

I. Compte rendu TP3



Quand la commande « `service` » n'est pas installée télécharger le paquet avec « `apt-get install sysvinit-utils` » puis rajouter dans le fichier « `~/.bashrc` » la ligne « `PATH=$PATH:/usr/sbin` » et relancer le tout avec la commande « `source ~/.bashrc` ».

1. Configuration DNS maître (machine Debian), serveur de type hybride (master + cache)

Dans un premier temps, j'ai installé un serveur DNS sur ma machine Debian. Cette [documentation](#) m'a été utile pour avoir plus d'information sur Bind9. Pour télécharger ce serveur j'ai rentré la commande suivante « `apt-get install bind9` ».

J'ai modifié le fichier « `/etc/default/named` » (on le retrouve parfois sous le nom de `etc/default/bind9`). Je ne sais pas si c'était nécessaire, cela permet en tout cas de désactiver les erreurs liées à IPv6 quand on ne l'utilise pas. Voici le fichier après modifications :

```
# run resolvconf?
RESOLVCONF=yes

# startup options for the server
OPTIONS="-4 -u bind"
```

a. Vous définirez donc le domaine suivant `grp1.mrt`.

Hostname

Dans un premier temps j'ai changé le nom présent dans le fichier « `/etc/hostname` » par « `grp1.mrt` ».

Interfaces réseau

Puis, j'ai rajouté dans le fichier de configuration des interfaces réseaux « `/etc/network/interfaces` » la ligne suivante dans les paramètres de ma carte réseau `enp0s3` :

```
dns-nameservers 172.18.10.10
```

L'adresse du serveur DNS est donc l'adresse de ma carte réseau.

Resolv.conf

J'ai également modifié le fichier « `resolv.conf` » ainsi :
`nano /etc/resolv.conf`

```
# Generated by NetworkManager
domain grp1.mrt
search grp1.mrt
nameserver 172.18.10.10
nameserver 8.8.8.8
```

named.conf.local

Ensuite, j'ai configuré le fichier « `/etc/bind/named.conf.local` » (si pas « `.local` » ne fonctionne pas !) qui permet d'avoir accès aux déclarations des zones DNS. Il faut rajouter une zone DNS pour chaque site internet hébergé sur le serveur.

J'ai donc rajouté une zone de nom « `grp1.mrt` » dans le fichier configuration « `/etc/bind/named.conf.local` ».

J'ai spécifié qu'elle était de type master et la localisation de son fichier de configuration que j'ai créé par la suite.

J'ai également rajouté la zone qui permet de faire la résolution DNS inverse.

Voici donc le code que j'ai rajouté dans le fichier « `named.conf.local` »:

```
zone "grp1.mrt" IN {
    type master;
    file "/var/lib/bind/db.grp1.mrt";
};

zone "01.81.271.in-addr.arpa" IN {
    type master;
    file "/var/lib/bind/db.grp1.mrt.inv";
};
```

Il est important d'écrire l'adresse IP complètement à l'envers.

- a. Dans cette configuration du serveur DNS vous associez aux différents serveurs les noms suivants :

Alpine	ip1.grp1.mrt
Debian	ip2.grp1.mrt
Ubuntu	ip3.grp1.mrt

J'ai donc par la suite créé le fichier de configuration de la zone.

Il suffit généralement de copier le fichier « `db.local` » et de changer les paramètres.

Les premières lignes permettent de rentrer les informations de la zone.

Les autres informations sont :

Serial	Il faut incrémenter ce paramètre à chaque fois qu'on modifie ce fichier. Le format est YYYYMMDDXX. Par exemple j'ai fait ce TP le 07/12/2022 donc le numéro est 2022120701.
Refresh	Définit la période de rafraichissement des données.
Retry	Si une erreur survient lors de la dernière actualisation, elle sera répétée à la fin de ce temps.
Expire	Le serveur est considéré comme indisponible après l'expiration de ce délai.

Le fichier comprend donc les lignes suivantes :

```
$TTL      3600
@         IN      SOA      ip2.grp1.mrt.  root.grp1.mrt (
                        2022120701      ; Serial
                        3600             ; Refresh [1h]
                        600              ; Retry  [10m]
                        86400            ; Expire  [1d]
                        600 )           ; Negative Cache TTL [1h]
;

@         IN      NS       ip2.grp1.mrt.
@         IN      A        172.18.10.10
ip2       IN      A        172.18.10.10
ip3       IN      A        172.18.10.12
ip1       IN      A        172.18.10.11
```

Le « A » permet d'associer un nom d'hôte à une adresse IPv4.

J'ai ensuite créé le fichier pour la résolution inverse. Voici le contenu de ce fichier, nommé « db.grp1.mrt.inv » :

```
$TTL      3600
@         IN      SOA      ip2.grp1.mrt. root.grp1.mrt (
                        2022120701      ; Serial
                        3600             ; Refresh [1h]
                        600              ; Retry  [10m]
                        86400            ; Expire  [1d]
                        600 )            ; Negative Cache TTL [1h]
                        ;

@         IN      NS       ip2.grp1.mrt.
12        IN      PRT      ip3.grp1.mrt.
10        IN      PRT      ip2.grp1.mrt.
11        IN      PRT      ip1.grp1.mrt.
```

Pour vérifier j'ai tapé la commande « dig 172.18.10.10 » ou « nslookup ip2.grp1.mrt » et cela a fonctionné :

```
root@grp1:/var/lib/bind# nslookup ip2.grp1.mrt
Server:         172.18.10.10
Address:        172.18.10.10#53

Name:   ip2.grp1.mrt
Address: 172.18.10.10
```

2. DNS et serveurs WEB

a. Sur chacune des machines vous installerez un serveur apache.

La méthodologie est la même pour la machine Ubuntu et Debian.

La configuration qui suit a été faite sur la machine Debian d'adresse IP 172.18.10.10.

La configuration de l'Alpine est un peu différente, elle est donc expliquée en détail à la fin de la partie 2.b.

J'ai téléchargé le serveur apache :

```
apt-get update
apt-get install apache2
```

b. Vous ferez une page minimale sur chaque serveur WEB, qui affiche le nom de la machine sur laquelle on se trouve. Chaque serveur devra offrir des sites différents pour chacun des accès suivants :

Alpine	Adresse de l'Alpine
	i1.grp1.mrt
Debian	Adresse de la Debian
	i2.grp1.mrt
Ubuntu	Adresse de l'Ubuntu
	i3.grp1.mrt

Apache est capable de gérer plusieurs sites web sur la même machine, c'est ce qu'on appelle des hôtes virtuels ou virtual hosts. Dans la requête que fera le client sur la machine, un en-tête HTTP précisera si le client veut plutôt consulter toto.com ou tata.org, qui sont tous les deux hébergés sur la machine. La configuration de chaque virtual host se fait dans un fichier *.conf dans /etc/apache2/sites-available/.

Voici les configurations faites pour créer le site i1.grp1.mrt :

```
nano /etc/apache2/sites-available/i2.grp1.mrt.conf
```

```
<VirtualHost *:80>
    ServerName i2.grp1.mrt
    ServerAdmin webmaster@i2.grp1.mrt
    DocumentRoot /var/www/html/i2.grp1.mrt
</VirtualHost>
```

^X

```
mkdir /var/www/html/i2.grp1.mrt/
```

```
nano /var/www/html/i2.grp1.mrt/index.html
```

```
<!DOCTYPE html>
<html>
<body>

<h1>Weclome on i2.grp1.mrt </h1>
<p>It's a Debian VM</p>

</body>
</html>
```

^X

```
a2ensite i2.grp1.mrt
```

```
systemctl reload apache2
```

Voici les configurations faites pour créer le site 172.18.10.10 :

```
nano /etc/apache2/sites-available/i2.conf
```

```
<VirtualHost *:80>
    ServerName 172.18.10.10
    ServerAdmin webmaster@i172.18.10.10
    DocumentRoot /var/www/html/i2
</VirtualHost>
```

^X

```
mkdir /var/www/html/i2/
```

```
nano /var/www/html/i2/index.html
```

```
<!DOCTYPE html>
<html>
<body>

<h1>Weclome on my Debian</h1>
<p>172.18.10.10</p>

</body>
</html>
```

`^X`

```
a2ensite i2
systemctl reload apache2
```

Enfin, j'ai rentré la commande « `a2enmod rewrite vhost_alias` »

Configurer la machine Alpine :

Pour Alpine, la commande pour télécharger un paquet est : `apk add nom_du_paquet`

Il faut également lancer le serveur web avec la commande : `rc-service apache2 start`

Voici dans l'ordre les commandes qui ont été réalisées. Le principe est le même que pour les machines précédentes mais l'architecture des dossiers est différente. Il n'y a pas par exemple le dossier `sites-available`.

Création du premier site « `i1.grp1.mrt` » :

```
mkdir -p /var/www/i1.grp1.mrt/html

chown -R apache:apache /var/www/i1.grp1.mrt/
chmod -R 755 /var/www/i1.grp1.mrt/

nano /var/www/i1.grp1.mrt/html/index.html

    <!DOCTYPE html>
    <html>
    <body>

    <h1>Weclome on i1.grp1.mrt </h1>
    <p>It's a Alpine VM</p>

    </body>
    </html>
^X

nano /etc/apache2/conf.d/i1.grp1.mrt.conf
    <VirtualHost 172.18.10.11:80>
        ServerName i1.grp1.mrt
        ServerAdmin webmaster@i1.grp1.mrt
        DocumentRoot /var/www/i1.grp1.mrt/html
    </VirtualHost>
```

`^X`

```
service apache2 restart
dig i1.grp1.mrt
```

Création du deuxième site « `172.18.10.11` » :

```
mkdir -p /var/www/i1/html

chown -R apache:apache /var/www/i1/
chmod -R 755 /var/www/i1/

nano /var/www/i1/html/index.html

    <!DOCTYPE html>
    <html>
    <body>
```

```
<h1>Weclome on my Alpine </h1>
<p>172.18.10.11</p>

</body>
</html>
```

^X

```
nano /etc/apache2/conf.d/i1.conf
<VirtualHost 172.18.10.11:80>
    ServerName 172.18.10.11
    ServerAdmin webmaster@i1172.18.10.11
    DocumentRoot /var/www/i1/html
</VirtualHost>
```

^X

```
service apache2 restart
dig 172.18.10.11
```

Mais il est **impératif** de changer la ligne « AllowOverride » qui se trouve dans le fichier de configuration « /etc/apache2/httpd.conf ». Mettre cette ligne :

AllowOverride AuthConfig FileInfo Limit Options=Indexes

3. [DNS esclave](#)

a. Installer les différents sites dans le serveur DNS maître

J'ai modifié le fichier « /var/lib/bind/db.grp1.mrt » ainsi :

```
$TTL      3600
@          IN      SOA      ip2.grp1.mrt.  root.grp1.mrt (
                        2022120701      ; Serial
                        3600              ; Refresh [1h]
                        600               ; Retry  [10m]
                        86400             ; Expire  [1d]
                        600 )             ; Negative Cache TTL [1h]
;

@          IN      NS       ip2.grp1.mrt.
@          IN      A        172.18.10.10
ip2        IN      A        172.18.10.10
ip3        IN      A        172.18.10.12
ip1        IN      A        172.18.10.11
i2         IN      A        172.18.10.10
i3         IN      A        172.18.10.12
i1         IN      A        172.18.10.11
```

Ne pas oublier de relancer le service bind9.

Enfin j'ai testé lynx 172.18.10.10, lynx 172.18.10.12, lynx i2.grp1.mrt et lynx i3.grp1.mrt . Mes 4 pages différentes se sont bien affichées.

b. Installer sur Ubuntu un serveur DNS esclave

Dans un premier temps il a fallu faire quelques modifications sur la machine Debian :

```
nano /etc/bind/named.conf
```

J'ai rajouté dans les deux zones ces lignes :

```
allow-transfer {172.18.10.12;};
also-notify {172.18.10.12;};
```

Puis j'ai relancé le service avec « `service bind9 restart` ».

Ensuite, sur la machine Ubuntu j'ai téléchargé le paquet `bind9` et créé les zones permettant de mettre en place le serveur DNS esclave :

```
apt-get update
apt-get install bind9

nano /etc/bind/named.conf

    zone "grp1.mrt" IN {
        type slave;
        file "/var/lib/bind/db.grp1.mrt";
        masters{ 172.18.10.10; };
    };

    zone "172.18.10.in-addr.arpa" IN {
        type slave;
        file "/var/lib/bind/db.grp1.mrt.inv";
        masters{172.18.10.10; };
    };
^X

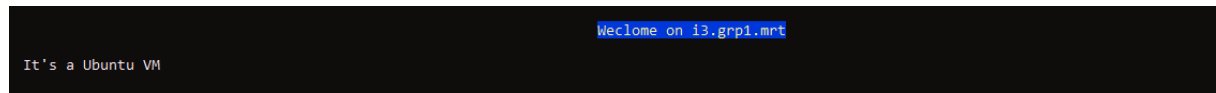
service bind9 restart
```

c. Configurer les accès aux serveurs DNS sur les différents serveurs.

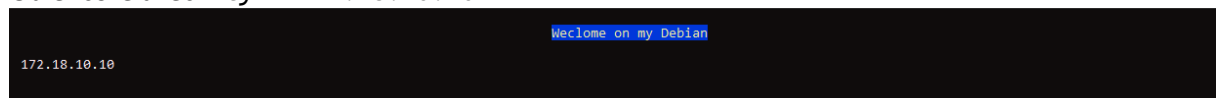
Pour configurer les accès aux serveurs DNS j'ai rajouté dans le fichier « `/etc/resolv.conf` » de l'Alpine la ligne « `nameserver 172.18.10.12` ».

Cela m'a permis de voir si mon serveur esclave fonctionnait avec la commande « `lynx i3.grp1.mrt` ».

Cela a bien fonctionné puisque le site s'est affiché correctement :



Ou encore avec « `lynx 172.18.10.10` » :



4. Test de configuration DNS

a. Présentez par des commandes le bon fonctionnement du serveur DNS maître

Testé dans la partie 2.b.

Pour tester il est utile si nous n'avons pas la possibilité d'avoir le paquet `lynx` d'utiliser `dig`.

b. Présentez par des commandes le bon fonctionnement la fonctionnalité de cache du serveur maître

Le serveur BIND9 est configuré par défaut en tant que serveur cache. Il suffit simplement d'ajouter les serveurs DNS de votre prestataire Internet. Il faut se rendre dans « `/etc/bind/named.conf.options` » et rajouter les adresses IP des serveurs DNS de son prestataire Internet.

```
forwarders {
    212.27.40.240;
    212.27.40.241;
};
```

```
service bind9 restart
```

Pour tester si le cache fonctionne j'ai utilisé la commande dig sur un site où je ne m'étais pas rendu :

```
dig yahoo.com
```

Ce qui m'a redonné ces informations et un temps de 60ms:

```
root@grp1:~# dig yahoo.com

;<<>> DiG 9.16.33-Debian <<>> yahoo.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60536
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: f832c6b0d3991662010000063b83127a6dcd5aed9d4335d (good)
;; QUESTION SECTION:
;yahoo.com.                        IN      A

;; ANSWER SECTION:
yahoo.com.      1377    IN      A       74.6.143.26
yahoo.com.      1377    IN      A       74.6.231.20
yahoo.com.      1377    IN      A       74.6.231.21
yahoo.com.      1377    IN      A       98.137.11.163
yahoo.com.      1377    IN      A       98.137.11.164
yahoo.com.      1377    IN      A       74.6.143.25

;; Query time: 60 msec
;; SERVER: 172.18.10.10#53(172.18.10.10)
;; WHEN: Fri Jan 06 15:33:11 CET 2023
;; MSG SIZE rcvd: 162
```


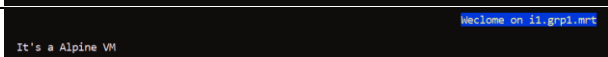

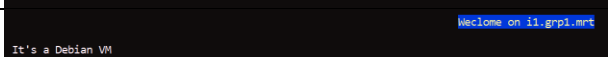
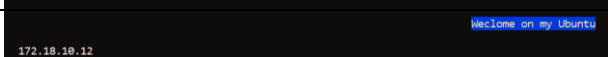
Quand j'ai relancé la même commande on remarque que le temps est maintenant beaucoup plus petit, étant même de 0ms. Le cache fonctionne donc correctement.

```
;; Query time: 0 msec
```

c. Présentez par des commandes le bon fonctionnement du serveur esclave

Testé dans la partie 3.b.

Mais pour confirmer voici les résultats obtenus depuis la machine Debian. La machine Alpine a comme serveur DNS l'adresse du serveur Ubuntu (esclave) et la machine Ubuntu l'adresse du serveur DNS maître de la machine Debian :

Affiche une page web de la machine ...	Commande	Résultat
Alpine	lynx 172.18.10.11	
	lynx i1.grp1.mrt	
Debian	lynx 172.18.10.10	
	lynx i2.grp1.mrt	
Ubuntu	lynx 172.18.10.12	
	lynx i3.grp1.mrt	