INFO3 -RESEAUX ET ADMINISTRATION
Professeur Mohamed Yassine SAMIRI
my.samiri@upm.ac.ma

TP2 DNS sous Linux

1. Environement

- Le domaine que nous allons configurer sera : **upm.local.**
- Ce nom de domaine sera géré par deux serveurs dns :

  ns1.upm.local - 192.168.1.101 sera notre serveur maître ;

  ns2.upm.local - 192.168.1.102 sera notre serveur esclave.

- Les adresses email de ce nom de domaine seront gérées par deux serveurs de messagerie :

  mx1.upm.local - 192.168.1.103 ;

  mx2.upm.local - 192.168.1.104.

- Ce nom de domaine possédera deux machines :

  cours.upm.local - 192.168.1.105 ;

  www.upm.local - 192.168.1.106.

- Il existera aussi une autre machine, **blog.upm.local** , qui sera un alias de **www.upm.local**.

Nous ne connaissons pas les serveurs de messagerie, mais vous devez simplement savoir que pour chaque domaine, il doit y avoir un serveur de messagerie qui permet de recevoir des mails pour les adresses de notre domaine.

2. Installation et configuration de BIND9

a. Machines à configurer

On aura besoin de 3 machines virtuelles :

- Une machine pour le serveur DNS maître (primaire) ubuntu server
- Une machine pour le serveur DNS esclave (secondaire) ubuntu server
- Une machine cliente

Les machine serveurs (maître et esclave) doivent être en premier temps connectées à internet (Mode NAT sous VMware) afin d'installer les packages nécessaires pour le serveur DNS.

Après la fin de l'installation des services et pour éviter tout conflits, les trois machines seront ensuite isolées dans le même réseau virtuelle (Virtual Network).

b. Configuration du serveur DNS maître

Lancer la machine virtuelle et installer BIND9.

apt-get install bind9

Après avoir terminer l'installation, la configuration se fait en deux temps. Nous devons tout d'abord déclarer à notre serveur quels seront les noms de domaine qu'il va devoir gérer, on appelle ça des zones. Ensuite, nous devrons configurer ces zones, grâce à un fichier de configuration par zones.

Pour ajouter une zone DNS à BIND9 tout ce que vous avez à faire est d'éditer le fichier **/etc/bind/named.conf.**

sudo nano /etc/bind/named.conf

Et on ajoute la zone d'autorité

```
zone "upm.local" {
                type master;
                file "/etc/bind/db.upm.local";
                allow-transfer { 192.168.1.102; };
        };
```

File indique le fichier dans lequel sera configurée votre zone.

Allow-transfer indique le serveur qui pourra recevoir vos mises à jour (serveur esclave ou secondaire)

Vous pouvez vérifier la syntaxe du fichier **named.conf** grâce à la commande

named-checkconf /etc/bind/named.conf

c. Configuration de la zone du serveur maître

On édite donc le fichier **/etc/bind/db.upm.local**

Afin d'avoir une configuration "basique", vous pouvez faire une copie de **/etc/bind/db.local.**

cp /etc/bind/db.local /etc/bind/db.upm.local

nano /etc/bind/db.upm.local

Dans ce fichier de zone, nous allons indiquer les enregistrements DNS. Il en existe de plusieurs types :

A : c'est le type le plus courant, il fait correspondre un nom d'hôte à une adresse IPv4 ;

AAAA : fait correspondre un nom d'hôte à une adresse IPv6 ;

CNAME : permet de créer un alias pointant sur un autre nom d'hôte ;

NS : définit le ou les serveurs DNS du domaine ;

MX : définit le ou les serveurs de mail du domaine ;

PTR : fait correspond une IP à un nom d'hôte. Il n'est utilisé que dans le cas d'une zone inverse, que nous verrons plus loin ;

SOA : donne les infos de la zone, comme le serveur DNS principal, l'adresse mail de l'administrateur de la zone, le numéro de série de la zone et des durées que nous détaillerons après.

```
 GNU nano 2.5.3                File: /etc/bind/db.upm.local

$TTL    604800 ;

$ORIGIN upm.local.

@       IN      SOA     ns1.upm.local. admin.upm.local. (
                        2018030101      ; Serial
                         3600           ; Refresh
                          3000          ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL

@       IN      NS      ns1.upm.local.
@       IN      NS      ns2
@       IN      MX      10 mx1
@       IN      MX      20 mx2
ns1     IN      A       192.168.1.101
ns2     IN      A       192.168.1.102
mx1     IN      A       192.168.1.103
mx2     IN      A       192.168.1.104
cours   IN      A       192.168.1.105
www     IN      A       192.168.1.106
blog    IN      CNAME   www
```

La première info est le **TTL** (Time to Live). Quand quelqu'un va interroger votre serveur DNS pour obtenir des informations, ces informations vont être stockées en cache chez cette personne (dans la mémoire cache DNS, pour éviter qu'il vienne réinterroger le serveur DNS de nombreuses fois s'il a de nouveau besoin d'une information).

Ce **TTL** est la durée pendant laquelle les informations sont conservées en cache. Ce délai passé, une nouvelle demande devra être faite au serveur. Le TTL est défini ici sur 1 semaine (604800 secondes = 1 semaine).

En fonction de la fréquence de vos mises à jour, vous pouvez décider de baisser cette valeur pour que vos clients aient leurs informations à jour.

La deuxième info est la variable **ORIGIN**. Celle-ci est optionnelle. Vous voyez les petits @? Ces @ prennent la valeur de la variable ORIGIN. En l'absence de

variable ils prendront la valeur du nom de votre zone défini dans le fichier named.conf (upm.local ici).

Voilà, notre zone est maintenant configurée sur notre serveur master. Vous devez redémarrer BIND pour que les changements soient prises en compte :

/etc/init.d/bind9 restart

```
root@ubuntu:~# /etc/init.d/bind9 restart
[ ok ] Restarting bind9 (via systemctl): bind9.service.
```

Et enfin assigner une adresse IP fixe au serveur DNS maître

nano /etc/network/interfaces

```
# The primary network interface
auto ens33
iface ens33 inet static
        address 192.168.1.101
        netmask 255.255.255.0
        network 192.168.1.0
        gateway 192.168.1.1
        dns-nameservers 192.168.1.101 192.168.1.102
```

Redémarrer l'interface réseau :

```
root@ubuntu:~# ip addr flush ens33
root@ubuntu:~# systemctl restart networking
```

d. Configuration du serveur esclave

Nous avons prévu deux serveurs dans notre architecture. Celui que nous venons de configurer est le master ; celui que nous allons faire sera l'esclave.

Les modifications DNS se font sur le master, et celui-ci enverra des notifications aux serveur(s) esclave(s) (il peut y en avoir plusieurs) pour que leurs zones soient mises à jour.

La configuration du serveur esclave est donc relativement simple, tout se passe dans le fichier **named.conf**. Il n'y a pas de fichier de zone à configurer étant donné que celui-ci sera reçu du serveur maître.

On commence par installer Bind comme pour le master et on édite le fichier **/etc/bind/named.conf**

```
zone "upm.local" {
                type slave;
                file "/var/cache/bind/db.upm.local";
                masters { 192.168.1.101;};
        };
```

Pui assigner une adresse IP fix au serveur esclave

```
# The primary network interface
auto ens33
iface ens33 inet static
        address 192.168.1.102
        netmask 255.255.255.0
        network 192.168.1.0
        gateway 192.168.1.1
        dns-nameservers 192.168.1.101 192.168.1.102_
```

Redémarrer l'interface réseau :

```
root@ubuntu:~# ip addr flush ens33
root@ubuntu:~# systemctl restart networking
```

Affecter les deux machines des serveurs DNS au même Virtual Network et redémarrer les serveurs avec la commande Reboot.

Vérifier si les adresses IP fix ont été assignées.

e. Résolution inverse

Votre serveur DNS doit pouvoir résoudre une adresse IP en un nom d'hôte. C'est ce que nous allons faire ici.

Pour l'instant, nous avons vu le protocole DNS comme un moyen de résoudre un nom d'hôte en une adresse IP. Nous avons parlé des enregistrements de type **PTR** et vous savez donc que DNS permet aussi de faire le travail inverse. C'est une résolution inverse.

Retournons dans notre fichier **named.conf** afin d'ajouter cette zone inverse. Nous allons déclarer la zone inverse de notre adressage IP, ici c'est 192.168.1.0/24.

Alors qu'une zone "normale" se déclare de façon plutôt logique, une zone inverse doit respecter une certaine forme concernant le nom de la zone.

Ajouter la zone inverse en dessous de la zone normale comme suit :

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";


zone "upm.local" {
                type master;
                file "/etc/bind/db.upm.local";
                allow-transfer { 192.168.1.102; };
        };

zone "1.168.192.in-addr.arpa."{
                type master;
                file "/etc/bind/db.192.168.1";
        };_
```

On crée ensuite le fichier de zone **db.192.168.1** et le remplir comme suit :

```
$TTL    604800 ;

$ORIGIN 1.168.192.in-addr.arpa.

@       IN      SOA     ns1.upm.local. admin.upm.local. (
                        2018030101      ; Serial
                         3600           ; Refresh
                          3000          ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL

@       IN      NS      ns1.upm.local.
@       IN      NS      ns2.upm.local.
101     IN      PTR     ns1.upm.local.
102     IN      PTR     ns2.upm.local.
103     IN      PTR     mx1.upm.local.
104     IN      PTR     mx2.upm.local.
105     IN      PTR     cours.upm.local.
106     IN      PTR     www.upm.local.
```

Les points auxquels il faut faire attention :

- une zone inverse ne contient que des enregistrements de type NS ou PTR ;
- dans notre zone "normale", blog redirigeait vers www, mais là une adresse IP ne peut pointer que vers un seul hôte ;
- la variable ORIGIN a changée ! Il faut donc penser à utiliser le **FQDN** de nos hôtes à chaque fois.

Voilà, notre zone inverse est maintenant configurée sur notre serveur master. Vous devez redémarrer BIND pour que les changements soient prises en compte :

/etc/init.d/bind9 restart

```
root@ubuntu:~# /etc/init.d/bind9 restart
[ ok ] Restarting bind9 (via systemctl): bind9.service.
```

f. Vérification des serveurs DNS

On va vérifier le fonctionnement de notre zone maintenant.

Il existe plusieurs commandes pour faire des interrogations DNS. La commande la plus utilisée est **host** mais **dig** fournit plus d'informations et permet un diagnostic plus précis en cas de problème.

Vérifiez d'abord le serveur DNS utilisé par votre machine. Comme cette machine est elle-même un serveur DNS, elle va devoir s'interroger elle-même.

Le programme qui fait toutes les résolutions DNS dans le serveur DNS pour une machine cliente ou un programme s'appelle le **resolver**. Ainsi, chaque programme qui a besoin de faire une résolution DNS s'adresse au **resolver**.

Son fichier de configuration se trouve dans /etc/resolv.conf qui doit au moins contenir l'adresse d'un serveur DNS à interroger

Vérifier le contenu de /etc/resolv.conf dans la machine DNS maitre.

```
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 192.168.1.101
nameserver 192.168.1.102
```

Vérifier aussi le contenu du **/etc/resolv.conf** sur les autres machines.

On peut maintenant commencer nos tests :

Démarrer la machine client.

Vérifier si la machine client a comme serveur DNS primaire 192.168.1.101 et comme DNS secondaire 192.168.1.102

La machine cliente devra avoir cette configuration. Normalement, le serveur DHCP configuré dans le TP1 est paramétré afin de fournir ces informations. Si ce n'est pas le cas, vérifier si vous aves bien configurer votre serveur DHCP.

Nous allons donc utiliser la commande host qui permet de faire une interrogation DNS.

Sa syntaxe est la suivante :

host -t type nom_a_chercher IP_serveur

On peut ainsi indiquer le type de la requête (NS, A, MX, CNAME, etc.), le nom à interroger, ainsi que l'adresse IP du serveur que l'on peut préciser.

Par exemple, si l'on cherche l'adresse des serveurs DNS du domaine upm.local :

host -t ns upm.local

```
root@ubuntu:~# host -t ns upm.local
upm.local name server ns1.upm.local.
upm.local name server ns2.upm.local.
root@ubuntu:~# _
```

La commande dig upm.local

```
root@ubuntu:~# dig upm.local

; <<>> DiG 9.10.3-P4-Ubuntu <<>> upm.local
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19438
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;upm.local.                     IN      A

;; AUTHORITY SECTION:
upm.local.              604800  IN      SOA     ns1.upm.local. admin.upm.local. 2018030101 3600 3000
 2419200 604800
```

On peut aussi utiliser la commande ping :

```
root@ubuntu:~# ping ns1.upm.local
PING ns1.upm.local (192.168.1.101) 56(84) bytes of data.
64 bytes from ns1.upm.local (192.168.1.101): icmp_seq=1 ttl=64 time=0.144 ms
64 bytes from ns1.upm.local (192.168.1.101): icmp_seq=2 ttl=64 time=0.656 ms
64 bytes from ns1.upm.local (192.168.1.101): icmp_seq=3 ttl=64 time=0.677 ms
```

```
root@ubuntu:~# ping www.upm.local
PING www.upm.local (192.168.1.106) 56(84) bytes of data.
From 192.168.1.150 icmp_seq=1 Destination Host Unreachable
From 192.168.1.150 icmp_seq=2 Destination Host Unreachable
From 192.168.1.150 icmp_seq=3 Destination Host Unreachable
From 192.168.1.150 icmp_seq=4 Destination Host Unreachable
```

Et enfin, vérifier la résolution inverse :

```
root@ubuntu:~# host 192.168.1.103
103.1.168.192.in-addr.arpa domain name pointer mx1.upm.local.
root@ubuntu:~# _
```