


```
forwarders {  
    8.8.8.8;  
    8.8.4.4;  
};  
  
allow-query { any; };
```

Puis, nous modifions notre fichier `/etc/bind/named.conf.local` pour avoir le contenu suivant:

```
zone "efrei.fr" IN {  
    type master;  
    file "/etc/bind/forward.efrei.fr.db";  
    allow-update { none; };  
};  
  
zone "29.16.172.in-addr.arpa" IN {  
    type master;  
    file "/etc/bind/reverse.efrei.fr.db";  
    allow-update { none; };  
};
```



```
@ IN A 192.168.1.137
@ IN AAAA ::1
;
ubuntu.efrei.fr. IN A 192.168.1.28
```

```
  
vim forward.efrei.fr.db  
  
1 ;  
2 ; BIND data file for local loopback interface  
3 ;  
4 $TTL      604800  
5 @         IN      SOA     efrei.fr. root.efrei.fr. (  
6             2           ; Serial  
7             604800       ; Refresh  
8             86400        ; Retry  
9             2419200      ; Expire  
10            604800 )     ; Negative Cache TTL  
11 ;  
12 @         IN      NS      efrei.fr.  
13 @         IN      A       192.168.1.137  
14 @         IN      AAAA    ::1  
15 ;  
16 ubuntu.efrei.fr                IN      A          192.168.1.137  
  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
forward.efrei.fr.db [+]                               16,50              All  
"forward.efrei.fr.db" 14L, 270B  
-- INSERT --
```

Puis, nous opérons à l'identique pour le fichier `reverse`.

```
$ cp /etc/bind/db.local /etc/bind/reverse.efrei.fr.db
```

Nous modifions son contenu pour avoir le suivant:

```
;
; BIND data file for local loopback interface
;
$TTL 604800
@ IN SOA efrei.fr. root.efrei.fr. (
    1      ; Serial
    604800 ; Refresh
```

```

        86400      ; Retry
        2419200   ; Expire
        604800 ) ; Negative Cache TTL
;
@      IN  NS   efrei.fr.
efrei.fr      IN  A  192.168.1.137
137      IN  PTR efrei.fr.
28      IN  PTR  ubuntu.efrei.fr.

```

Puis, nous testons à l'aide des 2 commandes suivantes :

```

$ named-checkzone efrei.fr forward.efrei.fr.db

$ named-checkzone 29.16.172.in-addr.arpa reverse.efrei.fr.db

```



```

root@tpnagios-2:/etc/bind

@UBUNTU /etc/bind 12:07:38
$ named-checkzone 29.16.172.in-addr.arpa reverse.efrei.fr.db
zone 29.16.172.in-addr.arpa/IN: loaded serial 1
OK

@UBUNTU /etc/bind 12:07:39
$ named-checkzone efrei.fr forward.efrei.fr.db
zone efrei.fr/IN: loaded serial 2
OK

@UBUNTU /etc/bind 12:07:50
$

```

Nous redémarrons le service.

```

$ systemctl restart named

```

Nous pouvons constater les adresses de nos DNS en regardant le contenu du fichier `/etc/resolv.conf`.

```
root@tpnagios-2:/etc/bind

@UBUNTU /etc/bind 12:12:46
$ cat /etc/resolv.conf
# --- BEGIN PVE ---
search dk
nameserver 192.168.1.137
nameserver 192.168.1.1
nameserver 8.8.8.8
# --- END PVE ---

@UBUNTU /etc/bind 12:12:52
$
```

Nous pouvons donc maintenant vérifier la bonne résolution de `ubuntu.efrei.fr` par la commande `dig ubuntu.efrei.fr`.

```
root@tpnagios-2:/etc/bind

@UBUNTU /etc/bind 12:39:21
$ dig ubuntu.efrei.fr

; <<>> DiG 9.18.18-0ubuntu0.22.04.2-Ubuntu <<>> ubuntu.efrei.fr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19752
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: ec46e9fd41ce09bb01000000662265fd7d5fe496d2a90cdb (good)
;; QUESTION SECTION:
;ubuntu.efrei.fr.                IN      A

;; ANSWER SECTION:
ubuntu.efrei.fr.                604800  IN      A      192.168.1.28

;; Query time: 0 msec
;; SERVER: 192.168.1.137#53(192.168.1.137) (UDP)
;; WHEN: Fri Apr 19 12:39:25 UTC 2024
;; MSG SIZE rcvd: 88
```

```
root@tpnagios-2:/etc/bind
@UBUNTU /etc/bind 12:39:25
$ dig -x 192.168.1.28

; <<>> DiG 9.18.18-0ubuntu0.22.04.2-Ubuntu <<>> -x 192.168.1.28
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21937
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 4eefa4639dd5e560010000000662266002108428fd1404035 (good)
;; QUESTION SECTION:
;28.1.168.192.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
28.1.168.192.in-addr.arpa. 604800 IN      PTR      ubuntu.efrei.fr.

;; Query time: 0 msec
;; SERVER: 192.168.1.137#53(192.168.1.137) (UDP)
;; WHEN: Fri Apr 19 12:39:28 UTC 2024
;; MSG SIZE rcvd: 111
```

```
root@tpnagios-2:/etc/bind

@UBUNTU /etc/bind 12:39:42
$ nslookup efrei.fr
Server:      192.168.1.137
Address:     192.168.1.137#53

Name:   efrei.fr
Address: 192.168.1.137
Name:   efrei.fr
Address: ::1

@UBUNTU /etc/bind 12:40:12
$ nslookup 192.168.1.137
137.1.168.192.in-addr.arpa      name = efrei.fr.

@UBUNTU /etc/bind 12:40:16
$ nslookup 192.168.1.137

@UBUNTU /etc/bind 12:40:22
$ dig ubuntu.efrei.fr

@UBUNTU /etc/bind 12:40:23
$ nslookup ubuntu.efrei.fr
Server:      192.168.1.137
Address:     192.168.1.137#53

Name:   ubuntu.efrei.fr
Address: 192.168.1.28

@UBUNTU /etc/bind 12:40:36
$
```

Nous pouvons constater que tout réssoud correctement.

Kerberos

Nous configurons maintenant une authentification SSH avec Kerberos.

Pour un souci de simplicité au niveau des dépendances, nous allons passer par une image Docker custom.

Nous commençons donc par créer un `Dockerfile` ayant le contenu suivant:

```
FROM ubuntu:20.04

RUN apt-get update && \
    apt-get install -y krb5-kdc krb5-admin-server && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

COPY krb5.conf /etc/
COPY kdc.conf /etc/krb5kdc/
COPY kadm5.acl /etc/krb5kdc/
```



```
RUN krb5_newrealm <<EOF
secret123
secret123
EOF

EXPOSE 8888

CMD ["krb5kdc", "-n"]
```

Puis, nous créons 3 fichiers de configurations pour Kerberos.

krb5.conf :

```
[libdefaults]
    default_realm = EFREI.FR
    dns_lookup_realm = false
    dns_lookup_kdc = false

[realms]
    EFREI.FR = {
        kdc = 192.168.1.137:8888
        admin_server = 192.168.1.137
    }

[domain_realm]
    .efrei.fr = EFREI.FR
    efrei.fr = EFREI.FR

[logging]
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kadmind.log
    default = FILE:/var/log/krb5libs.log
```

kdc.conf :

```
[kdcdefaults]
    kdc_ports = 8888

[realms]
    EFREI.FR = {
        database_module = DB2
        acl_file = /etc/krb5kdc/kadm5.acl
        dict_file = /usr/share/dict/words
        admin_keytab = /etc/krb5kdc/kadm5.keytab
        supported_enctypes = aes256-cts:normal aes128-cts:normal
    }

[logging]
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kadmind.log
```

kadm5.acl :

```
*/admin@EFREI.FR *
```

Nous construisons notre image Docker avec la commande suivante:

```
$ docker build -t kerberos-server .
```

Puis nous lançons notre conteneur de serveur.

```
docker run --name kerberos-server -p 8888:8888 --network="host" -d kerberos-server
```

Pour nous connecter en tant que client, nous allons également faire appel à un conteneur Docker (problèmes de dépendances sur les instances d'Ubuntu ne rendant pas possible l'installation des paquets).

Nous créons donc un `Dockerfile` pour créer une image custom de notre client kerberos.

```
FROM ubuntu:20.04

RUN apt-get update && \
    apt-get install -y krb5-user ssh && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

COPY krb5.conf /etc/

CMD ["bash"]
```

Le fichier `krb5.conf` est rigoureusement identique à celui mentionné précédemment.

Puis, une fois l'image construite, nous lançons notre conteneur client et nous connectons en shell dessus à l'aide de la commande suivante:

```
$ docker run -it --network="host" kerberos-client
```

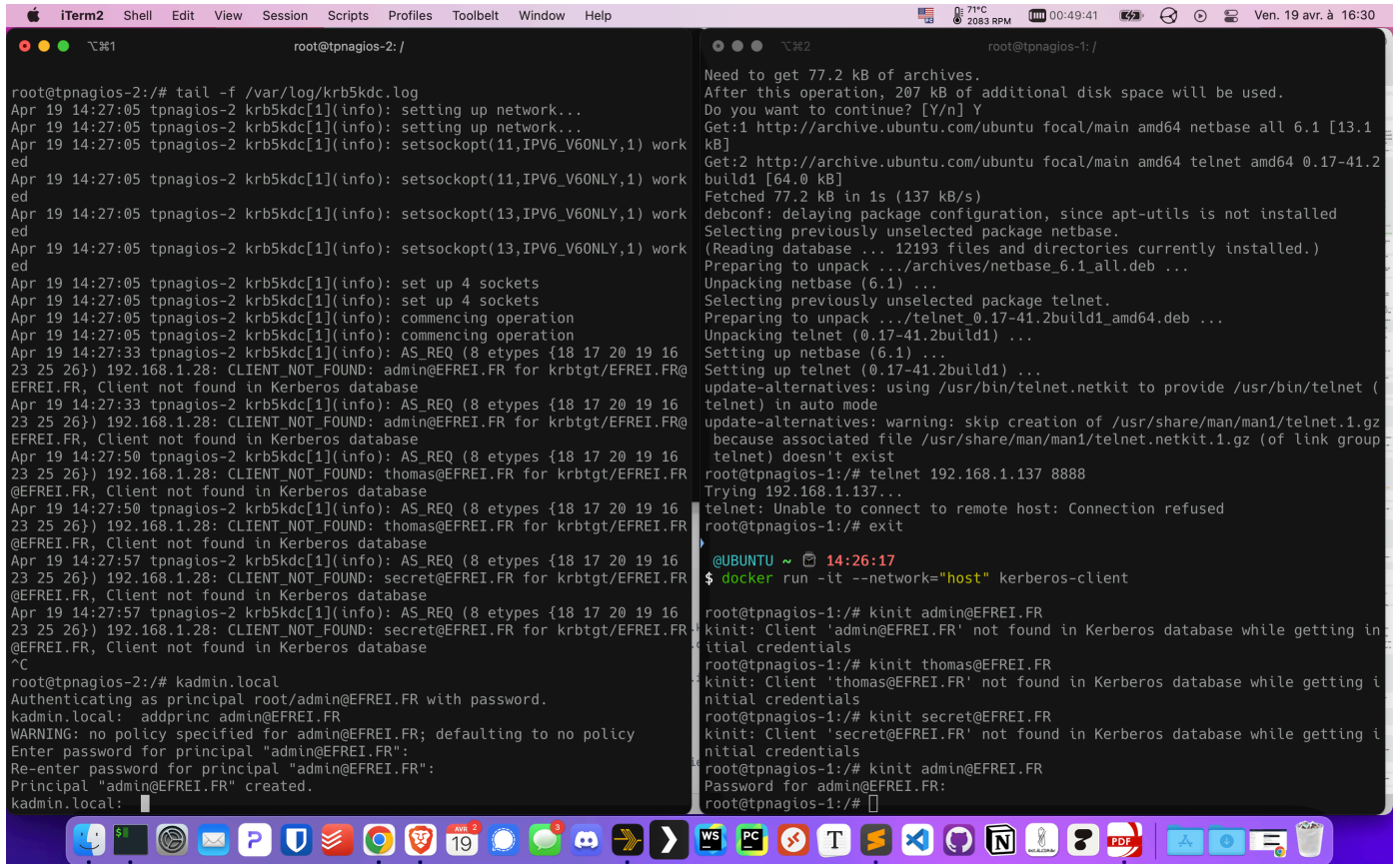
Sur le serveur, nous nous connectons au shell de notre conteneur :

```
$ docker exec -it kerberos-server /bin/bash
```

Puis, nous ajoutons un utilisateur à la base de données :

```
# Connexion à la console
$ kadmin.local

# Ajout de l'utilisateur thomas
$ kadmin.local: addprinc thomas@EFREI.FR
```



Nous à avons ici, à gauche les logs du serveur, et à droite le `kinit admin@EFREI.FR` du client.

Nous utilisons la commande `klist` pour vérifier que nous avons bien notre ticket.



```
root@tpnagios-1:/# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: admin@EFREI.FR

Valid starting          Expires                Service principal
04/19/24 14:29:18      04/20/24 14:29:18      krbtgt/EFREI.FR@EFREI.FR
root@tpnagios-1:/#
```

Nous nous connectons ensuite à notre serveur depuis notre client en utilisant la commande suivante :

```
$ ssh -K thomas@192.168.1.137
```

Et nous obtenons le résultat suivant:

```
thomas@tpnagios-2: ~  
Welcome to Ubuntu 22.10 (GNU/Linux 5.15.131-2-pve x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
Your Ubuntu release is not supported anymore.  
For upgrade information, please visit:  
http://www.ubuntu.com/releaseendoflife  
  
New release '23.10' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Fri Apr 19 11:34:47 2024 from 192.168.27.65  
thomas@tpnagios-2:~$
```

Notre authentication kerberos via SSH fonctionne donc correctement.