Configuration du serveur DNS principal et secondaire

Contexte

Configuration DHCP

- **Adresse réseau** : `192.168.49.0/24`
- **Passerelle** : `192.168.49.1`
- **Distribution des adresses** : `192.168.49.128-254`
- **Adresse broadcast** : `192.168.49.255`

Machines

- **Client Ubuntu desktop 20.04**
    - **Configuration IP** : DHCP avec DNS manuel
- **Client Ubuntu server 20.04**
    - **Configuration IP** : Statique (`192.168.49.133`) avec DNS manuel
    - **Domaine** : `ubuntu.uncle.esgi`
- **Serveur DNS primaire Debian 10**
    - **Configuration IP** : Statique (`192.168.49.254`) avec DNS en localhost
    - **Domaine** : `ns1.uncle.esgi`
    - **Alias (CNAME)** : `server.uncle.esgi`
- **Serveur DNS secondaireDebian 10**
    - **Configuration IP** : Statique (`192.168.49.253`) avec DNS sur le serveur principal
    - **Domaine** : `ns2.uncle.esgi`
- **Zone DNS** : `uncle.esgi`
- **Alias externes** :
    - `gogole.uncle.esgi` ⇒ `www.google.com`
    - `school.uncle.esgi` ⇒ `www.esgi.fr`

1. Configuration des serveurs DNS

1.1 Serveur Primaire

Installation des ressources

Avant toute chose, nous allons installer les ressources dont nous avons besoins :

```
apt install bind9 bind9utils bind9-doc dnsutils resolvconf -y
```

Configuration réseau

Notre serveur DNS doit avoir une configuration IP statique afin d'être configuré sur les postes clients Nous allons donc modifier cela dans le fichier `/etc/network/interfaces` en se référant au contexte donné :

```
auto ens33
iface ens33 inet static
        address 192.168.49.254
        netmask 255.255.255.0
        gateway 192.168.49.1
        dns-nameservers 127.0.0.1
```

Nous redémarrons ensuite le service avec la commande :

```
systemctl restart networking
```

Nous vérifions que tout s'est bien passé en regardant la configuration IP avec la commande `ip a` et en vérifiant que le fichier `/etc/resolv.conf` contient bien uniquement la ligne `nameserver 127.0.0.1`

```
user@dns-server:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:1e:06:45 brd ff:ff:ff:ff:ff:ff
    inet 192.168.49.254/24 brd 192.168.49.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe1e:645/64 scope link
       valid_lft forever preferred_lft forever
user@dns-server:~$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 127.0.0.1
```

Test du serveur cache

A partir de ce moment là, notre serveur possède son propre serveur DNS local, nous allons vérifier que celui-ci fonctionne avec la commande `dig www.debian.org`, celle-ci devrait être capable de résoudre le nom de domaine, nous devons nous en assurer avant d'aller plus loin :

```
root@dns-server:/home/server dig www.debian.org
; <<>> DiG 9.11.5-P4-5.1+deb10u1-Raspbian <<>> www.debian.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64130
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.debian.org.                        IN      A

;; ANSWER SECTION:
www.debian.org.         168     IN      A       130.89.148.77

;; AUTHORITY SECTION:
www.debian.org.         300     IN      NS      geo1.debian.org
www.debian.org.         300     IN      NS      geo2.debian.org
www.debian.org.         300     IN      NS      geo3.debian.org

;; ADDITIONAL SECTION:
geo1.debian.org.        28799   IN      A       82.195.75.105
geo2.debian.org.        28799   IN      A       209.87.16.31
geo1.debian.org.        28799   IN      AAAA    2001:41b8:202:deb:1a1a:0:52c3:4b69

;; Query time: 568 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Jun 29 18:29:49 BST 2020
;; MSG SIZE  rcvd: 204
```

Notre serveur DNS bind9 a réussi à résoudre cette adresse, nous pouvons donc mettre en place notre serveur primaire.

Mise en place du serveur primaire

Définition de la zone DNS et reverse DNS

Nous allons maintenant mettre en place notre zone DNS et reverse DNS selon le contexte définit. Pour cela nous allons modifier le fichier `/etc/bind/named.conf.local` et y ajouter ces lignes :

```
zone "uncle.esgi" {
     type master;
     file "/etc/bind/db.uncle.esgi";
};

zone "49.168.192.in-addr.arpa" {
     type master;
     file "/etc/bind/db.49.168.192.in-addr.arpa";
};
```

Une fois cela fait, il faudra créé les fichiers indiqués dans la conf et les remplir.

Configuration de la zone DNS et reverse DNS

Remplissons tout d'abord notre fichier `/etc/bind/db.uncle.esgi` contenant la définition de notre zone DNS ainsi que les domaines, les alias, etc.

```
$TTL 10800
$ORIGIN uncle.esgi.
@       IN SOA ns1.uncle.esgi. admin.uncle.esgi. (
        20200628; Serial
        3h; Refresh
        1h; Retry
        1w; Expire
        1h); Minimum
@       IN NS    ns1.uncle.esgi.
ubuntu  IN A     192.168.49.133
ns1     IN A     192.168.49.254
ns2     IN A     192.168.49.253
primary-server  IN CNAME        ns1
gogole          IN CNAME        www.google.com.
school          IN CNAME        www.esgi.fr.
```

- **@** : remplace le nom de la zone uncle.esgi
- **ns1.uncle.esgi.** : serveur primaire de la zone
- **admin.uncle.esgi.** : adresse mail de l'administrateur (l'@ est remplacée par un .)
- **Serial** : indique le numéro de version du serveur : il doit être incrémenté à chaque modification, ce qui indique aux autres serveurs que leurs données doivent être mises à jour.
- **Refresh** : indique le temps entre deux mises à jour des serveurs secondaires par rapport au serveur primaire.
- **Retry** : indique le temps entre 2 tentatives de Refresh s'il y a échec.
- **Expire** : indique le temps au bout duquel un serveur secondaire ne fait plus autorité s'il n'a pas réussi à contacter le serveur primaire.
- **Minimum** : indique le temps durant lequel une réponse négative doit être conservée en cache.

Il faut ensuite définir le reverse DNS pour nos noms de domaine à l'aide du fichier `/etc/bind/db.49.168.192.in-addr.arpa` :

```
$TTL 10800
$ORIGIN 49.168.192.in-addr.arpa.
@       IN SOA ns1.uncle.esgi. admin.uncle.esgi (
        20200628;
        3h;
        1h;
        1w;
        1h);
@       IN NS    ns1.uncle.esgi.
133     IN PTR ubuntu.uncle.esgi.
```

Une fois cela fait, nous redemarons le service bind9 avec la commande `systemctl restart bind9`

Test de la zone DNS

Si tout s'est bien, la zone DNS et reverse DNS devraient être reconnu par notre serveur. Nous nous en assurons en allant résoudre dans un premier temps l'adresse `ubuntu.uncle.esgi` avec la commande `dig` :

```
user@dns-server:~$ dig ubuntu.uncle.esgi

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> ubuntu.uncle.esgi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37398
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 16fab81d8a5925ff15021fbe5efa5d6242c49ab1ff5ea44b (good)
;; QUESTION SECTION:
;ubuntu.uncle.esgi.             IN      A

;; ANSWER SECTION:
ubuntu.uncle.esgi.      10800   IN      A       192.168.49.133

;; AUTHORITY SECTION:
uncle.esgi.             10800   IN      NS      ns1.uncle.esgi.

;; ADDITIONAL SECTION:
ns1.uncle.esgi.         10800   IN      A       192.168.49.254

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: lun. juin 29 23:30:10 CEST 2020
;; MSG SIZE  rcvd: 158
```

Très bien, le domaine est reconnu !

Testons maintenant la zone reverse dns de ce même domaine `192.168.49.133` avec la commande `dig -x 192.168.49.133 +short` :

```
user@dns-server:~$ dig -x 192.168.49.133 +short
ubuntu.uncle.esgi.
```

La zone de reverse DNS fonctionne également.

1.2 Serveur Secondaire

Mettons en place maintenant le serveur DNS secondaire qui servira à assurer le relais en tant que serveur DNS si le principale tombe en panne. On utilisera une relation "maitre" "esclave"entre notre serveur primaire et secondaire. Ainsi, le serveur secondaire sera la copie confirme du premier, et ne pourra se mettre à jour que si le principal est actif.

Installation des ressources

Nous reinstallons les mêmes outils que sur le premier serveur, à savoir :

```
apt install bind9 bind9utils bind9-doc dnsutils resolvconf -y
```

Configuration réseau

Notre serveur DNS doit avoir une configuration IP statique afin d'être configuré sur les postes clients Nous allons donc modifier cela dans le fichier `/etc/network/interfaces` en se référant au contexte donné :

```
auto ens33
iface ens33 inet static
        address 192.168.49.253
        netmask 255.255.255.0
        gateway 192.168.49.1
        dns-nameservers 192.168.49.254
```

Nous redémarrons ensuite le service avec la commande :

```
systemctl restart networking
```

Nous vérifions que tout s'est bien passé en regardant la configuration IP avec la commande `ip a` et en vérifiant que le fichier `/etc/resolv.conf` contient bien uniquement la ligne `nameserver 127.0.0.1`

```
user@dns-server-2:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:36:81:c4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.49.253/24 brd 192.168.49.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe36:81c4/64 scope link
       valid_lft forever preferred_lft forever
user@dns-server:~$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 192.168.49.254
search uncle.esgi
```

Mise en place du serveur secondaire

Déclaration du serveur secondaire sur le serveur primaire

Il faut ensuite configurer le serveur principal pour qu'il interagisse avec le serveur secondaire. Tout d'abord, il faut lui dire de notifier le secondaire quand un des fichiers de zone est modifié. Pour cela, on ajoute la directive notify yes pour chaque zone dont on veut qu'elle soit reproduite. On ajoute ensuite la liste des serveurs secondaires autorisés à effectuer un transfert de zone avec la directive allow-transfer :

Serveur DNS principal `/etc/bind/named.conf.local`

```
zone "uncle.esgi" {
     type master;
     file "/etc/bind/db.uncle.esgi";
     notity yes;
     allow-transfer { 192.168.49.253; };
};

zone "49.168.192.in-addr.arpa" {
     type master;
     file "/etc/bind/db.49.168.192.in-addr.arpa";
     notity yes;
     allow-transfer { 192.168.49.253; };
};
```

Nous déclarons également notre nouveau serveur dans la zone DNS et reverse DNS :

Serveur DNS principal `/etc/bind/db.uncle.esgi`

```
$TTL 10800
$ORIGIN uncle.esgi.
@       IN SOA ns1.uncle.esgi. admin.uncle.esgi. (
        20200628;
        3h;
        1h;
        1w;
        1h);
@       IN NS   ns1.uncle.esgi.
@       IN NS   ns2.uncle.esgi.
ubuntu  IN A    192.168.49.133
ns1     IN A    192.168.49.254
ns2     IN A    192.168.49.253
primary-server  IN CNAME        ns1
gogole          IN CNAME        www.google.com.
school          IN CNAME        www.esgi.fr.
```

Serveur dn principal `/etc/bind/db.49.168.192.in-addr.arpa`

```
$TTL 10800
$ORIGIN 49.168.192.in-addr.arpa.
@       IN SOA ns1.uncle.esgi. admin.uncle.esgi (
        20200628;
        3h;
        1h;
        1w;
        1h);
@       IN NS   ns1.uncle.esgi.
@       IN NS   ns2.uncle.esgi.
133     IN PTR ubuntu.uncle.esgi.
```

Une fois cela fait, nous redemarons le service bind9 avec la commande `systemctl restart bind9`

Si tout s'est bien, le serveur secondaire devraient être reconnu par la zone DNS et reverse DNS. Nous nous en assurons en allant résoudre dans un premier temps l'adresse `ns2.uncle.esgi` avec la commande `dig` :

```
dig ns2.uncle.esgi

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> ns2.uncle.esgi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22898
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 954560d971df8139d8d9adc55efa63fd94049bcab5a7c36f (good)
;; QUESTION SECTION:
;ns2.uncle.esgi.                        IN      A

;; ANSWER SECTION:
ns2.uncle.esgi.         10800   IN      A       192.168.49.253

;; AUTHORITY SECTION:
uncle.esgi.             10800   IN      NS      ns2.uncle.esgi.
uncle.esgi.             10800   IN      NS      ns1.uncle.esgi.

;; ADDITIONAL SECTION:
ns1.uncle.esgi.         10800   IN      A       192.168.49.254

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: lun. juin 29 23:58:21 CEST 2020
;; MSG SIZE  rcvd: 135
```

Définition de la zone DNS et reverse DNS

Retournons maintenant sur notre serveur secondaire, il faut déclarer ce serveur secondaire dans le fichier `/etc/bind/named.conf.local` de la machine qui l'héberge :

```
zone "uncle.esgi" {
     type slave;
     file "/var/lib/bind/db.uncle.esgi";
     notity yes;
     masters { 192.168.49.254; };
};

zone "49.168.192.in-addr.arpa" {
     type master;
     file "/var/lib/bind/db.49.168.192.in-addr.arpa";
     notity yes;
     masters { 192.168.49.254; };
};
```

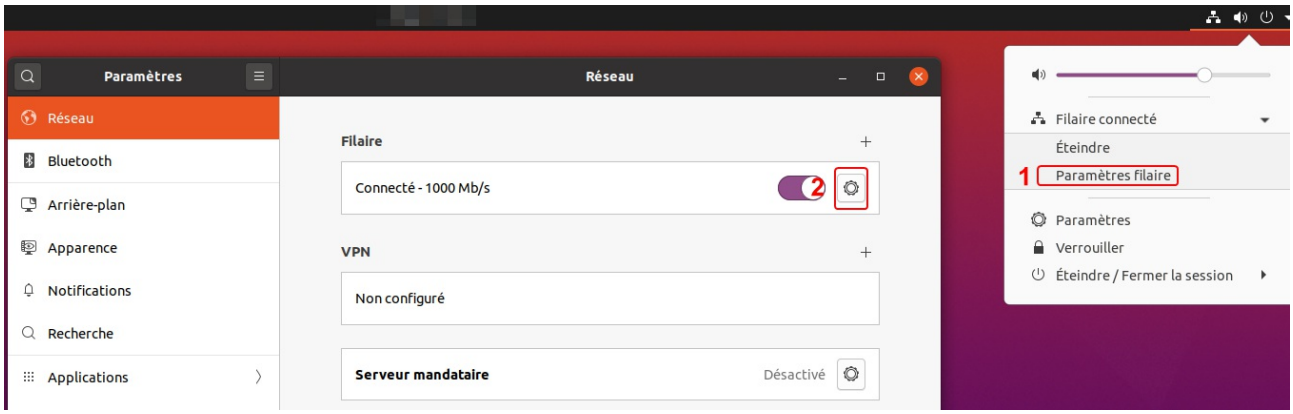
Pour avoir les droits nécéssaires, nos fichiers téléchargé depuis le serveur principal seront stockés dans le repertoire `/var/lib/bind/`

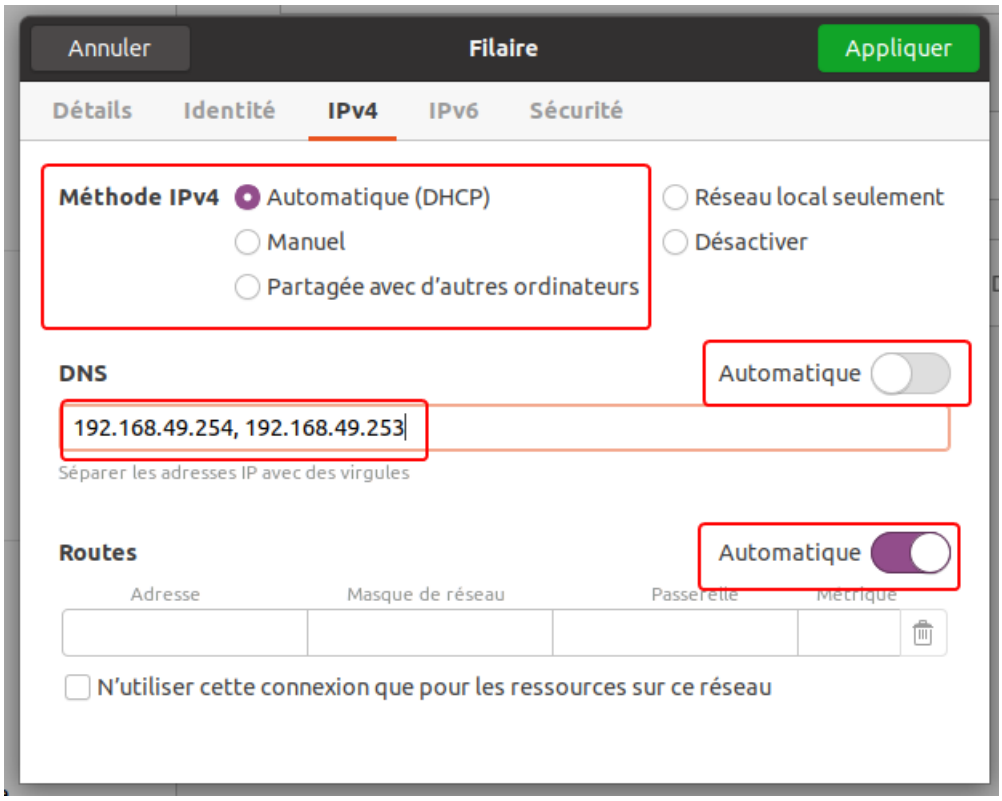
1.3 Configuration du client Ubuntu desktop 20.04

Comme indiqué dans le Contexte, le client ubuntu desktop est en DHCP avec le dns configuré manuellement.

Configuration réseau

Pour cela, nous allons dans les paramètres réseaux :





Test du serveur DNS

Nous testons la zone DNS et reverse DNS grace à la commande `dig` :

```
dig primary-server.uncle.esgi

; <<>> DiG 9.16.1-Ubuntu <<>> primary-server.uncle.esgi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27593
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;primary-server.uncle.esgi. IN A

;; ANSWER SECTION:
primary-server.uncle.esgi. 7092 IN CNAME ns1.uncle.esgi.
ns1.uncle.esgi.  7092 IN A 192.168.49.254

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: lun. juin 29 15:19:09 PDT 2020
;; MSG SIZE  rcvd: 88
```

Notre zone dns est correct, essayons maintenant la zone reverse DNS :

```
user@dns-server:~$ dig -x 192.168.49.133 +short
ubuntu.uncle.esgi.
```

1.4 Configuration du client Ubuntu server 20.04

Comme indiqué dans le Contexte, le client ubuntu serveur est en ip statique avec le dns configuré manuellement.

Configuration réseau

Pour cela, nous allons utilisons l'utilitaire réseau installé par défaut sur ubuntu server, à savoir netplan. Pour cela, il faut modifier le fichier `/etc/netplan/00-installer-config.yaml` :

```
network:
  ethernets:
    ens33:
      addresses: [192.168.49.133/24]
      gateway4: 192.168.49.1
      nameservers:
        addresses: [192.168.49.254,192.168.49.253]
  version: 2
```

Pour appliquer notre configuration, nous lançon la ocmmande `netplan apply`. Si aucune erreur n'a été detecté, nous pouvons vérifier que notre adresse ip est bien défini :

```
user@dnsclient:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:09:70:42 brd ff:ff:ff:ff:ff:ff
    inet 192.168.49.133/24 brd 192.168.49.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe09:7042/64 scope link
       valid_lft forever preferred_lft forever
```

Test du serveur DNS

Nous testons la zone DNS et reverse DNS grace à la commande `dig` :

```
user@dnsclient:~$ dig ns2.uncle.esgi

; <<>> DiG 9.16.1-Ubuntu <<>> ns2.uncle.esgi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33076
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;ns2.uncle.esgi.                        IN      A

;; ANSWER SECTION:
ns2.uncle.esgi.         10800   IN      A       192.168.49.253

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Mon Jun 29 22:29:55 UTC 2020
;; MSG SIZE  rcvd: 59
```

Notre zone dns est correct, essayons maintenant la zone reverse DNS :

```
user@dnsclient:~$ dig -x 192.168.49.133 +short
ubuntu.uncle.esgi.
```

2. Test de panne du serveur DNS primaire

Lorsque nos deux serveurs sont actifs, si nous regardons le status de résolution dns sur notre client **ubuntu server 20.04**, nous observons cela :

```
systemd-resolve --status
Global
       LLMNR setting: no
MulticastDNS setting: no
  DNSOverTLS setting: no
      DNSSEC setting: no
    DNSSEC supported: no
          DNSSEC NTA: 10.in-addr.arpa
                      ...
                      corp
                      d.f.ip6.arpa
                      home
                      internal
                      intranet
                      lan
                      local
                      private
                      test

Link 2 (ens33)
      Current Scopes: DNS
DefaultRoute setting: yes
       LLMNR setting: yes
MulticastDNS setting: no
  DNSOverTLS setting: no
      DNSSEC setting: no
    DNSSEC supported: no
  Current DNS Server: 192.168.49.254
         DNS Servers: 192.168.49.254
                      192.168.49.253
```

Nous constatons que le **Current DNS server** est notre serveur DNS principal

Si j'éteint mon serveur principal, lorsque je demarre **ubuntu server 20.04**, nous obersvons cela :

```
systemd-resolve --status
Global
       LLMNR setting: no
MulticastDNS setting: no
  DNSOverTLS setting: no
      DNSSEC setting: no
    DNSSEC supported: no
          DNSSEC NTA: 10.in-addr.arpa
                      ...
                      corp
                      d.f.ip6.arpa
                      home
                      internal
                      intranet
                      lan
                      local
                      private
                      test

Link 2 (ens33)
      Current Scopes: DNS
DefaultRoute setting: yes
       LLMNR setting: yes
MulticastDNS setting: no
  DNSOverTLS setting: no
      DNSSEC setting: no
    DNSSEC supported: no
  Current DNS Server: 192.168.49.253
         DNS Servers: 192.168.49.254
                      192.168.49.253
```

Nous constatons que le **Current DNS server** est désormais notre serveur DNS secondaire

faisons un essai avec la commande `dig`, puis en essayant un `ping` un de nos domaine :

```
user@dnsclient:~$ dig www.google.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56632
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         243     IN      A       216.58.213.132

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Mon Jun 29 22:39:05 UTC 2020
;; MSG SIZE  rcvd: 59
```

```
user@dnsclient:~$ ping ns2.uncle.esgi
PING ns2.uncle.esgi (192.168.49.253) 56(84) bytes of data.
64 bytes from 192.168.49.253 (192.168.49.253): icmp_seq=1 ttl=64 time=0.088 ms
64 bytes from 192.168.49.253 (192.168.49.253): icmp_seq=2 ttl=64 time=0.173 ms
64 bytes from 192.168.49.253 (192.168.49.253): icmp_seq=3 ttl=64 time=0.151 ms
64 bytes from 192.168.49.253 (192.168.49.253): icmp_seq=4 ttl=64 time=0.161 ms
--- ns2.uncle.esgi ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3035ms
rtt min/avg/max/mdev = 0.088/0.143/0.173/0.032 ms
```