses IP aux appareils, on a mis en place un serveur DHCP.

Pour la configuration de celui-ci, on doit modifier le fichier « /etc/dhcp/dhcpd.conf » dans la machine correspondante.

On y crée des pools DHCP pour les réseaux demandés

Comme évoqué précédemment, le DHCP utilise le pare-feu en tant que routeur, ainsi on utilise l'ip de l'interface du pare-feu connecté au réseau où se trouve le serveur DHCP pour option routers. On ajoute les infos sur le serveur dns, un range d'ip à assigner et enfin on associe au serveur DNS une adresse fixe en utilisant l'adresse mac du poste.

```
# option definitions common to all supported networks...
option domain-name "st-potache.fr";
option domain-name-servers 192.168.20.15 ,10.0.2.3;

default-lease-time 600;
max-lease-time 7200;
```

```
subnet 192.168.40.0 netmask 255.255.255.0 {
    option routers 192.168.40.5;

    range 192.168.40.1 192.168.40.9;

    host nfs {
        hardware ethernet 02:04:06:81:3e:ae;
        fixed-address 192.168.40.10;
    }
}

subnet 192.168.10.0 netmask 255.255.255.0 {
    option routers 192.168.10.5;

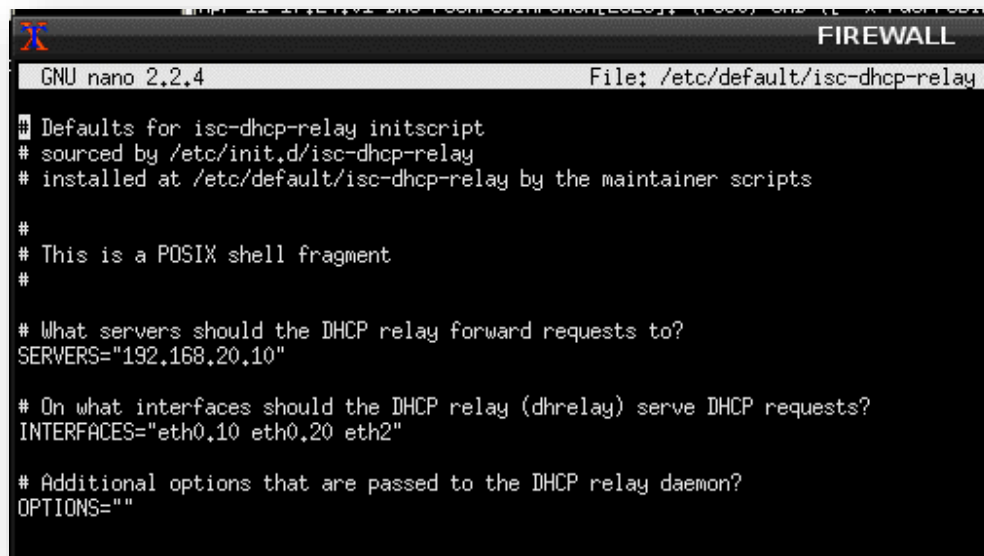
    range 192.168.10.1 192.168.10.10;
}

subnet 192.168.20.0 netmask 255.255.255.0 {
    option routers 192.168.20.5;

    range 192.168.20.1 192.168.20.15;

    host DNS {
        hardware ethernet 02:04:06:73:b5:31;
        fixed-address 192.168.20.15;
    }
}
```


Pour permettre le relay de paquets DHCP via le pare-feu (agit comme un routeur), on configure le fichier « /etc/default/isc-dhcp-relay » sur celui-ci :



```

FIREWALL
GNU nano 2.2.4 File: /etc/default/isc-dhcp-relay

# Defaults for isc-dhcp-relay initscript
# sourced by /etc/init.d/isc-dhcp-relay
# installed at /etc/default/isc-dhcp-relay by the maintainer scripts
#
# This is a POSIX shell fragment
#
# What servers should the DHCP relay forward requests to?
SERVERS="192.168.20.10"
#
# On what interfaces should the DHCP relay (dhrelay) serve DHCP requests?
INTERFACES="eth0,10 eth0,20 eth2"
#
# Additional options that are passed to the DHCP relay daemon?
OPTIONS=""
```

On a précédemment mis en place le **packet forwarding** via le fichier « /proc/sys/net/ipv4/ip_forward » mais pour maintenir cette configuration de façon pérenne, on peut modifier le fichier « /etc/sysctl.conf ».




```

FIREWALL
GNU nano 2.2.4 File: /etc/sysctl.conf

#####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

[ Read 60 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Une fois les configurations établies et le fichier « /etc/dhcp/dhcp.conf » complet, on peut configurer l'interface du DHCP de façon statique dans le fichier interfaces.



```
GNU nano 2.2.4  File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

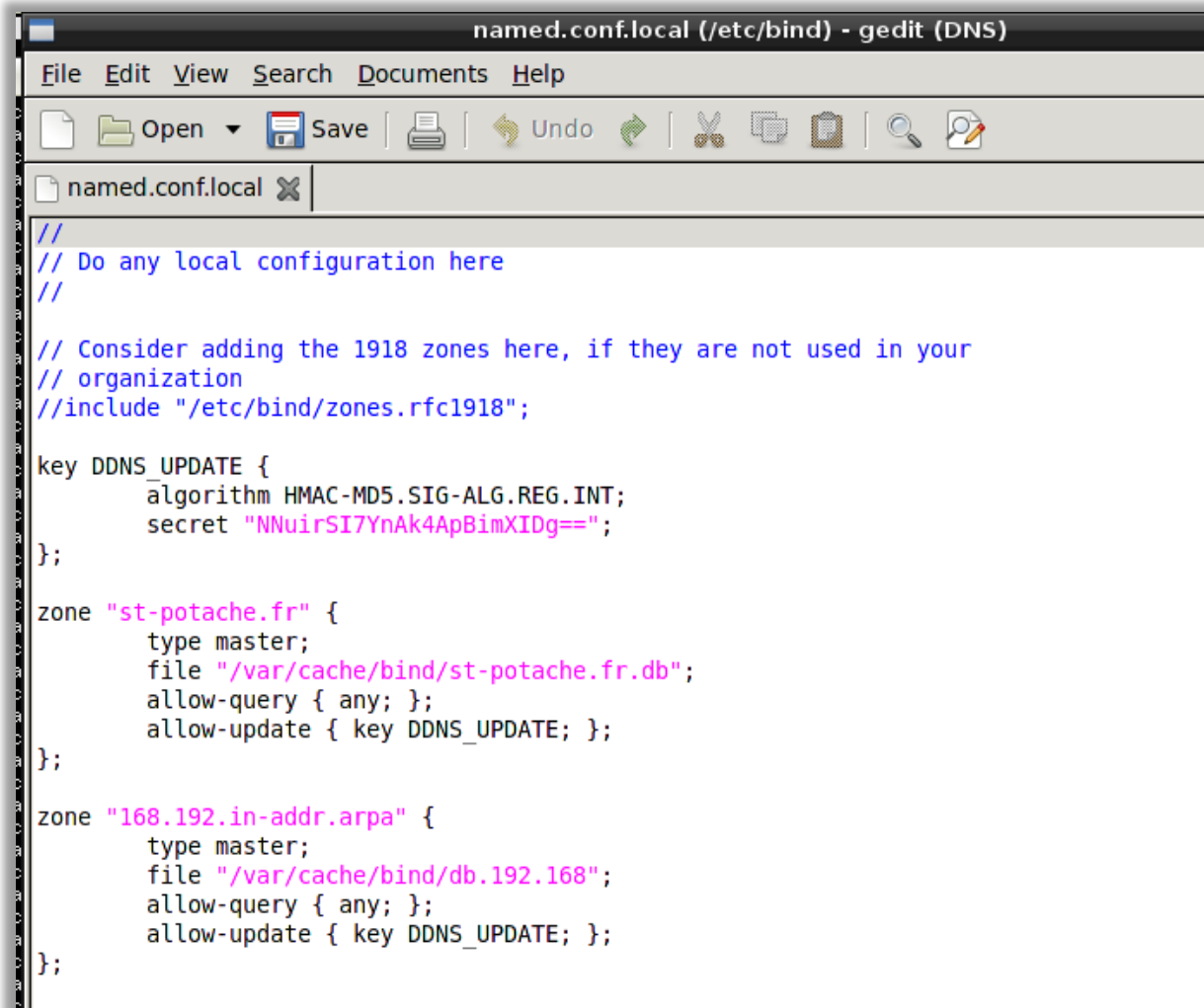
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 192.168.20.10
    netmask 255.255.255.0
    dns-nameservers 192.168.20.15
    dns-search st-potache.fr
    gateway 192.168.20.5
```

En dernier lieu, on peut lancer les différents services DHCP, isc-dhcp-server sur le serveur DHCP (service isc-dhcp-server start) et isc-dhcp-relay sur le pare-feu (service isc-dhcp-relay start).

d) Mise en place du serveur DNS

Pour mettre en place le DNS, on configure deux zones dans le fichier « /etc/bind/named.conf.local » :



```
//  
// Do any local configuration here  
//  
  
// Consider adding the 1918 zones here, if they are not used in your  
// organization  
//include "/etc/bind/zones.rfc1918";  
  
key DDNS_UPDATE {  
    algorithm HMAC-MD5.SIG-ALG.REG.INT;  
    secret "NNuIrSI7YnAk4ApBimXIDg==";  
};  
  
zone "st-potache.fr" {  
    type master;  
    file "/var/cache/bind/st-potache.fr.db";  
    allow-query { any; };  
    allow-update { key DDNS_UPDATE; };  
};  
  
zone "168.192.in-addr.arpa" {  
    type master;  
    file "/var/cache/bind/db.192.168";  
    allow-query { any; };  
    allow-update { key DDNS_UPDATE; };  
};
```

On utilise la commande :

```
« root# dnssec-keygen -a HMAC-MD5 -b 128 -r /dev/urandom -n USER  
DDNS_UPDATE »
```

Afin de générer une clé unique qui permettra au serveur DHCP de mettre à jour les entrées du serveur DNS de manière sécurisée.

On configure ensuite un fichier par zone, le premier : « /var/cache/bind/st-potache.fr.db »

```

$ORIGIN .
$TTL 86400      ; 1 day
st-potache.fr  IN SOA  st-potache.fr. root.st-potache.fr. (
                    2027      ; serial
                    604800    ; refresh (1 week)
                    86400     ; retry (1 day)
                    2419200   ; expire (4 weeks)
                    86400     ; minimum (1 day)
                    )
                    NS      dns.st-potache.fr.
                    A       192.168.20.15

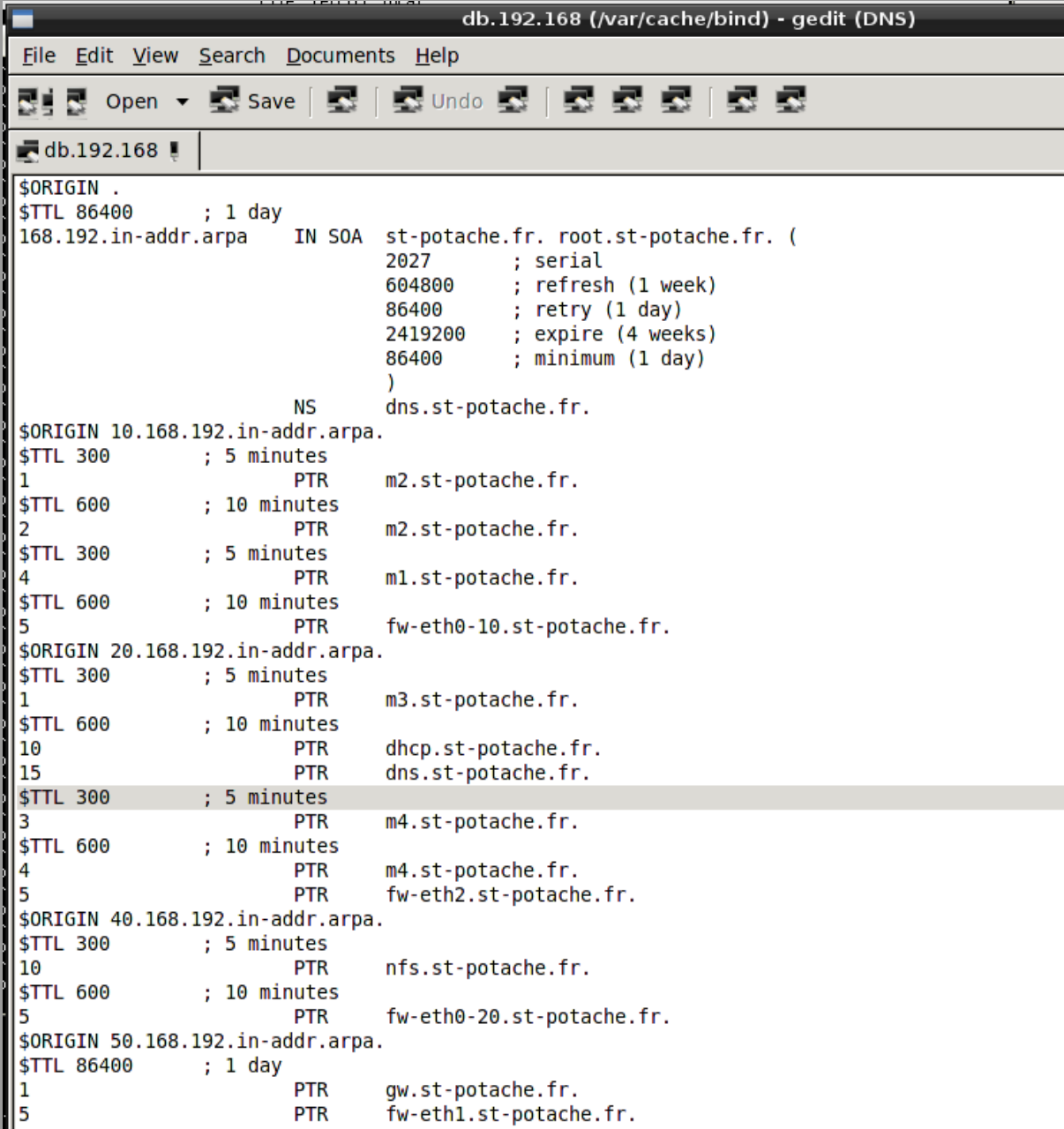
$ORIGIN st-potache.fr.
dhcp          A       192.168.20.10
dns           A       192.168.20.15
fw-eth0-10    A       192.168.10.5
fw-eth0-20    A       192.168.40.5
fw-eth1       A       192.168.50.5
fw-eth2       A       192.168.20.5
gw            A       192.168.50.1
$TTL 300      ; 5 minutes
m1            A       192.168.10.4
              TXT     "008d04f3111543fa3727659e1eede67e4d"
m2            A       192.168.10.1
              TXT     "0011637e3296f0d4a8b48b77e5e0ecb7cd"
m3            A       192.168.20.1
              TXT     "006864092d44e642879795ea64c85ef981"
m4            A       192.168.20.3
              TXT     "00a234a0a5a2277091dfbe9c90b1ce4325"
nfs           A       192.168.40.10
              TXT     "004281c88ce11ecabf4d2cbc1765faa917"
    
```

On peut voir que le premier fichier est le fichier de résolution de nom d'hôte qui renseigne des adresses IPs. Il est mis à jour par le serveur DHCP lorsque les baux sont mis à jour. Exemple de mise à jour lorsque M3 récupère une adresse IP :

```

Apr 11 17:18:43 DNS named[981]: client 192.168.20.10#36609: updating zone 'st-potache.fr/IN': deleting an RR at m3.st-potache.fr A
Apr 11 17:18:43 DNS named[981]: zone st-potache.fr/IN: sending notifies (serial 2032)
Apr 11 17:18:43 DNS named[981]: client 192.168.20.10#51162: signer "ddns_update" approved
Apr 11 17:18:43 DNS named[981]: client 192.168.20.10#51162: updating zone 'st-potache.fr/IN': deleting an RR at m3.st-potache.fr TXT
Apr 11 17:18:43 DNS named[981]: client 192.168.20.10#48070: signer "ddns_update" approved
Apr 11 17:18:43 DNS named[981]: client 192.168.20.10#48070: updating zone '168.192.in-addr.arpa/IN': deleting rrsset at '1.20.168.192.in-addr.arpa' PTR
Apr 11 17:18:43 DNS named[981]: zone 168.192.in-addr.arpa/IN: sending notifies (serial 2030)
Apr 11 17:18:46 DNS named[981]: client 192.168.20.10#37688: signer "ddns_update" approved
Apr 11 17:18:46 DNS named[981]: client 192.168.20.10#37688: updating zone 'st-potache.fr/IN': adding an RR at 'm3.st-potache.fr' A
Apr 11 17:18:46 DNS named[981]: client 192.168.20.10#37688: updating zone 'st-potache.fr/IN': adding an RR at 'm3.st-potache.fr' TXT
Apr 11 17:18:46 DNS named[981]: client 192.168.20.10#56380: signer "ddns_update" approved
Apr 11 17:18:46 DNS named[981]: client 192.168.20.10#56380: updating zone '168.192.in-addr.arpa/IN': deleting rrsset at '1.20.168.192.in-addr.arpa' PTR
Apr 11 17:18:46 DNS named[981]: client 192.168.20.10#56380: updating zone '168.192.in-addr.arpa/IN': adding an RR at '1.20.168.192.in-addr.arpa' PTR
Apr 11 17:18:48 DNS named[981]: zone st-potache.fr/IN: sending notifies (serial 2034)
Apr 11 17:18:48 DNS named[981]: zone 168.192.in-addr.arpa/IN: sending notifies (serial 2031)
    
```

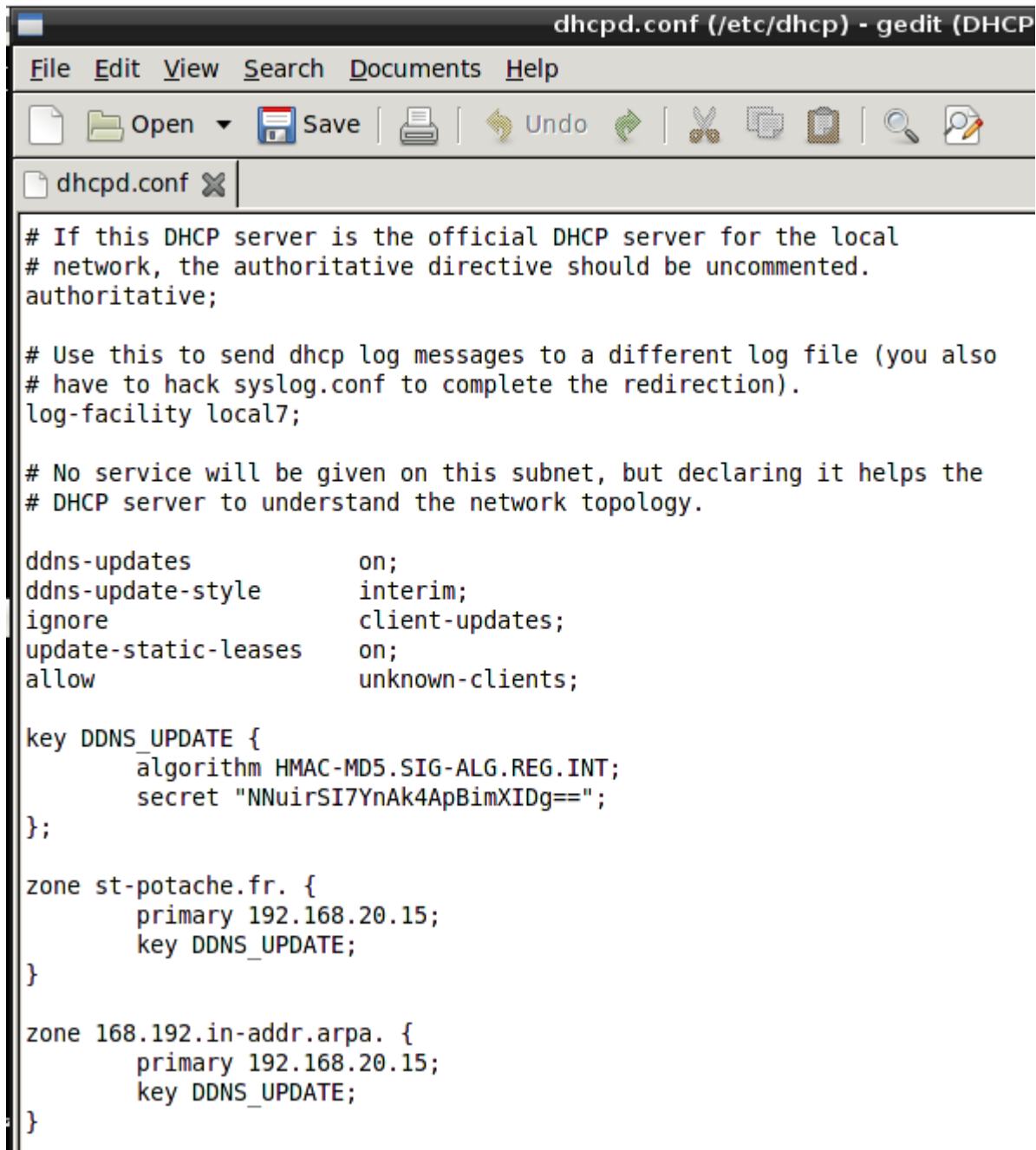
Le deuxième, « /var/cache/bind/db.192.168 » :



```
db.192.168 (/var/cache/bind) - gedit (DNS)
File Edit View Search Documents Help
Open Save Undo
db.192.168
$ORIGIN .
$TTL 86400      ; 1 day
168.192.in-addr.arpa  IN SOA  st-potache.fr. root.st-potache.fr. (
                        2027      ; serial
                        604800    ; refresh (1 week)
                        86400     ; retry (1 day)
                        2419200   ; expire (4 weeks)
                        86400     ; minimum (1 day)
                        )
                        NS      dns.st-potache.fr.
$ORIGIN 10.168.192.in-addr.arpa.
$TTL 300        ; 5 minutes
1               PTR      m2.st-potache.fr.
$TTL 600        ; 10 minutes
2               PTR      m2.st-potache.fr.
$TTL 300        ; 5 minutes
4               PTR      m1.st-potache.fr.
$TTL 600        ; 10 minutes
5               PTR      fw-eth0-10.st-potache.fr.
$ORIGIN 20.168.192.in-addr.arpa.
$TTL 300        ; 5 minutes
1               PTR      m3.st-potache.fr.
$TTL 600        ; 10 minutes
10              PTR      dhcp.st-potache.fr.
15              PTR      dns.st-potache.fr.
$TTL 300        ; 5 minutes
3               PTR      m4.st-potache.fr.
$TTL 600        ; 10 minutes
4               PTR      m4.st-potache.fr.
5               PTR      fw-eth2.st-potache.fr.
$ORIGIN 40.168.192.in-addr.arpa.
$TTL 300        ; 5 minutes
10              PTR      nfs.st-potache.fr.
$TTL 600        ; 10 minutes
5               PTR      fw-eth0-20.st-potache.fr.
$ORIGIN 50.168.192.in-addr.arpa.
$TTL 86400      ; 1 day
1               PTR      gw.st-potache.fr.
5               PTR      fw-eth1.st-potache.fr.
```

On peut voir que le premier fichier est le fichier de résolution d'adresses (REVERSE DNS) qui renseigne des noms d'hôtes. Il est mis à jour par le serveur DHCP lorsque les baux sont mis à jour.

Enfin, il faut mettre à jour le fichier de configuration du serveur DHCP : « /etc/dhcp/dhcpd.conf »
Il suffit de renseigner les zones à mettre à jour et la clé privée partagée pour permettre la mise à jour.



```
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
authoritative;

# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
log-facility local7;

# No service will be given on this subnet, but declaring it helps the
# DHCP server to understand the network topology.

ddns-updates                on;
ddns-update-style            interim;
ignore                       client-updates;
update-static-leases         on;
allow                        unknown-clients;

key DDNS_UPDATE {
    algorithm HMAC-MD5.SIG-ALG.REG.INT;
    secret "NNuirSI7YnAk4ApBimXIDg==";
};

zone st-potache.fr. {
    primary 192.168.20.15;
    key DDNS_UPDATE;
}

zone 168.192.in-addr.arpa. {
    primary 192.168.20.15;
    key DDNS_UPDATE;
}
```

e) Test du serveur DNS

```
m3:~# nslookup st-potache.fr
Server:      192.168.20.15
Address:     192.168.20.15#53

Name:   st-potache.fr
Address: 192.168.20.15

m3:~# nslookup dhcp
Server:      192.168.20.15
Address:     192.168.20.15#53

Name:   dhcp.st-potache.fr
Address: 192.168.20.10

m3:~# dig dns.st-potache.fr
; <<> DiG 9.7.3 <<> dns.st-potache.fr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39805
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;dns.st-potache.fr.      IN      A

;; ANSWER SECTION:
dns.st-potache.fr.      86400   IN      A      192.168.20.15

;; AUTHORITY SECTION:
st-potache.fr.          86400   IN      NS      dns.st-potache.fr.

;; Query time: 2 msec
;; SERVER: 192.168.20.15#53(192.168.20.15)
;; WHEN: Sun Apr 11 17:24:22 2021
;; MSG SIZE rcvd: 65

m3:~#
```

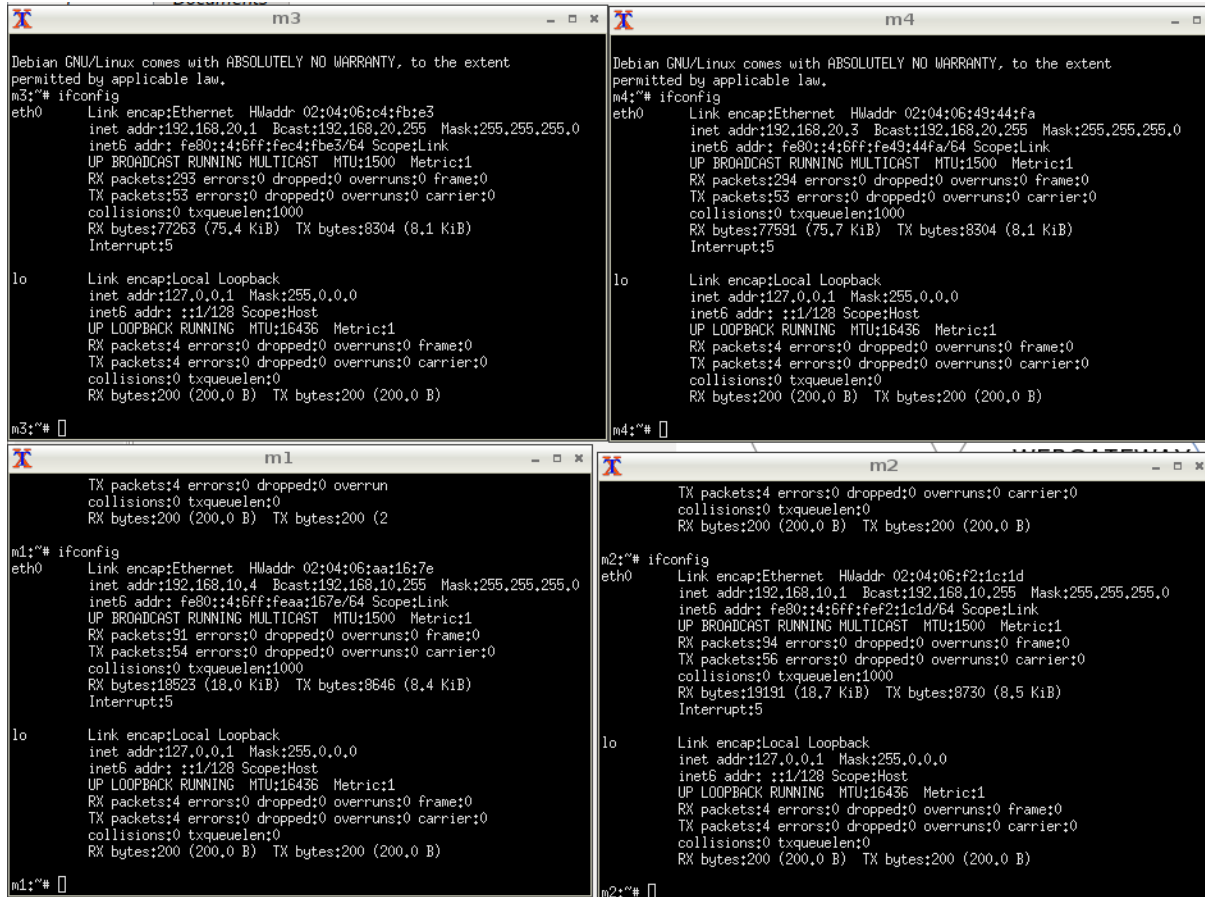
Depuis la machine M3, on peut voir que les commandes **DIG**, **NSLOOKUP** fonctionnent.
De plus, on peut utiliser le **PING** sur les machines en utilisant leurs noms d'hôtes :

```
m3:~# ping gw
PING gw.st-potache.fr (192.168.50.1) 56(84) bytes of data:
64 bytes from gw.st-potache.fr (192.168.50.1): icmp_req=1 ttl=63 time=0.934 ms
64 bytes from gw.st-potache.fr (192.168.50.1): icmp_req=2 ttl=63 time=4.26 ms
64 bytes from gw.st-potache.fr (192.168.50.1): icmp_req=3 ttl=63 time=2.97 ms
^C
--- gw.st-potache.fr ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2129ms
rtt min/avg/max/mdev = 0.934/2.724/4.264/1.371 ms
m3:~#
```

Tests de connectivité

2. Vérification de l'adressage

En utilisant la commande `ifconfig`, on peut vérifier pour chaque machine la configuration des interfaces réseaux.



The image displays four terminal windows, each showing the output of the `ifconfig` command for a specific machine (m3, m4, m1, m2). Each window shows the configuration for the `eth0` and `lo` interfaces. The `eth0` interface is configured with a static IP address, a subnet mask, and a broadcast address. The `lo` interface is configured with a loopback address. The output also includes statistics for each interface, such as RX and TX packets, errors, dropped frames, and collisions.

```
m3:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 02:04:06:c4:fb:e3
          inet addr:192.168.20.1  Bcast:192.168.20.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fe4:fbe3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:293 errors:0 dropped:0 overruns:0 frame:0
          TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:77263 (75.4 KiB)  TX bytes:8304 (8.1 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:200 (200.0 B)  TX bytes:200 (200.0 B)

m3:~#
```

```
m4:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 02:04:06:49:44:fa
          inet addr:192.168.20.3  Bcast:192.168.20.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fe49:44fa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:294 errors:0 dropped:0 overruns:0 frame:0
          TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:77581 (75.7 KiB)  TX bytes:8304 (8.1 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:200 (200.0 B)  TX bytes:200 (200.0 B)

m4:~#
```

```
m1:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 02:04:06:aa:16:7e
          inet addr:192.168.10.4  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:feaa:167e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:54 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18523 (18.0 KiB)  TX bytes:8646 (8.4 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:200 (200.0 B)  TX bytes:200 (200.0 B)

m1:~#
```

```
m2:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 02:04:06:f2:1c:1d
          inet addr:192.168.10.1  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fe2:1c1d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:94 errors:0 dropped:0 overruns:0 frame:0
          TX packets:56 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:19191 (18.7 KiB)  TX bytes:8730 (8.5 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:200 (200.0 B)  TX bytes:200 (200.0 B)

m2:~#
```

Configuration IP sur m1, 2, 3, 4

The image displays four terminal windows, each showing the output of the 'ifconfig' command for different network interfaces. Each window also includes a disclaimer: 'Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.'

- DNS:** Shows configuration for 'eth0' (Ethernet) and 'lo' (Local Loopback). 'eth0' has IP 192.168.20.15 and MAC 02:04:06:73:b5:31. 'lo' has IP 127.0.0.1.
- DHCP:** Shows configuration for 'eth0' and 'lo'. 'eth0' has IP 192.168.20.10 and MAC 02:04:06:e1:27:0e. 'lo' has IP 127.0.0.1.
- WEBGATEWAY:** Shows configuration for 'eth0' and 'eth1'. 'eth0' has IP 192.168.50.1 and MAC 02:04:06:9e:5c:a8. 'eth1' has IP 10.0.2.15 and MAC 02:04:06:93:85:e6.
- NFS:** Shows configuration for 'eth0' and 'lo'. 'eth0' has IP 192.168.40.10 and MAC 02:04:06:81:3e:ae. 'lo' has IP 127.0.0.1.

Configuration IP sur DNS, DHCP, WEBGATEWAY et NFS

```
FIREWALL:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 02:04:06:d3:ed:55
          inet addr:192.168.100.5  Bcast:192.168.100.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fed3:ed55/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:170 errors:0 dropped:0 overruns:0 frame:0
          TX packets:230 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25424 (24.8 KiB)  TX bytes:48450 (47.3 KiB)
          Interrupt:5

eth1      Link encap:Ethernet  HWaddr 02:04:06:71:84:c0
          inet addr:192.168.50.5  Bcast:192.168.50.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fe71:84c0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:18 errors:0 dropped:0 overruns:0 frame:0
          TX packets:36 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1782 (1.7 KiB)  TX bytes:2040 (1.9 KiB)
          Interrupt:5

eth2      Link encap:Ethernet  HWaddr 02:04:06:59:33:93
          inet addr:192.168.20.5  Bcast:192.168.20.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fe59:3393/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:466 errors:0 dropped:0 overruns:0 frame:0
          TX packets:488 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:100562 (98.2 KiB)  TX bytes:124366 (121.4 KiB)
          Interrupt:5

eth0.10   Link encap:Ethernet  HWaddr 02:04:06:d3:ed:55
          inet addr:192.168.10.5  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fed3:ed55/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:114 errors:0 dropped:0 overruns:0 frame:0
          TX packets:149 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:16548 (16.1 KiB)  TX bytes:31376 (30.6 KiB)

eth0.20   Link encap:Ethernet  HWaddr 02:04:06:d3:ed:55
          inet addr:192.168.40.5  Bcast:192.168.40.255  Mask:255.255.255.0
          inet6 addr: fe80::4:6ff:fed3:ed55/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:56 errors:0 dropped:0 overruns:0 frame:0
          TX packets:75 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:8196 (8.0 KiB)  TX bytes:15710 (15.3 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:14 errors:0 dropped:0 overruns:0 frame:0
```

Configuration IP sur FIREWALL

f) Connectivité interne

Grâce à la mise en place du DDNS, on peut pinger les machines entre elles en utilisant leurs noms d'hôte sans risque de problèmes.

Procédons à quelques tests de ping :

- Ping de m1 à m2, 3, 4 :

```
m1
m1:~# ping m2
PING m2.st-potache.fr (192.168.10.1) 56(84) bytes of data.
64 bytes from m2.st-potache.fr (192.168.10.1): icmp_req=1 ttl=64 time=12.6 ms
64 bytes from m2.st-potache.fr (192.168.10.1): icmp_req=2 ttl=64 time=2.76 ms
^C
--- m2.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1003ms
rtt min/avg/max/mdev = 2.763/7.725/12.688/4.963 ms
m1:~# ping m3
PING m3.st-potache.fr (192.168.20.1) 56(84) bytes of data.
64 bytes from m3.st-potache.fr (192.168.20.1): icmp_req=1 ttl=63 time=1.66 ms
64 bytes from m3.st-potache.fr (192.168.20.1): icmp_req=2 ttl=63 time=2.46 ms
^C
--- m3.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.660/2.062/2.465/0.405 ms
m1:~# ping m4
PING m4.st-potache.fr (192.168.20.3) 56(84) bytes of data.
64 bytes from m4.st-potache.fr (192.168.20.3): icmp_req=1 ttl=63 time=12.9 ms
^C
--- m4.st-potache.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 12.906/12.906/12.906/0.000 ms
m1:~#
```

- Ping de m2 aux interfaces du firewall :

```
m2
m2:~# ping fw-eth0-10
PING fw-eth0-10.st-potache.fr (192.168.10.5) 56(84) bytes of data.
64 bytes from fw-eth0-10.st-potache.fr (192.168.10.5): icmp_req=1 ttl=64 time=0.458 ms
64 bytes from fw-eth0-10.st-potache.fr (192.168.10.5): icmp_req=2 ttl=64 time=1.97 ms
^C
--- fw-eth0-10.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1006ms
rtt min/avg/max/mdev = 0.458/1.215/1.973/0.758 ms
m2:~# ping fw-eth0-20
PING fw-eth0-20.st-potache.fr (192.168.40.5) 56(84) bytes of data.
64 bytes from fw-eth0-20.st-potache.fr (192.168.40.5): icmp_req=1 ttl=64 time=0.307 ms
^C
--- fw-eth0-20.st-potache.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.307/0.307/0.307/0.000 ms
m2:~# ping fw-eth1
PING fw-eth1.st-potache.fr (192.168.50.5) 56(84) bytes of data.
64 bytes from fw-eth1.st-potache.fr (192.168.50.5): icmp_req=1 ttl=64 time=0.365 ms
64 bytes from fw-eth1.st-potache.fr (192.168.50.5): icmp_req=2 ttl=64 time=2.07 ms
^C
--- fw-eth1.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1014ms
rtt min/avg/max/mdev = 0.365/1.221/2.078/0.857 ms
m2:~# ping fw-eth2
PING fw-eth2.st-potache.fr (192.168.20.5) 56(84) bytes of data.
64 bytes from fw-eth2.st-potache.fr (192.168.20.5): icmp_req=1 ttl=64 time=0.428 ms
64 bytes from fw-eth2.st-potache.fr (192.168.20.5): icmp_req=2 ttl=64 time=1.11 ms
^C
--- fw-eth2.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.428/0.771/1.115/0.344 ms
m2:~#
```

- Ping de DNS à NFS, DHCP et WebGateway :

```
DNS
DNS:~# ping nfs
PING nfs.st-potache.fr (192.168.40.10) 56(84) bytes of data.
64 bytes from nfs.st-potache.fr (192.168.40.10): icmp_req=1 ttl=63 time=3.05 ms
64 bytes from nfs.st-potache.fr (192.168.40.10): icmp_req=2 ttl=63 time=2.63 ms
^C
--- nfs.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1010ms
rtt min/avg/max/mdev = 2.639/2.846/3.054/0.214 ms
DNS:~# ping dhcp
PING dhcp.st-potache.fr (192.168.20.10) 56(84) bytes of data.
64 bytes from dhcp.st-potache.fr (192.168.20.10): icmp_req=1 ttl=64 time=2.23 ms
64 bytes from dhcp.st-potache.fr (192.168.20.10): icmp_req=2 ttl=64 time=3.07 ms
^C
--- dhcp.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1008ms
rtt min/avg/max/mdev = 2.233/2.655/3.077/0.422 ms
DNS:~# ping gw
PING gw.st-potache.fr (192.168.50.1) 56(84) bytes of data.
64 bytes from gw.st-potache.fr (192.168.50.1): icmp_req=1 ttl=63 time=1.93 ms
64 bytes from gw.st-potache.fr (192.168.50.1): icmp_req=2 ttl=63 time=3.06 ms
^C
--- gw.st-potache.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.932/2.496/3.060/0.564 ms
DNS:~#
```

g) Connectivité vers l'extérieur

- Links de la WebGateway vers google.com :

```
WEBGATEWAY
Recherche  images Maps Play YouTube Actualites Gmail Drive Plus >>
Historique Web | Parametres | Connexion
Google

[ Recherche Google ] [ J'ai de la chance ] Recherche avancee

Google disponible en : English
Solutions publicitaires Solutions d'entreprise A propos de Google
Google.fr

(c) 2021 - Confidentialite - Conditions

http://www.google.fr/imghp?hl=fr&tab=wi
```

- Links du m1 vers le serveur web de la machine hôte ([voir configuration du serveur WEB](#)):



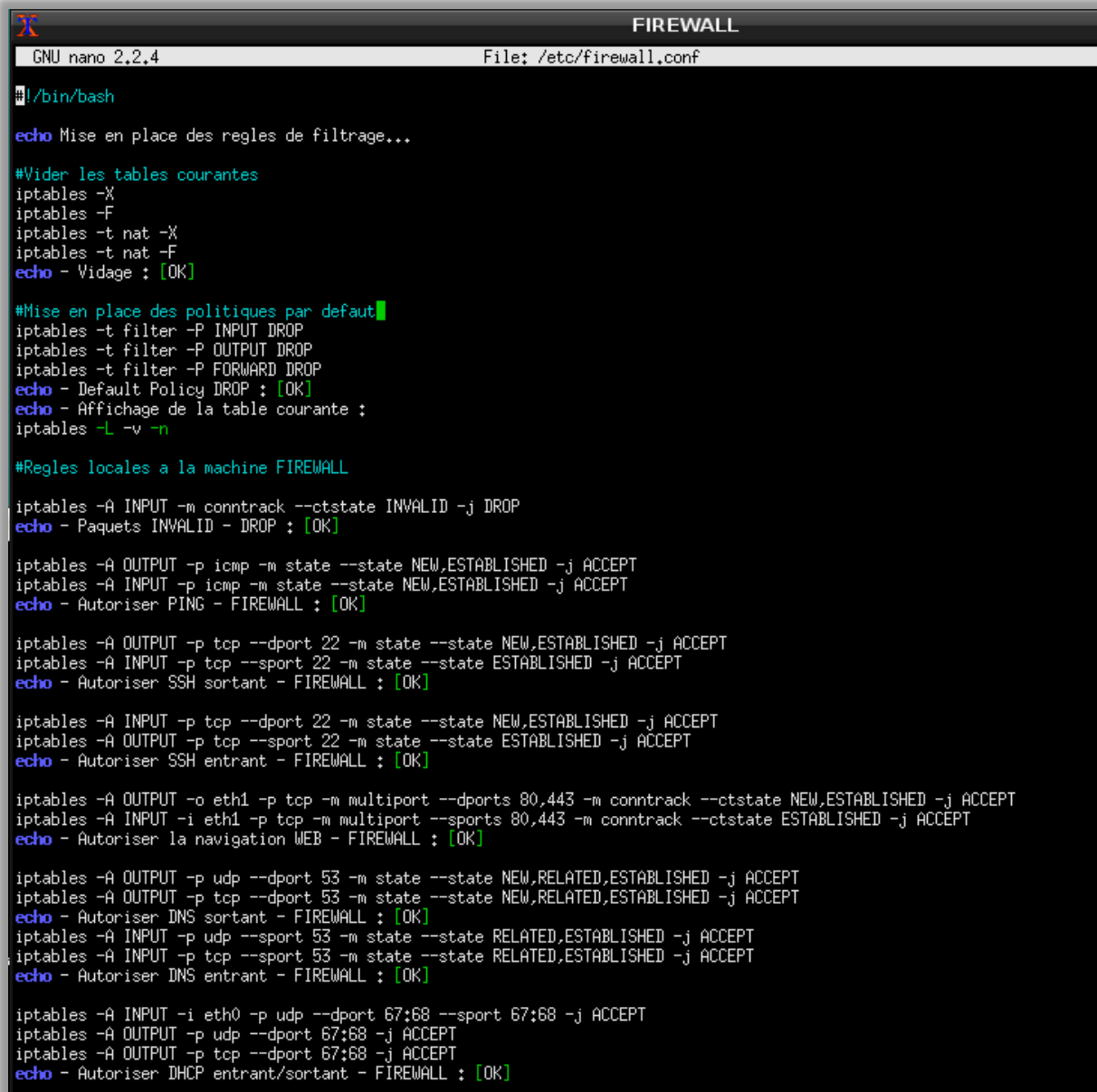
- Links de m3 vers google.com :



Configuration du Pare-feu

a) Configuration de base de la machine FIREWALL

On met en place un script BASH de configuration IPTABLES qui sera invoqué à chaque démarrage par « /etc/rc.local ».



```
GNU nano 2.2.4 File: /etc/firewall.conf

#!/bin/bash

echo Mise en place des regles de filtrage...

#Vider les tables courantes
iptables -X
iptables -F
iptables -t nat -X
iptables -t nat -F
echo - Vidage : [OK]

#Mise en place des politiques par default
iptables -t filter -P INPUT DROP
iptables -t filter -P OUTPUT DROP
iptables -t filter -P FORWARD DROP
echo - Default Policy DROP : [OK]
echo - Affichage de la table courante :
iptables -L -v -n

#Regles locales a la machine FIREWALL

iptables -A INPUT -m conntrack --ctstate INVALID -j DROP
echo - Paquets INVALID - DROP : [OK]

iptables -A OUTPUT -p icmp -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -p icmp -m state --state NEW,ESTABLISHED -j ACCEPT
echo - Autoriser PING - FIREWALL : [OK]

iptables -A OUTPUT -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
echo - Autoriser SSH sortant - FIREWALL : [OK]

iptables -A INPUT -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
echo - Autoriser SSH entrant - FIREWALL : [OK]

iptables -A OUTPUT -o eth1 -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -i eth1 -p tcp -m multiport --sports 80,443 -m conntrack --ctstate ESTABLISHED -j ACCEPT
echo - Autoriser la navigation WEB - FIREWALL : [OK]

iptables -A OUTPUT -p udp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser DNS sortant - FIREWALL : [OK]
iptables -A INPUT -p udp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -p tcp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser DNS entrant - FIREWALL : [OK]

iptables -A INPUT -i eth0 -p udp --dport 67:68 --sport 67:68 -j ACCEPT
iptables -A OUTPUT -p udp --dport 67:68 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 67:68 -j ACCEPT
echo - Autoriser DHCP entrant/sortant - FIREWALL : [OK]
```

```
#Communication au sein du réseau local
echo - Autoriser DHCP entre les reseaux - FIREWALL est DHCP-relay : [OK]
iptables -A FORWARD -p icmp --icmp-type any -j ACCEPT
echo - Autoriser PING entre les reseaux : [OK]

iptables -A FORWARD -p udp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -p udp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -p tcp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser DNS entre les reseaux : [OK]

#Protocole SSH a destination du port 22
iptables -A FORWARD -i eth0,20 -o eth2 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth0,20 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -i eth0,10 -o eth2 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth0,10 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -d 192.168.50,1 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

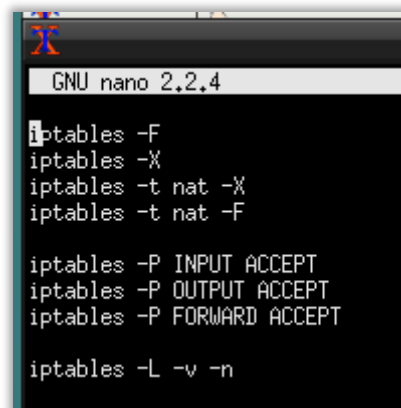
#Protocole SSH venant du port 22
iptables -A FORWARD -i eth0,20 -o eth2 -p tcp --sport 22 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth0,20 -p tcp --sport 22 -j ACCEPT
iptables -A FORWARD -i eth0,10 -o eth2 -p tcp --sport 22 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth0,10 -p tcp --sport 22 -j ACCEPT
iptables -A FORWARD -s 192.168.50,1 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
echo - Autoriser SSH au sein des reseaux locaux : [OK]

iptables -A FORWARD -i eth0,10 -o eth1 -p tcp -m multiport --dports 80,443 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0,10 -p tcp -m multiport --sports 80,443 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth0,20 -o eth1 -p tcp -m multiport --dports 80,443 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0,20 -p tcp -m multiport --sports 80,443 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth2 -o eth1 -p tcp -m multiport --dports 80,443 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth1 -o eth2 -p tcp -m multiport --sports 80,443 -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Routage du trafic WEB vers WGW : [OK]

iptables -A OUTPUT -s 192.168.40,10 -j ACCEPT
iptables -A INPUT -s 192.168.40,10 -j ACCEPT
iptables -A OUTPUT -d 192.168.40,10 -j ACCEPT
iptables -A INPUT -d 192.168.40,10 -j ACCEPT
iptables -A FORWARD -s 192.168.40,10 -j ACCEPT
iptables -A FORWARD -s 192.168.40,10 -j ACCEPT
echo - Regles souples pour le fonctionnement du serveur NFS : [OK]

echo - Affichage de la table courante :
iptables -L -v -n
```

Pour les tests, un autre script de remise à zéro (FLUSH) a été créé :



```
GNU nano 2.2.4

iptables -F
iptables -X
iptables -t nat -X
iptables -t nat -F

iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT

iptables -L -v -n
```

Démonstration des deux scripts :

```

- Paquets INVALID - DROP : [OK]
- Autoriser PING - FIREWALL : [OK]
- Autoriser SSH sortant - FIREWALL : [OK]
- Autoriser SSH entrant - FIREWALL : [OK]
- Autoriser la navigation WEB - FIREWALL : [OK]
- Autoriser DNS sortant - FIREWALL : [OK]
- Autoriser DNS entrant - FIREWALL : [OK]
- Autoriser DHCP entrant/sortant - FIREWALL : [OK]
- Autoriser DHCP entre les reseaux - FIREWALL est DHCP-relay : [OK]
- Autoriser PING entre les reseaux : [OK]
- Autoriser DNS entre les reseaux : [OK]
- Autoriser SSH au sein des reseaux locaux : [OK]
- Routage du traffic WEB vers MGM : [OK]
- Regles souples pour le fonctionnement du serveur NFS : [OK]
- Affichage de la table courante :
Chain INPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination ctstate
0 0 DROP all -- * * 0,0,0,0/0 0,0,0,0/0 INVALID
0 0 ACCEPT icmp -- * * 0,0,0,0/0 0,0,0,0/0 state NEW,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp spt:22 state ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp dpt:22 state NEW,ESTABLISHED
0 0 ACCEPT tcp -- eth1 * 0,0,0,0/0 0,0,0,0/0 multiport sports 80,443 ctstate ESTABLISHED
0 0 ACCEPT udp -- * * 0,0,0,0/0 0,0,0,0/0 udp spt:53 state RELATED,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp spt:53 state RELATED,ESTABLISHED
0 0 ACCEPT udp -- eth0 * 0,0,0,0/0 0,0,0,0/0 udp spts:67:68 dpts:67:68
0 0 ACCEPT all -- * * 192.168,40,10 0,0,0,0/0
0 0 ACCEPT all -- * * 0,0,0,0/0 192.168,40,10

Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 ACCEPT icmp -- * * 0,0,0,0/0 0,0,0,0/0 icmp type 255
0 0 ACCEPT udp -- * * 0,0,0,0/0 0,0,0,0/0 udp dpt:53 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp dpt:53 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT udp -- * * 0,0,0,0/0 0,0,0,0/0 udp spt:53 state RELATED,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp spt:53 state RELATED,ESTABLISHED
0 0 ACCEPT tcp -- eth0,20 eth2 0,0,0,0/0 0,0,0,0/0 tcp dpt:22
0 0 ACCEPT tcp -- eth2 eth0,20 0,0,0,0/0 0,0,0,0/0 tcp dpt:22
0 0 ACCEPT tcp -- eth0,10 eth2 0,0,0,0/0 0,0,0,0/0 tcp dpt:22
0 0 ACCEPT tcp -- eth2 eth0,10 0,0,0,0/0 0,0,0,0/0 tcp dpt:22
0 0 ACCEPT tcp -- * * 0,0,0,0/0 192.168,50,1 tcp dpt:22 state NEW,ESTABLISHED
0 0 ACCEPT tcp -- eth0,20 eth2 0,0,0,0/0 0,0,0,0/0 tcp spt:22
0 0 ACCEPT tcp -- eth2 eth0,20 0,0,0,0/0 0,0,0,0/0 tcp spt:22
0 0 ACCEPT tcp -- eth0,10 eth2 0,0,0,0/0 0,0,0,0/0 tcp spt:22
0 0 ACCEPT tcp -- eth2 eth0,10 0,0,0,0/0 0,0,0,0/0 tcp spt:22
0 0 ACCEPT tcp -- * * 192.168,50,1 0,0,0,0/0 tcp spt:22 state ESTABLISHED
0 0 ACCEPT tcp -- eth0,10 eth1 0,0,0,0/0 0,0,0,0/0 multiport dports 80,443 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT tcp -- eth1 eth0,10 0,0,0,0/0 0,0,0,0/0 multiport sports 80,443 state RELATED,ESTABLISHED
0 0 ACCEPT tcp -- eth0,20 eth1 0,0,0,0/0 0,0,0,0/0 multiport dports 80,443 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT tcp -- eth1 eth0,20 0,0,0,0/0 0,0,0,0/0 multiport sports 80,443 state RELATED,ESTABLISHED
0 0 ACCEPT tcp -- eth2 eth1 0,0,0,0/0 0,0,0,0/0 multiport dports 80,443 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT tcp -- eth1 eth2 0,0,0,0/0 0,0,0,0/0 multiport sports 80,443 state RELATED,ESTABLISHED
0 0 ACCEPT all -- * * 192.168,40,10 0,0,0,0/0
0 0 ACCEPT all -- * * 192.168,40,10 0,0,0,0/0

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 ACCEPT icmp -- * * 0,0,0,0/0 0,0,0,0/0 state NEW,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp dpt:22 state NEW,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp spt:22 state ESTABLISHED
0 0 ACCEPT tcp -- * eth1 0,0,0,0/0 0,0,0,0/0 multiport dports 80,443 ctstate NEW,ESTABLISHED
0 0 ACCEPT udp -- * * 0,0,0,0/0 0,0,0,0/0 udp dpt:53 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp dpt:53 state NEW,RELATED,ESTABLISHED
0 0 ACCEPT udp -- * * 0,0,0,0/0 0,0,0,0/0 udp dpts:67:68
0 0 ACCEPT tcp -- * * 0,0,0,0/0 0,0,0,0/0 tcp dpts:67:68
0 0 ACCEPT all -- * * 192.168,40,10 0,0,0,0/0
0 0 ACCEPT all -- * * 0,0,0,0/0 192.168,40,10
FIREWALL:~#

```

```

FIREWALL:~# /bin/bash /etc/fwflush.conf
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
FIREWALL:~#

```


Le firewall permet, comme le demande le cahier des charges :

- Supprimer tous les paquets INVALID (DROP)
- Autoriser VERS et DEPUIS (chaîne INPUT/OUTPUT) la machine FIREWALL :
 - Les connections SSH
 - Les requêtes PING
 - Les requêtes DNS
 - La navigation WEB
 - Les requêtes DHCP
- Autoriser entre les réseaux locaux (chaîne FORWARD sur les interfaces eth0.10 eth0.20 et eth2) :
 - Les connections SSH
 - Les requêtes PING
 - Les requêtes DNS
 - La navigation WEB
 - Les requêtes DHCP
- Autoriser les requêtes vers la machine NFS (192.168.40.10) pour le bon fonctionnement du NFS.
- Bloquer tout le trafic qui ne correspond pas aux règles précédentes.

b) Durcissement du filtrage au sein du réseau

Afin de renforcer le filtrage au sein du réseau, la machine WEBGATEWAY agit comme un pare-feu qui bloque tout le trafic qui n'est pas relatif aux PINGS, aux requêtes DNS ou au trafic WEB.

De plus, le SSH n'est possible que DEPUIS les réseaux locaux (interface eth1 de FIREWALL) et pas dans l'autre sens. Cela empêche quiconque d'utiliser la passerelle web comme moyen de se connecter en SSH au réseau.

```
WEBGATEWAY
GNU nano 2.2.4 File: /etc/firewall.conf
#!/bin/bash

echo Mise en place des regles de filtrage...

#Vider les tables courantes
iptables -X
iptables -F
iptables -t nat -X
iptables -t nat -F
echo - Vidage : [OK]

#Mise en place des politiques par default
iptables -t filter -P INPUT DROP
iptables -t filter -P OUTPUT DROP
iptables -t filter -P FORWARD DROP
echo - Default Policy DROP : [OK]
echo - Affichage de la table courante :
iptables -L -v -n

#Regles locales a la machine WEBGATEWAY

iptables -A INPUT -m conntrack --ctstate INVALID -j DROP
echo - Paquets INVALID - DROP : [OK]

iptables -A OUTPUT -p icmp -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -p icmp -m state --state NEW,ESTABLISHED -j ACCEPT
echo - Autoriser PING - WEBGATEWAY : [OK]

#iptables -A OUTPUT -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
#iptables -A INPUT -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
#echo - Autoriser SSH sortant - WEBGATEWAY : [OK]

iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
echo - Autoriser SSH entrant - WEBGATEWAY : [OK]

iptables -A OUTPUT -o eth1 -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -i eth1 -p tcp -m multiport --sports 80,443 -m conntrack --ctstate ESTABLISHED -j ACCEPT
echo - Autoriser la navigation WEB - GW1 : [OK]
```

```
iptables -A OUTPUT -p udp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -p tcp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser DNS sortant - WEBGATEWAY : [OK]
iptables -A INPUT -p udp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -p tcp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser DNS entrant - WEBGATEWAY : [OK]

iptables -A INPUT -i eth0 -p udp --dport 67:68 --sport 67:68 -j ACCEPT
#iptables -A OUTPUT -p udp --dport 67:68 -j ACCEPT
#iptables -A OUTPUT -p tcp --dport 67:68 -j ACCEPT
echo - Autoriser DHCP entrant - WEBGATEWAY : [OK]

#Communication au sein du réseau local

iptables -A FORWARD -p icmp --icmp-type any -j ACCEPT
echo - Autoriser PING traversant WEBGATEWAY : [OK]

iptables -A FORWARD -p udp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -p udp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -p tcp --sport 53 -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser DNS traversant WEBGATEWAY : [OK]

iptables -A FORWARD -i eth0 -o eth1 -p tcp -m multiport --dports 80,443 -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp -m multiport --sports 80,443 -m state --state RELATED,ESTABLISHED -j ACCEPT
echo - Autoriser trafic WEB traversant WEBGATEWAY : [OK]

echo - Affichage de la table courante :
iptables -L -v -n
```

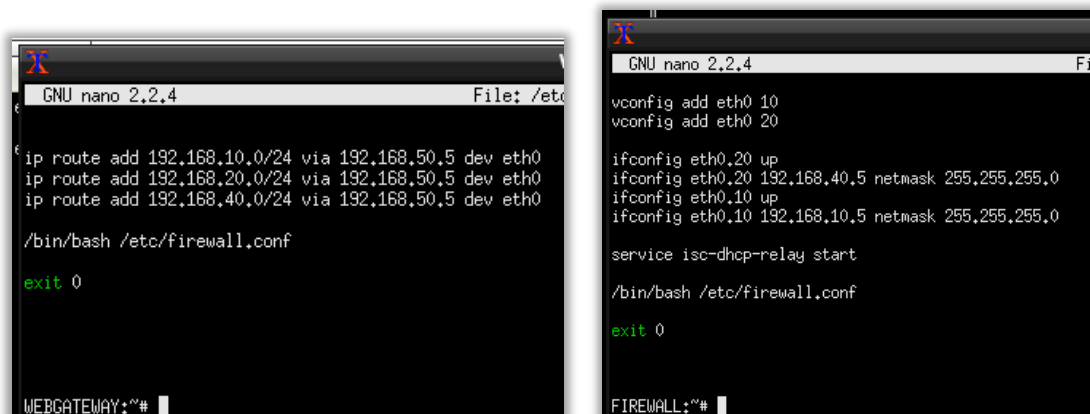
Démonstration du script IPTABLES de la machine WEBGATEWAY :

```

WEBGATEWAY:~# /bin/bash /etc/firewall.conf
Mise en place des regles de filtrage...
- Vidage : [OK]
- Default Policy DROP : [OK]
- Affichage de la table courante :
Chain INPUT (policy DROP 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
Chain FORWARD (policy DROP 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
Chain OUTPUT (policy DROP 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
- Paquets INVALID - DROP : [OK]
- Autoriser PING - WEBGATEWAY : [OK]
- Autoriser SSH entrant - WEBGATEWAY : [OK]
- Autoriser la navigation WEB - GW1 : [OK]
- Autoriser DNS sortant - WEBGATEWAY : [OK]
- Autoriser DNS entrant - WEBGATEWAY : [OK]
- Autoriser DHCP entrant - WEBGATEWAY : [OK]
- Autoriser PING traversant WEBGATEWAY : [OK]
- Autoriser DNS traversant WEBGATEWAY : [OK]
- Autoriser trafic WEB traversant WEBGATEWAY : [OK]
- Affichage de la table courante :
Chain INPUT (policy DROP 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
  0      0 DROP      all  --  *      *      0.0.0.0/0           0.0.0.0/0           ctstate INVALID
  0      0 ACCEPT    icmp --  *      *      0.0.0.0/0           0.0.0.0/0           state NEW,ESTABLISHED
  0      0 ACCEPT    tcp  --  eth0   *      0.0.0.0/0           0.0.0.0/0           tcp dpt:22 state NEW,ESTABLISHED
  0      0 ACCEPT    tcp  --  eth1   *      0.0.0.0/0           0.0.0.0/0           multiport sports 80,443 ctstate ESTABLISHED
  0      0 ACCEPT    udp  --  *      *      0.0.0.0/0           0.0.0.0/0           udp spt:53 state RELATED,ESTABLISHED
  0      0 ACCEPT    tcp  --  *      *      0.0.0.0/0           0.0.0.0/0           tcp spt:53 state RELATED,ESTABLISHED
  0      0 ACCEPT    udp  --  *      *      0.0.0.0/0           0.0.0.0/0           udp spts:67:68 dpts:67:68
Chain FORWARD (policy DROP 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
  0      0 ACCEPT    icmp --  *      *      0.0.0.0/0           0.0.0.0/0           icmp type 255
  0      0 ACCEPT    udp  --  *      *      0.0.0.0/0           0.0.0.0/0           udp dpt:53 state NEW,RELATED,ESTABLISHED
  0      0 ACCEPT    tcp  --  *      *      0.0.0.0/0           0.0.0.0/0           tcp dpt:53 state NEW,RELATED,ESTABLISHED
  0      0 ACCEPT    udp  --  *      *      0.0.0.0/0           0.0.0.0/0           udp spt:53 state RELATED,ESTABLISHED
  0      0 ACCEPT    tcp  --  *      *      0.0.0.0/0           0.0.0.0/0           tcp spt:53 state RELATED,ESTABLISHED
  0      0 ACCEPT    tcp  --  eth0   eth1   0.0.0.0/0           0.0.0.0/0           multiport dports 80,443 state NEW,RELATED,ESTABLISHED
Chain OUTPUT (policy DROP 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
  0      0 ACCEPT    icmp --  *      *      0.0.0.0/0           0.0.0.0/0           state NEW,ESTABLISHED
  0      0 ACCEPT    tcp  --  *      *      0.0.0.0/0           0.0.0.0/0           tcp spt:22 state ESTABLISHED
  0      0 ACCEPT    tcp  --  *      *      0.0.0.0/0           0.0.0.0/0           multiport dports 80,443 ctstate NEW,ESTABLISHED
  0      0 ACCEPT    udp  --  *      *      0.0.0.0/0           0.0.0.0/0           udp dpt:53 state NEW,RELATED,ESTABLISHED
  0      0 ACCEPT    tcp  --  *      *      0.0.0.0/0           0.0.0.0/0           tcp dpt:53 state NEW,RELATED,ESTABLISHED
WEBGATEWAY:~#

```

Ces deux pare-feux ont été mis en place automatiquement au démarrage grâce au fichier « /etc/rc.local » :



```

GNU nano 2.2.4 File: /etc/rc.local
ip route add 192.168.10.0/24 via 192.168.50.5 dev eth0
ip route add 192.168.20.0/24 via 192.168.50.5 dev eth0
ip route add 192.168.40.0/24 via 192.168.50.5 dev eth0

/bin/bash /etc/firewall.conf
exit 0
WEBGATEWAY:~#

GNU nano 2.2.4 File: /etc/rc.local
vconfig add eth0 10
vconfig add eth0 20

ifconfig eth0.20 up
ifconfig eth0.20 192.168.40.5 netmask 255.255.255.0
ifconfig eth0.10 up
ifconfig eth0.10 192.168.10.5 netmask 255.255.255.0

service isc-dhcp-relay start

/bin/bash /etc/firewall.conf
exit 0
FIREWALL:~#

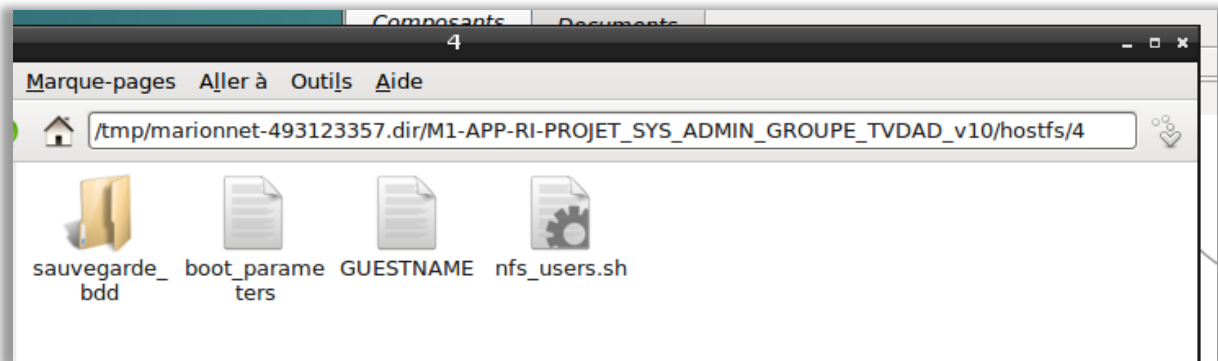
```

Configuration de la gestion des utilisateurs

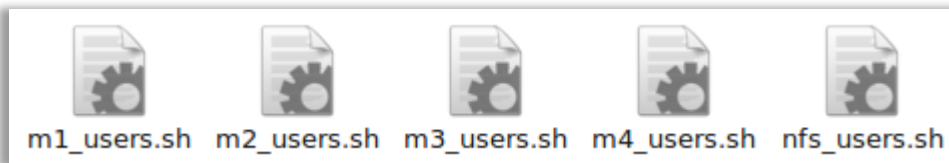
a) Mise en place des groupes et des utilisateurs sur les machines m1, m2, m3, m4

Pour permettre d'automatiser cette partie, nous utilisons un script BASH pour chaque configuration. Ce script a été écrit sur la machine hôte, et est envoyé par l'intermédiaire des dossiers « /tmp/marionnet-XXX.dir/NOM-DU-PROJET-MARIONNET/hostfs/XX ».

Exemple avec la machine NFS :



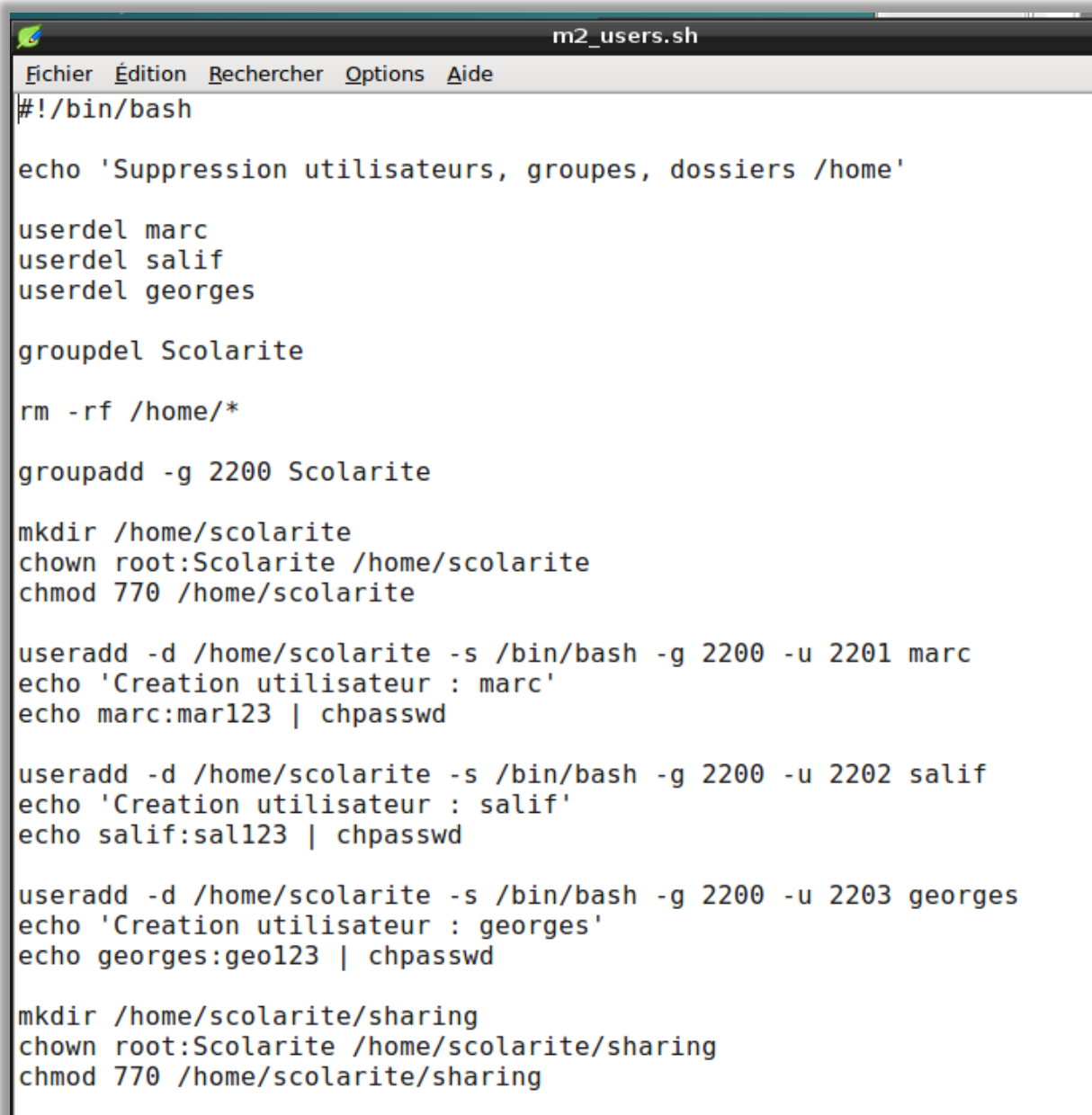
Ces scripts nous permettent de configurer chaque machine automatiquement :



Exemple d'utilisation du script sur la machine M2 :



Exemple du contenu du fichier de M2 :



```
m2_users.sh
Fichier  Édition  Rechercher  Options  Aide
#!/bin/bash

echo 'Suppression utilisateurs, groupes, dossiers /home'

userdel marc
userdel salif
userdel georges

groupdel Sclarite

rm -rf /home/*

groupadd -g 2200 Sclarite

mkdir /home/sclarite
chown root:Sclarite /home/sclarite
chmod 770 /home/sclarite

useradd -d /home/sclarite -s /bin/bash -g 2200 -u 2201 marc
echo 'Creation utilisateur : marc'
echo marc:mar123 | chpasswd

useradd -d /home/sclarite -s /bin/bash -g 2200 -u 2202 salif
echo 'Creation utilisateur : salif'
echo salif:sal123 | chpasswd

useradd -d /home/sclarite -s /bin/bash -g 2200 -u 2203 georges
echo 'Creation utilisateur : georges'
echo georges:geol23 | chpasswd

mkdir /home/sclarite/sharing
chown root:Sclarite /home/sclarite/sharing
chmod 770 /home/sclarite/sharing
```

- Pour les machines M1, M2, M3, M4 et NFS, nous utilisons un script BASH de la même manière pour configurer les utilisateurs, les dossiers et les groupes. Ces fichiers sont accessibles dans les sources.

b) Mise en place du serveur NFS

On configure le serveur NFS avec un script :

```
#!/bin/bash

echo 'Suppression utilisateurs, groupes, dossiers /home'

groupdel Comptabilite
groupdel Scholarite
groupdel Secretariat
groupdel Direction

rm -rf /home/*

groupadd -g 2100 Comptabilite
groupadd -g 2200 Scholarite
groupadd -g 2300 Secretariat
groupadd -g 2400 Direction

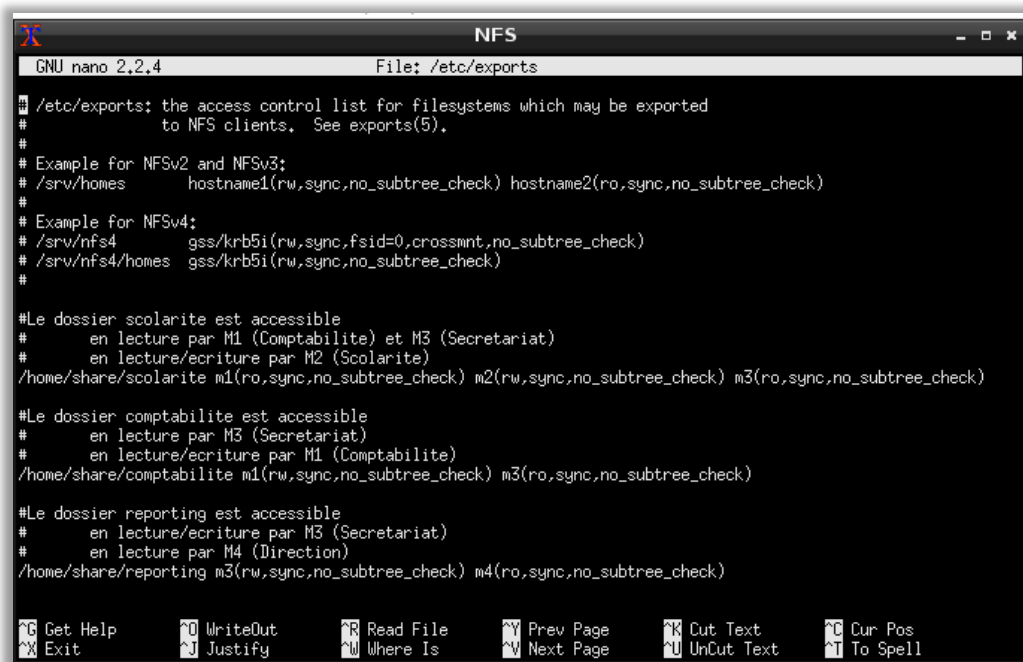
mkdir /home/share

mkdir /home/share/scholarite
chown root:Scholarite /home/share/scholarite
chmod 775 /home/share/scholarite

mkdir /home/share/comptabilite
chown root:Comptabilite /home/share/comptabilite
chmod 775 /home/share/comptabilite

mkdir /home/share/reporting
chown root:Secretariat /home/share/reporting
chmod 775 /home/share/reporting
```

Pour on configure le fichier « /etc/exports ».



```
GNU nano 2.2.4      File: /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
# to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes      hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4       gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
#
#Le dossier scholarite est accessible
#   en lecture par M1 (Comptabilite) et M3 (Secretariat)
#   en lecture/ecriture par M2 (Scholarite)
/home/share/scholarite m1(ro,sync,no_subtree_check) m2(rw,sync,no_subtree_check) m3(ro,sync,no_subtree_check)
#Le dossier comptabilite est accessible
#   en lecture par M3 (Secretariat)
#   en lecture/ecriture par M1 (Comptabilite)
/home/share/comptabilite m1(rw,sync,no_subtree_check) m3(ro,sync,no_subtree_check)
#Le dossier reporting est accessible
#   en lecture/ecriture par M3 (Secretariat)
#   en lecture par M4 (Direction)
/home/share/reporting m3(rw,sync,no_subtree_check) m4(ro,sync,no_subtree_check)

^G Get Help      ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit          ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

Enfin, pour activer la configuration automatique du serveur de manière pérenne, on modifie le fichier « /etc/rc.local ».

```
/etc/init.d/portmap start
/etc/init.d/nfs-kernel-server start

exit 0
```

c) Mise en place des dossiers partagés sur les machines m1, m2, m3, m4

Enfin, pour terminer la configuration du service NFS, on met à jour les fichiers « /etc/rc.local » des machines M1, M2, M3 et M4.

```
mount -t nfs nfs:/home/share/comptabilite /home/comptabilite
mount -t nfs nfs:/home/share/scolarite /home/scolarite

exit 0
```

[Read 17 lines]

m1:~#

```
mount -t nfs nfs:/home/share/scolarite /home/scolarite/sharing

exit 0
```

[Wrote 16 lines]

m2:~#

```
mount -t nfs nfs:/home/share/comptabilite /home/comptabilite
mount -t nfs nfs:/home/share/scolarite /home/scolarite
mount -t nfs nfs:/home/share/reporting /home/lucie/reporting

exit 0
```

[Read 18 lines]

m3:~#

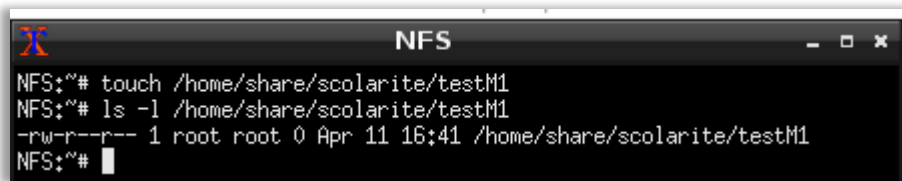
```
mount -t nfs nfs:/home/share/reporting /home/didier/reporting

exit 0
```

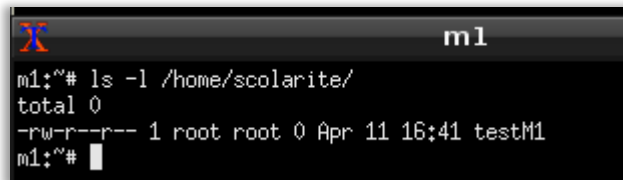
[Wrote 16 lines]

m4:~#

Enfin, on teste avec M1 par exemple :



```
NFS:~# touch /home/share/scolarité/testM1
NFS:~# ls -l /home/share/scolarité/testM1
-rw-r--r-- 1 root root 0 Apr 11 16:41 /home/share/scolarité/testM1
NFS:~#
```



```
m1:~# ls -l /home/scolarité/
total 0
-rw-r--r-- 1 root root 0 Apr 11 16:41 testM1
m1:~#
```

La configuration du serveur NFS est fonctionnelle.

Configuration du serveur Web

a) Vérification de communication entre les VM et la machine linux hôte

Une fois qu'on s'est assuré que la machine hôte debian pouvait communiquer avec les machines du réseau de l'école saint Potache, on peut commencer à mettre en place le serveur web.

A première vue, nous avons eu de la peine à vérifier la connectivité. En effet, le ping d'un appareil du réseau interne est impossible dû à un blocage sur le gateway. En testant avec un ssh la connexion au webgateway de la machine hôte, on voit que la communication est possible.

```
WEBGATEWAY:~# ping 192.168.1.37
PING 192.168.1.37 (192.168.1.37) 56(84) bytes of data.
^C
--- 192.168.1.37 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1003ms

WEBGATEWAY:~# ssh etudiant@192.168.1.37
etudiant@192.168.1.37's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
etudiant@debian:~$ S
```

Test de connexion du webgateway à la machine hôte

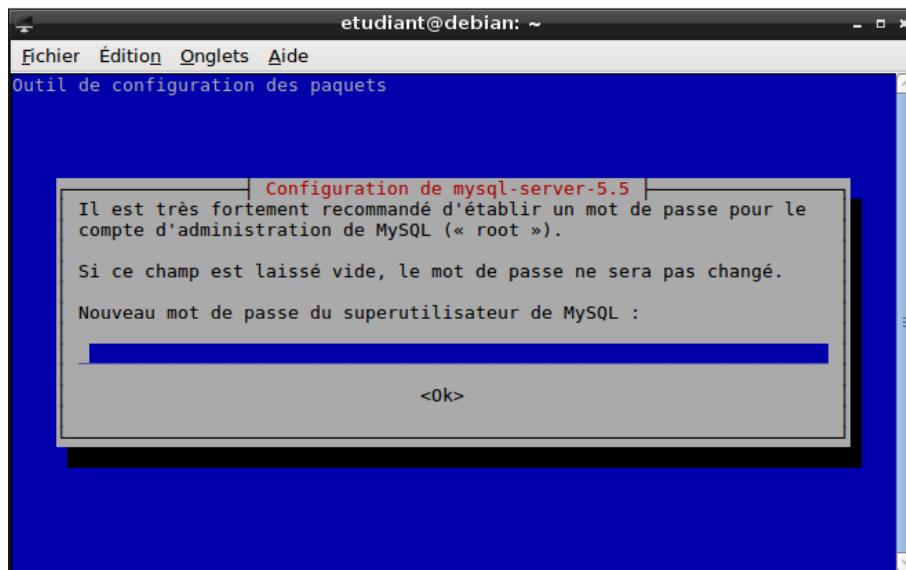
Dans notre cas, on a configuré notre VM hôte en mode bridge. Une interface virtuelle est créée sur notre hôte (eth42) et permet la communication de la machine hôte au réseau interne.

b) Installation des services

On procède à l'installation des différents services utilisés par le serveur web tout ça sur la machine linux hôte.

```
etudiant@debian:~$ sudo apt-get install mysql-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  gstreamer1.0-libav libgconf2-4 libuuid-perl
Veuillez utiliser « apt-get autoremove » pour les supprimer.
Les NOUVEAUX paquets suivants seront installés :
  mysql-server
0 mis à jour, 1 nouvellement installés, 0 à enlever et 6 non mis à jour.
Il est nécessaire de prendre 0 o/73,9 ko dans les archives.
Après cette opération, 121 ko d'espace disque supplémentaires seront utilisés.
Sélection du paquet mysql-server précédemment désélectionné.
(Lecture de la base de données... 133132 fichiers et répertoires déjà installés.
)
Préparation du dépaquetage de .../mysql-server_5.5.62-0+deb8u1_all.deb ...
Dépaquetage de mysql-server (5.5.62-0+deb8u1) ...
Paramétrage de mysql-server (5.5.62-0+deb8u1) ...
etudiant@debian:~$
```

Installation de mySQL



```
etudiant@debian:~$ sudo apt-get install apache2
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  gstreamer1.0-libav libgconf2-4 libuuid-perl
Veuillez utiliser « apt-get autoremove » pour les supprimer.
Les paquets supplémentaires suivants seront installés :
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.1-0 ssl-cert
Paquets suggérés :
  apache2-doc apache2-suexec-pristine apache2-suexec-custom openssl-blacklist
Les NOUVEAUX paquets suivants seront installés :
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.1-0 ssl-cert
0 mis à jour, 10 nouvellement installés, 0 à enlever et 6 non mis à jour.
Il est nécessaire de prendre 1 956 ko dans les archives.
```

Installation de Apache2

```
etudiant@debian:~$ sudo apt-get install php5
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  gstreamer1.0-libav libgconf2-4 libuuid-perl
Veuillez utiliser « apt-get autoremove » pour les supprimer.
Les paquets supplémentaires suivants seront installés :
  libapache2-mod-php5 libonig2 libqdbm14 php5-cli php5-common php5-json
  php5-readline
Paquets suggérés :
  php-pear php5-user-cache
Les NOUVEAUX paquets suivants seront installés :
  libapache2-mod-php5 libonig2 libqdbm14 php5 php5-cli php5-common php5-json
  php5-readline
0 mis à jour, 8 nouvellement installés, 0 à enlever et 6 non mis à jour.
Il est nécessaire de prendre 5 443 ko dans les archives.
Après cette opération, 21,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] o
```

Installation de php5

```
etudiant@debian:~$ sudo apt-get install php5-mysql
php5-mysql      php5-mysqldb      php5-mysqldb-ms
etudiant@debian:~$ sudo apt-get install php5-mysql
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  gstreamer1.0-libav libgconf2-4 libuuid-perl
Veuillez utiliser « apt-get autoremove » pour les supprimer.
Les NOUVEAUX paquets suivants seront installés :
```

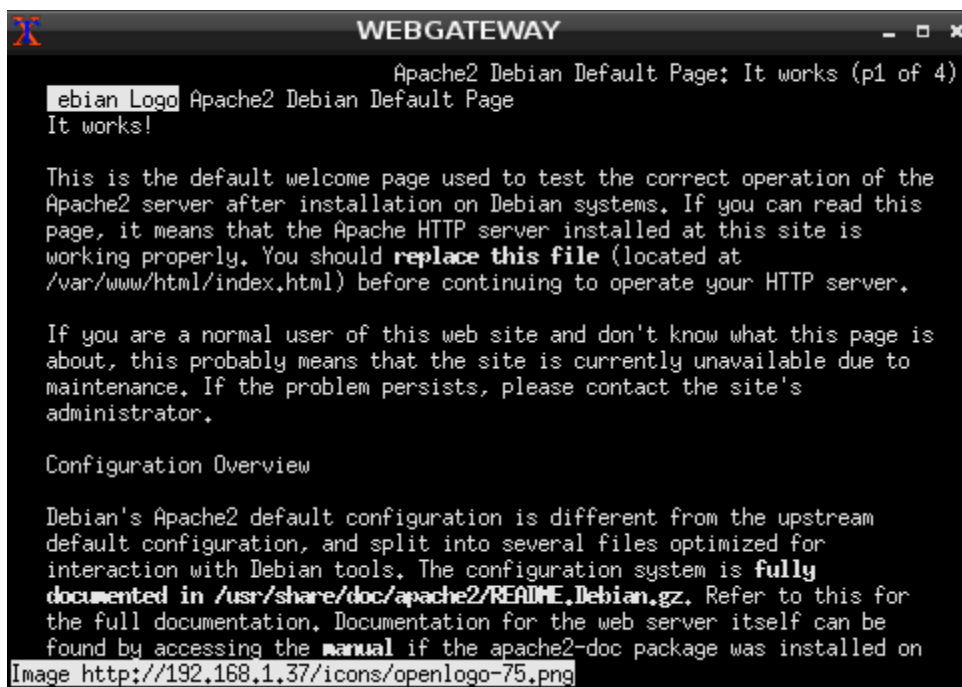
Installation de php5-mysql

c) Start up des services

On lance apache2 :

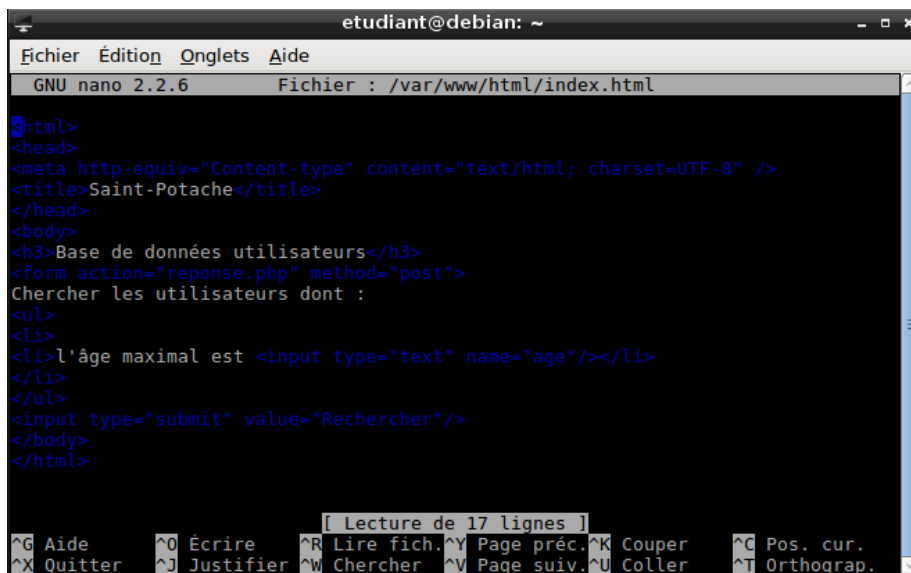
```
etudiant@debian:~$ sudo /etc/init.d/apache2 start
[ ok ] Starting apache2 (via systemctl): apache2.service.
```

Après avoir installé on test le service sur l'adresse de l'hôte depuis la webgateway :



Contenu d'index.html contenu dans /var/www/html

Ensuite on crée les fichiers web index.html pour l'interface en front-end et response.php pour le back-end (code exécuté dans le fichier html). On va placer ces fichiers dans le dossier /var/www/html qui est le fichier racine interprété par le serveur web apache lors de son lancement. Par exemple, en se connectant en http sur le serveur web, celui-ci exécute en premier index.html présent dans le dossier www/html.

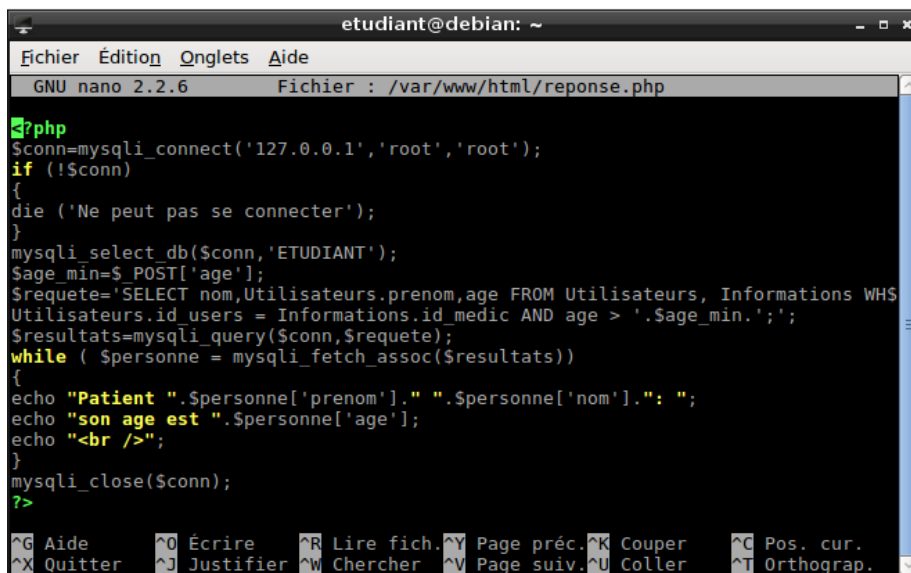


```
etudiant@debian: ~
Fichier  Édition  Onglets  Aide
GNU nano 2.2.6  Fichier : /var/www/html/index.html

<html>
<head>
<meta http-equiv="Content-type" content="text/html; charset=UTF-8" />
<title>Saint-Potache</title>
</head>
<body>
<h3>Base de données utilisateurs</h3>
<form action="reponse.php" method="post">
Chercher les utilisateurs dont :
<ul>
<li>
<li>l'âge maximal est <input type="text" name="age"/></li>
</li>
</ul>
<input type="submit" value="Rechercher"/>
</body>
</html>

[ Lecture de 17 lignes ]
^G Aide      ^O Écrire    ^R Lire fich.^Y Page préc.^K Couper      ^C Pos. cur.
^X Quitter   ^J Justifier ^W Chercher  ^V Page suiv.^U Coller     ^T Orthograp.
```

Réécriture du fichier index.html avec le code présent en annexe

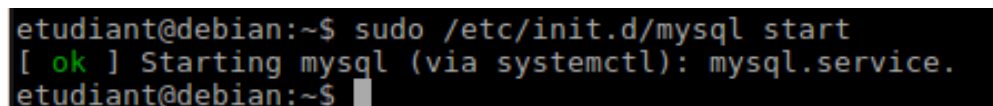


```
etudiant@debian: ~
Fichier  Édition  Onglets  Aide
GNU nano 2.2.6  Fichier : /var/www/html/reponse.php

?php
$conn=mysqli_connect('127.0.0.1','root','root');
if (!$conn)
{
die ('Ne peut pas se connecter');
}
mysqli_select_db($conn,'ETUDIANT');
$age_min=$_POST['age'];
$query="SELECT nom,Utilisateurs.prenom,age FROM Utilisateurs, Informations WHS
Utilisateurs.id_users = Informations.id_medec AND age > ".$age_min."";
$resultats=mysqli_query($conn,$query);
while ( $personne = mysqli_fetch_assoc($resultats))
{
echo "Patient ".$personne['prenom']." ".$personne['nom'].": ";
echo "son age est ".$personne['age'];
echo "<br />";
}
mysqli_close($conn);
?>
```

Ecriture du fichier php

On va ensuite procéder à la mise en place de la base de données SQL, on lance d'abord le service mysql.



```
etudiant@debian:~$ sudo /etc/init.d/mysql start
[ ok ] Starting mysql (via systemctl): mysql.service.
etudiant@debian:~$
```

Pour accéder à l'interface MySQL, on utilise la commande suivante, elle signifie qu'on se connecte via l'utilisateur root possédant un mot de passe, on entre ensuite le mot de passe établi à l'installation du service (root).

```
etudiant@debian:~$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 41
Server version: 5.5.62-0+deb8u1 (Debian)

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.

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

mysql> █
```

On ajoute ensuite les commandes sql suivantes :

```
mysql> CREATE DATABASE ETUDIANT;
Query OK, 1 row affected (0.00 sec)

mysql> USE ETUDIANT;
Database changed
mysql> CREATE TABLE Utilisateurs(
  -> `id_users` int(15) NOT NULL AUTO_INCREMENT,
  -> `nom` varchar(15) NOT NULL,
  -> `prenom` varchar(50) NOT NULL,
  -> `login` varchar(50) NOT NULL,
  -> `password` varchar(500) NOT NULL,
  -> PRIMARY KEY (`id_users`)
  -> ) ENGINE=InnoDB DEFAULT CHARSET=latin1 ;
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE Informations (
  -> `id_medec` int(15) NOT NULL AUTO_INCREMENT,
  -> `prenom` varchar(50) NOT NULL,
  -> `age` int(20) NOT NULL,
  -> `poids` int(20) NOT NULL,
  -> PRIMARY KEY (`id_medec`)
  -> ) ENGINE=InnoDB DEFAULT CHARSET=latin1 ;
Query OK, 0 rows affected (0.01 sec)

mysql> INSERT INTO `Utilisateurs` (`id_users`, `nom`, `prenom`, `login`, `password`) VALUES
  -> (1,'BIDOCHON','Robert','rob_le_ouf','mdrlol'),
  -> (2,'YAU','Tatiana','tatayoyo','mdrlol'),
  -> (3,'OURSON','Winnie','ton_ami','mdrlol'),
  -> (4,'DUPONT','Pierre','pierrot75','mdrlol');
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

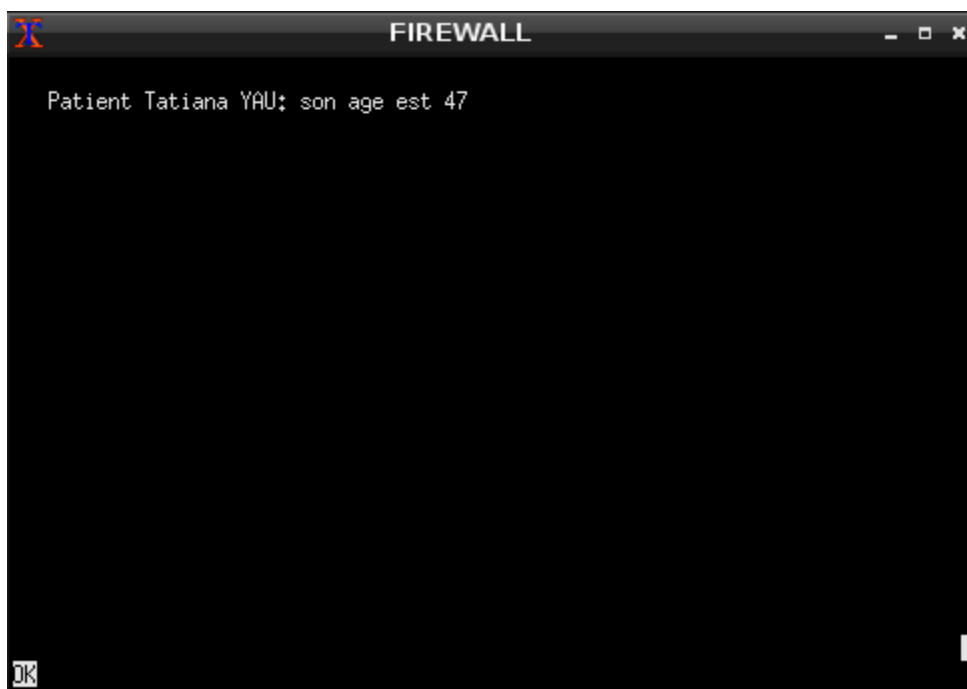
mysql> INSERT INTO `Informations` (`id_medec`, `prenom`, `age`, `poids`) VALUES
  -> (1,'BIDOCHON',34,100),
  -> (2,'YAU',47,60),
  -> (3,'OURSON',20,110),
  -> (4,'DUPONT',4,20);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> █
```

Ces instructions ont permis de créer une base de données nommée ETUDIANT et possédant deux tables liées informations et utilisateurs possédant les attributs de divers utilisateurs et leurs informations.

Une fois la base de données et les fichiers web créés, on peut relancer le serveur apache et tester le fonctionnement.

On teste depuis le FIREWALL avec la commande links :



L'accès au site web fonctionnel.

Exploitation

a) Problème initial sur la mise en place de cette partie

Nous nous sommes vite rendu compte que la communication machines > hôte fonctionnait tout à fait normalement. Néanmoins, la communication hôte > machine elle était plus complexe. Il était donc impossible de *mount* le nfs sur notre machine hôte.

Solution : Nous sommes passés par /mnt/hostfs du nfs pour créer un dossier /sauvegarde_bdd.

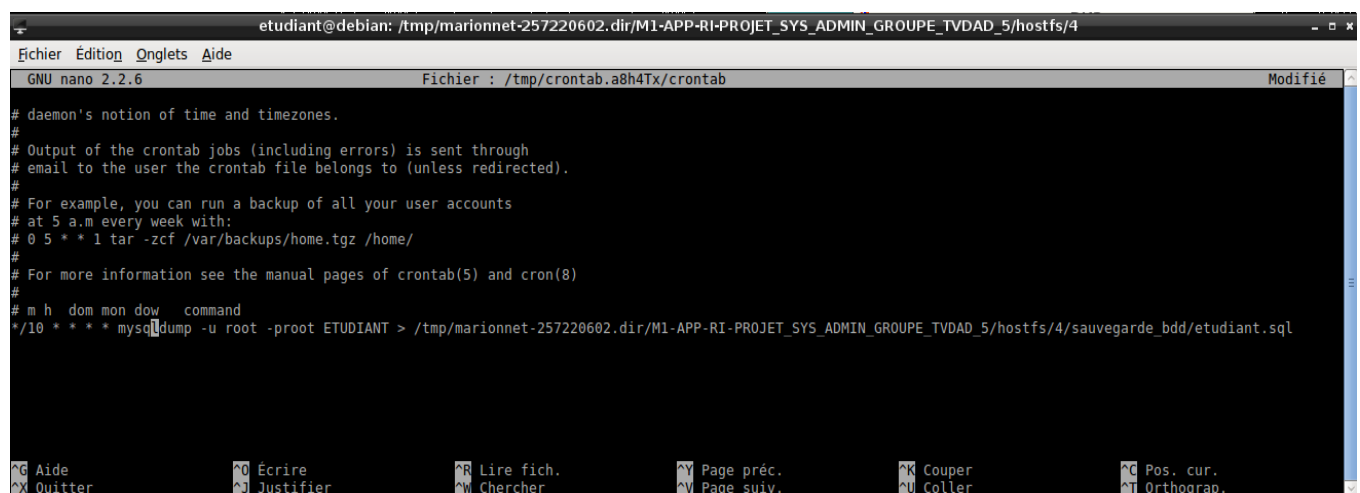
b) Mise en place

Sur la machine **NFS**, on crée le dossier *sauvegarde_bdd* :

```
NFS:/mnt/hostfs# chmod 600 sauvegarde_bdd/
NFS:/mnt/hostfs# ls -alh
total 28K
drwxr-xr-x 4 1000 1000 4.0K Mar 25 15:10 .
drwxr-xr-x 3 root root 4.0K May 6 2012 ..
drwxr-xr-x 5 1000 1000 4.0K Apr 10 12:21 .corauto
drwxr-xr-x 2 1000 1000 4.0K Apr 1 15:28 .history
-rw-r--r-- 1 1000 1000 4 Apr 10 12:24 GUESTNAME
-rwxr-xr-x 1 1000 1000 199 Apr 10 12:22 boot_parameters
drw----- 2 1000 1000 4.0K Apr 10 12:40 sauvegarde_bdd
```

Ensuite on met les droits de lecture et d'écriture à root.

Depuis notre machine hôte, on lance la commande `crontab -e` en super-utilisateur :



```
etudiant@debian: /tmp/marionnet-257220602.dir/M1-APP-RI-PROJET_SYS_ADMIN_GROUPE_TVDAD_5/hostfs/4
GNU nano 2.2.6 Fichier : /tmp/crontab.a8h4Tx/crontab
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow command
*/10 * * * * mysqldump -u root -proot ETUDIANT > /tmp/marionnet-257220602.dir/M1-APP-RI-PROJET_SYS_ADMIN_GROUPE_TVDAD_5/hostfs/4/sauvegarde_bdd/etudiant.sql
```

(* /10 correspond à 10 minutes)

Le reste de la commande permet de sauvegarder la base de données SQL sur la partie « /mnt/hostfs/sauvegarde_bdd ». Ainsi toutes les 10 minutes la sauvegarde est effectuée sur le NFS.

Pour aller plus loin...

a) Scripting

Scripting : configuration utilisateurs

Pour permettre une facilitation de la configuration des machines et un lancement rapide des services, nous avons établi quelques scripts à lancer sur les machines.

Pour mettre en place les configurations utilisateurs sur les machines m1, m2, m3 et m4, nous avons préparé des scripts à exécuter sur chaque machine. Pour faciliter leur lancement, ils sont automatiquement trouvables dans le dossier /root de chaque machine après exécution du fichier .mar sur Marionnet. Il suffit de les exécuter.

```
m1:~# ls
m1_users.sh
m1:~# pwd
/root
m1:~# ./m1_users.sh
Suppression utilisateurs, groupes, dossiers /home
Creation utilisateur : robert
Creation utilisateur : tatiana
m1:~#
```

Une fois fait, la configuration utilisateur est à jour sur les machines.

```
1  #!/bin/bash
2
3  echo 'Suppression utilisateurs, groupes, dossiers /home'
4
5  userdel robert
6  userdel tatiana
7
8  groupdel Comptabilite
9  groupdel Scolarite
10
11 rm -rf /home/*
12
13 groupadd -g 2100 Comptabilite
14 groupadd -g 2200 Scolarite
15
16 useradd -d /home/robert -s /bin/bash -g 2100 -m -u 2101 robert
17 echo 'Creation utilisateur : robert'
18 echo robert:rob123 | chpasswd
19 #passwd robert
20
21 useradd -d /home/tatiana -s /bin/bash -g 2100 -m -u 2102 tatiana
22 echo 'Creation utilisateur : tatiana'
23 echo tatiana:tat123 | chpasswd
24 #passwd tatiana
25
26 chmod 700 /home/*
27
28 mkdir /home/comptabilite
29 mkdir /home/scolarite
30 chown root:Comptabilite /home/comptabilite
31 chown root:Scolarite /home/scolarite
32 chmod 770 /home/comptabilite
33 chmod 775 /home/scolarite
```

Aperçu du fichier de script pour m1

Scripting : configuration NFS

Dans le même temps, nous avons également établi un script de configuration des dossiers associés et des groupes d'utilisateurs dans le NFS. Tout comme pour la configuration utilisateur, son lancement est facile et le script se trouve dans le dossier /root de la machine.

```
NFS:~# ./nfs_users.sh  
Suppression utilisateurs, groupes, dossiers /home  
NFS:~#
```

Scripting : création du webServeur, de la base de données et des fichiers associés.

Pour la mise en place du serveur web, nous avons créé un script pour mettre en place le serveur web de A à Z. Le script permet notamment l'installation des packages nécessaires au fonctionnement du serveur (apache, mysql, php-mysql). Il crée également automatiquement les fichiers html et php aux bons endroits et avec le bon contenu. Il permet la création de la base de données mysql nécessaire pour enfin relancer les services et rendre le serveur web totalement fonctionnel et exécutable.

Lors de son lancement il conviendra cependant d'établir le mot de passe du user root de la base de données à root comme dans la consigne.

Il suffit d'exécuter le script dans la machine hôte (SiteWEB_HOST.sh est trouvable en annexe).

```
etudiant@debian:~$ ./script.sh  
Atteint http://security.debian.org jessie/updates InRelease  
Ign http://ftp.fr.debian.org jessie InRelease  
Atteint http://ftp.fr.debian.org jessie-updates InRelease  
Atteint http://ftp.fr.debian.org jessie Release.gpg  
Atteint http://security.debian.org jessie/updates/main Sources  
Atteint http://ftp.fr.debian.org jessie Release  
Atteint http://security.debian.org jessie/updates/main amd64 Packages  
Atteint http://security.debian.org jessie/updates/main Translation-en  
Atteint http://ftp.fr.debian.org jessie-updates/main Sources  
Atteint http://ftp.fr.debian.org jessie-updates/main amd64 Packages  
Atteint http://ftp.fr.debian.org jessie-updates/main Translation-en  
Atteint http://ftp.fr.debian.org jessie/main Sources  
Atteint http://ftp.fr.debian.org jessie/main amd64 Packages  
Atteint http://ftp.fr.debian.org jessie/main Translation-fr  
Atteint http://ftp.fr.debian.org jessie/main Translation-en  
Lecture des listes de paquets... Fait  
Lecture des listes de paquets... Fait  
Construction de l'arbre des dépendances  
Lecture des informations d'état... Fait  
links est déjà la plus récente version disponible.  
Les paquets suivants ont été installés automatiquement et ne sont plus nécessair
```

b) Amélioration du firewall, filtrage webgateway

En plus de la configuration du pare-feu établie dans le cahier des charges, nous avons ajouté des règles de filtrage sur le webgateway pour la communication vers l'extérieur ([voir configuration du Pare-feu partie b](#)). Ces règles permettent une meilleure sécurité et un risque moindre d'attaques provenant de l'extérieur.

c) Configuration du DDNS

Nous avons tenu à mettre en place le serveur DNS en mode dynamique malgré son caractère facultatif. Cette étape fut compliquée mais elle nous permet une meilleure stabilité de notre réseau. Les machines sont ainsi toutes continuellement atteignable malgré la mise à jour éventuelle de leurs adresses ip. Les appareils sont donc tous associés à un nom de domaine non-soumis à un changement éventuel d'adresse ip ([voir partie DNS associée](#)).

d) Lancement automatique des services avec rc.local

Le fichier rc.local situé dans le dossier /etc des machines permet d'exécuter les commandes qui y sont présentes lors du démarrage de la machine. Nous avons utilisé ce fichier pour automatiser le lancement des services dans les différentes machines.

Rc.local du pare-feu et du WebGateway

Nous avons ajouté la mise en place des vlans et des sous-interfaces associés à ceux-ci dans le fichier rc.local du pare-feu, nous y exécutons également le service isc-dhcp-relay ainsi que le système de firewalling.

Même procédé pour le webgateway où nous avons également mis en place des règles de firewalling ([voir partie associée au filtrage via le webgateway](#)). Nous ajoutons également des règles de routage via le firewall.



```
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

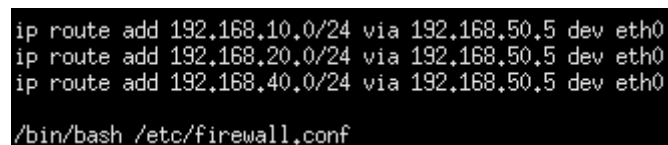
vconfig add eth0 10
vconfig add eth0 20

ifconfig eth0.20 up
ifconfig eth0.20 192.168.40.5 netmask 255.255.255.0
ifconfig eth0.10 up
ifconfig eth0.10 192.168.10.5 netmask 255.255.255.0

service isc-dhcp-relay start

/bin/bash /etc/firewall.conf
```

Rc.local du firewall



```
ip route add 192.168.10.0/24 via 192.168.50.5 dev eth0
ip route add 192.168.20.0/24 via 192.168.50.5 dev eth0
ip route add 192.168.40.0/24 via 192.168.50.5 dev eth0

/bin/bash /etc/firewall.conf
```

Rc.local du webgateway

Rc.local des machines m1, 2, 3, 4, NFS, DNS et DHCP

Sur les machines m1, 2, 3, 4 le rc.local permet de lancer le montage nfs automatiquement.

```
mount -t nfs nfs:/home/share/comptabilite /home/comptabilite
mount -t nfs nfs:/home/share/scolarite /home/scolarite
exit 0
m1:~#
```

Rc.local sur m1

Sur les machines NFS, DHCP et DNS, on lance via le rc.local les services nécessaires à leur fonctionnement respectif :

```
/etc/init.d/portmap start
/etc/init.d/nfs-kernel-server start
exit 0
NFS:~#
```

Rc.local sur NFS

```
service bind9 start
exit 0
DNS:~#
```

Rc.local sur DNS

```
service isc-dhcp-server start
exit 0
```

Rc.local sur DHCP

Pour résumer, une fois le fichier .mar possédé, il suffit de lancer les machines dans le bon ordre avec leurs scripts ainsi qu'exécuter le script sur la machine hôte.

La seule action manuelle à entreprendre est la création des VLAN sur le switch. Nous n'avons trouvé aucun système d'automatisation, nous avons envisagé le lancement d'un script via la commande load mais l'accès au dossier associé au switch était impossible.