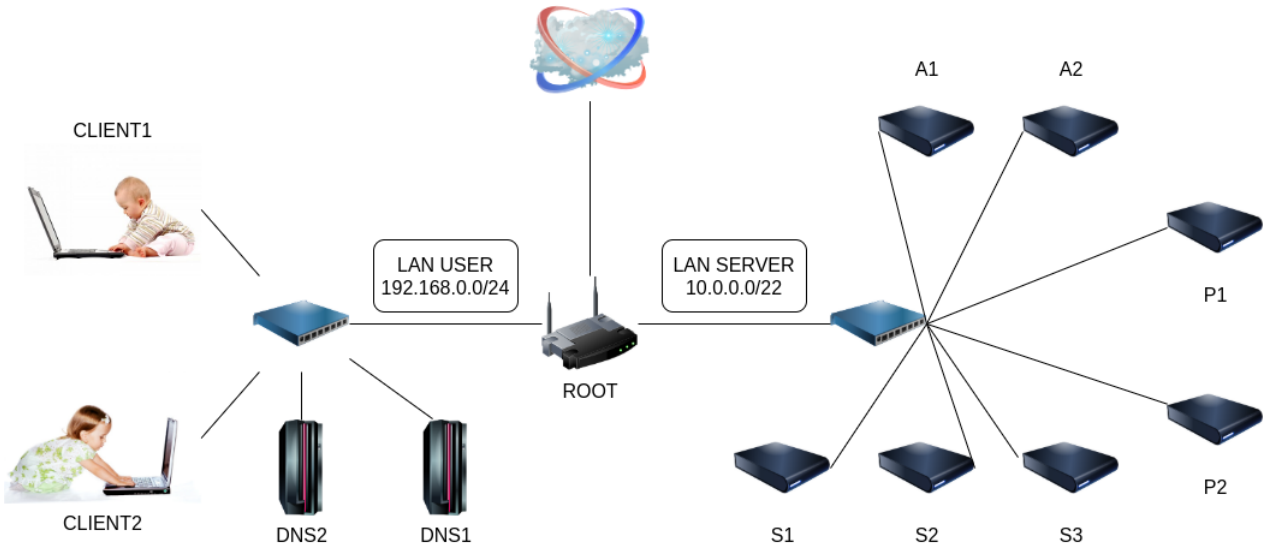
1 Objectifs

L'objectif de ce TP est de vous initier à la configuration d'un serveur **DNS** afin de simplifier la gestion d'un sous-réseau. Pour ce faire, vous allez utiliser un environnement virtuel émulant un réseau de machines sous *Debian*[1]. L'environnement virtuel que nous allons utiliser est *NEmu*[2].

2 Le réseau

Nous allons travailler sur le réseau suivant :



Nous pouvons constater que ce réseau est composé de deux sous-réseaux : *LAN USER* et *LAN SERVER*. *LAN USER* est composé des machines *dns1*, *dns2*, *client1* et *client2*. *LAN SERVER* est composé d'un ensemble de serveurs auxquels vous n'aurez pas accès dans un premier temps. Les deux sous-réseaux *LAN USER* et *LAN SERVER* sont raccordés à routeur *root* lui même connecté à internet. Les machines virtuelles vous sont livrées *nues*. C'est à dire qu'elles disposent uniquement des réglages élémentaires du système. Le mot de passe *root* est **plop**.

sectionAvant de commencer...

- Dans un terminal régulier :
  - Pour lancer le réseau virtuel :
    `/mnt/netta/apps/vnet/nemu-vnet netdns`
  - Pour restaurer le réseau virtuel précédemment sauvegardé :
    `/mnt/netta/apps/vnet/nemu-restore ~/vnet/netdns.tgz`
- Dans le terminal de *NEmu* :
  - Pour quitter le réseau virtuel, tapez `quit()` dans le terminal principal
  - Pour sauvegarder le réseau virtuel, tapez `save()` et validez dans le terminal principal. Le réseau sera sauvegardé dans *sim*`/vnet/netdns.tgz`
  - Pour redémarrer (violemment) l'ensemble du réseau virtuel, tapez `reboot()` et validez dans le terminal principal
  - Pour redémarrer une seule machine virtuelle : `RebootVNode("<nom de la VM>")`

1) Lancer le réseau virtuel comme indiqué ci-dessus. 4 fenêtres correspondant aux consoles de chacune des machines du *LAN USER* devraient apparaître.

1. http://www.debian.org
2. https://gitlab.com/v-a/nemu

Vincent Autefage
vincent.autefage@u-bordeaux.fr

3 Mise en jambe

3.1 Configuration IP des machines

Les machines sur *LAN SERVER* sont déjà configurées. Elles ne nécessiteront donc aucune configuration pendant toute la durée du TP.

2) Le routeur **root** dispose d'un serveur DHCP actif sur le *LAN USER* configuré statiquement. Éditer le fichier `/etc/network/interfaces` des deux serveurs DNS ainsi que des deux clients afin de pouvoir récupérer une adresse par DHCP.

3) Après avoir récupéré une adresse pour chaque machine via la commande `ifup`, noter la correspondance entre les adresses IP fournies et chaque machine.

4) Vérifier que les machines puissent communiquer entre elles grâce à la commande `ping`.

5) Analyser la table de routage de chaque machine grâce à la commande `route`.

3.2 Configuration du nommage locale

6) Le fichier `/etc/hosts` permet d'associer nom et IP de manière locale. Pour cela il suffit d'y écrire une correspondance :

```
<IP> <name>
```

Exemple :

```
127.0.0.1 localhost
127.0.0.1 moi
192.168.0.1 toto
```

Ainsi, les noms *localhost* et *moi* pointeront sur l'adresse 127.0.0.1 tandis que *toto* pointera sur 192.168.0.1.

7) Éditer le fichier `/etc/hosts` de **dns1** afin qu'il puisse communiquer avec les machines **dns2**, **client1** et **client2** par leur nom.

8) Effectuer un **ping** en utilisant ces noms plutôt que les IPs des machines afin de vérifier que la correspondance fonctionne.

3.3 Configuration des clients DNS

Bien que fort pratique, le nommage local ne permet pas d'effectuer une configuration globale pour l'ensemble d'un ou plusieurs sous-réseaux. Pour cela, nous allons devoir configurer un client DNS. Le client DNS a pour but d'interroger un serveur DNS lorsque le nom demandé n'est pas trouvé localement (*i.e.* dans le fichier `/etc/hosts`).

9) Vérifier le contenu du fichier `/etc/resolv.conf` sur chacune de nos machines. Vous constatez que ce fichier contient une adresse IP extérieure à votre sous-réseau, fournie par le serveur DHCP **root**.

10) L'objectif de ce TP étant de configurer nos propres serveurs DNS, nous allons ajouter une règle dans le fichier `/etc/dhcp/dhclient.conf` afin de forcer l'utilisation de nos propres serveurs. Pour cela ajouter la ligne suivante en fin de fichier :

```
# Pour dns1 et dns2
supersede domain-name-servers 127.0.0.1;

# Pour client1
supersede domain-name-servers  <IP de dns1>;

# Pour client2
supersede domain-name-servers  <IP de dns2>;
```

11) Analyser l'effet des commandes `ifdown --force eth0` puis `ifup --force eth0` sur le contenu du fichier `/etc/resolv.conf`.

12) Afin de demander à nos futurs serveurs DNS de relayer les requêtes DNS externes, éditer le fichier `/etc/bind/named.conf.options` sur **dns1** et **dns2** :

```
forwarders {
  172.16.0.3;
};

allow-query {
  any;
};

masterfile-format text;
```

4 Configuration des serveurs DNS

4.1 Configuration du serveur primaire

Nous allons dans un premier temps configurer le serveur **dns1** afin qu'il soit responsable du domaine principal **netas**.

13) Pour cela, déclarer une nouvelle zone DNS primaire dans le fichier `/etc/bind/named.conf.local` :

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

14) Il faut maintenant remplir notre zone DNS, pour cela copier le fichier `/etc/bind/db.empty` dans le fichier de zone que vous avez indiqué dans la déclaration de la zone :

```
# cp /etc/bind/db.empty /etc/bind/db.netas
```

15) Mettre à jour l'entête du fichier de zone :

```
@ IN SOA dns1.netas. contact.netas. (
 1      ; on oubliera pas d'incrémenter ce numéro de serial à chaque modification de la zone
 604800
 86400
 2419200
 86400 )
```

16) Ajouter une entrée **NS** permettant d'indiquer le nom du serveur DNS principal (*i.e.* **dns1**) :

```
@ IN NS dns1
```

17) Ajouter une entrée **A** permettant de spécifier l'adresse IP du nom **dns1** :

```
dns1 IN A <IP du serveur dns1>
```

18) Redémarrer le service DNS grâce à la commande suivante :

```
# systemctl restart named
```

Attention : Il faudra penser à mettre à jour le *serial* de la zone et redémarrer le service après chaque modification d'une zone DNS.

Astuce : L'action **reload** permet de recharger les fichiers de configuration sans avoir à redémarrer le service :

```
# systemctl reload named
```

19) Afin de vérifier que le service est bien démarré, vérifier son état à l'aide de la commande suivante :

```
# systemctl status named
```

Info : Les *logs* du service DNS sont stockés dans le *journal de systemd*. Il faudra vérifier le contenu de ce journal à chaque redémarrage du service DNS afin de s'assurer qu'aucune erreur silencieuse ne se soit produite. Ce journal contenant les *logs* accumulés, il peut devenir extrêmement volumineux et donc difficile à analyser. Vous pouvez n'afficher que les N dernières lignes d'un fichier grâce à la commande suivante :

```
# journalctl --unit <service> --lines <N>
```

Exemple :

```
# journalctl --unit named --lines 20  # affiche les 20 derniers logs du service DNS
```

Il est également possible de consulter les *logs* en temps réel via la commande suivante :

```
# journalctl --unit named --follow
```

20) Tester votre configuration depuis **client1** à l'aide de la commande `ping` effectuée sur le nom **dns1.netas** configuré précédemment.

21) Ajouter une entrée **A** pour **dns2** dans la zone.

22) Ajouter deux entrées **CNAME** afin de donner les alias **dns-primaire** et **dns-secondaire** respectivement aux serveurs **dns1** et **dns2**.

23) Tester votre configuration depuis **client1** à l'aide de la commande `ping` vers **dns1.netas**, **dns2.netas**, **dns-primaire.netas**, **dns-secondaire.netas**.

Astuce : La commande `host` permet d'effectuer une requête DNS à un serveur spécifique :

```
# host toto 192.168.0.1    # demande au serveur 192.168.0.1 de résoudre le nom toto

# host toto                # demande au serveur indiqué dans /etc/resolv.conf de résoudre
                           # le nom toto sauf si toto est référencé dans /etc/hosts
```

4.2 Configuration du serveur secondaire

Le serveur **dns2** va être utilisé comme serveur secondaire pour la zone précédemment définie sur le serveur **dns1** (*i.e.* **netas**). L'objectif est donc de mettre en place une mécanique permettant à **dns2** de récupérer de façon autonome la configuration établie sur **dns1**.

24) Dans la configuration de la zone sur **dns1**, ajouter une nouvelle entrée **NS** permettant d'identifier **dns2** comme un serveur de la zone :

```
@ IN NS dns2
```

25) Sur le serveur **dns2**, déclarer la zone **netas** comme secondaire du serveur **dns1** :

```
zone "netas" {
  type slave;
  file "/var/lib/bind/db.netas";
  masters { <IP du serveur dns1>; };
};
```

26) Après avoir redémarrer le service DNS sur **dns1** puis **dns2**, vérifier que la machine **client2** peux joindre les machines déclarées dans la zone **netas** à l'aide de la commande `ping`.

27) Depuis les deux clients, effectuer une requête avec la commande `host` en utilisant en premier le serveur **dns1** puis ensuite **dns2** afin de s'assurer que les deux serveurs contiennent bien notre zone **netas** :

Exemple :

```
# host dns-primaire.netas <IP de dns1>
# host dns-primaire.netas <IP de dns2>
# host dns-secondaire.netas <IP de dns1>
# host dns-secondaire.netas <IP de dns2>
```

4.3 Étude du *LAN SERVER*

Nous allons maintenant étudier la configuration réseau du *LAN SERVER*.

28) À quoi correspond le masque /**22** ? Lister les adresses possibles sur ce sous-réseaux.

29) Depuis **dns1**, nous allons tout d'abord effectuer un scan du réseau *LAN SERVER* afin de récupérer les adresses IP des machines qui composent ce sous-réseaux :

```
# nmap -T5 -sP 10.0.0.0/22   # le scan va prendre environ 30 secondes
```

30) Se connecter en **ssh** à chacune des IPs collectées afin d'établir la correspondance entre chaque machine qui composent le *LAN SERVER* et leur IP respective. Un compte local est disponible sur chacune de ces machines dont le nom d'utilisateur est *tc* et le mot de passe *plop*.

Rappel :

```
# hostname   # permet de récupérer le nom d'une machine lorsqu'on est connecté à celle-ci
```

4.4 Configuration du domaine principale

Les machines **s1**, **s2** et **s3** héberge chacune un site web. Nous allons donc ajouter ces machines à notre zone DNS **netas** afin de simplifier leur accès.

31) Ajouter une entrée **A** pour chaque serveur web indiqué ci-dessus.

32) Ajouter une entrée **CNAME** pour chaque serveur web indiqué ci-dessus selon la correspondance suivante :

s1 : **creative**

s2 : **grayscale**

s3 : **wonder**

33) Tester le bon fonctionnement de vos entrées principales ainsi que de vos alias depuis les machines **client1** (qui utilise **dns1**) et **client2** (qui utilise **dns2**).

34) Passer en mode graphique grâce à la commande `startx` et démarrer un navigateur web sur **client1**.

35) Tenter de vous connecter aux sites web précédemment enregistrés dans notre zone DNS.

36) Effectuer le même test sur **client2** afin de vérifier que la zone du serveur secondaire **dns2** ait bien été mise à jour.

4.5 Configuration de la zone inverse

Le DNS permet de retrouver une IP à partir d'un nom mais il permet également l'opération inverse.

37) Déclarer la zone inverse du domaine principal **netas** dans le fichier `/etc/bind/named.conf.local` du serveur **dns1** et créer le fichier de zone associée :

Pour `/etc/bind/named.conf.local` :

```
zone "2.0.10.in-addr.arpa" {
  type master;
  file "/etc/bind/db.netas-rev";
};
```

Pour `/etc/bind/db.netas-rev` :

```
@ IN SOA dns1.netas. contact.netas. (
 1
 604800
 86400
 2419200
 86400 )

@ IN NS dns1.netas.
@ IN NS dns2.netas.
1 IN PTR s1.netas.
2 IN PTR s2.netas.
3 IN PTR s3.netas.
```

38) Vérifier sur **client1** avec l'aide des commandes `ping` et `host` que vous obtenez bien le nom associé en utilisant l'adresse IP ou les alias des machines **s1**, **s2** et **s3**.

39) Mettre à jour la configuration du serveur **dns2** pour que ce dernier devienne un serveur secondaire de la zone inverse **netas**.

40) Vérifier le bon fonctionnement de la zone inverse secondaire sur **client2** avec l'aide des commandes `ping` et `host`.

4.6 Configuration d'un sous-domaine

Les machines **p1** et **p2** proposent un service de mesure de débit. Nous allons ajouter un sous-domaine **perf.netas** afin de proposer ce service.

41) Ajouter dans la zone principale **netas** une nouvelle entrée **NS** afin d'inclure le nouveau sous-domaine géré par le serveur **dns1** lui même :

```
perf IN NS dns1
```

42) Déclarer ce nouveau sous-domaine dans `/etc/bind/named.conf.local` :

```
zone "perf.netas" {
  type master;
  file "/etc/bind/db.perf.netas";
};
```

43) Créer le fichier de zone associé :

```
@ IN SOA dns1.perf.netas. contact.netas. (
 1
 604800
 86400
 2419200
 86400 )
@ IN NS dns1
dns1 IN A  <IP de dns1>
```

44) Compléter le fichier de zone afin d'inclure les entrées **A** pointant sur **p1** et **p2**.

45) Vérifier votre configuration depuis **client1** à l'aide des commandes `ping` et `host`.

46) Dans le but de répartir la charge de calcul entre ces deux serveurs, nous allons associer la même entrée **A** pour l'adresse **scale.perf.netas** aux deux serveurs **p1** et **p2** :

```
scale IN A <IP de p1>
scale IN A <IP de p2>
```

47) Sur **client1**, vérifier que l'adresse retournée change bien de façon pseudo-aléatoire grâce à la commande `ping`.

48) Sur **client1** et **client2**, installer le programme `iperf` :

```
# apt install iperf
```

49) Mettre à jour la configuration des serveurs **dns1** et **dns2** pour que **dns2** deviennent un serveur secondaire de la zone **perf.netas**.

50) Sur **client2**, vérifier votre configuration grâce à la commande `ping`.

51) Effectuer des tests de débits simultanément depuis **client1** et **client2** afin de vérifier que les deux serveurs **p1** et **p2** sont bien utilisés en parallèle :

```
# iperf -t 5 -c scale.perf.netas    # prend entre 5 et 10 secondes pour retourner un résultat
```

52) Ajouter la zone inverse de la zone **perf.netas** sur **dns1** (principal) et **dns2** (secondaire) ; **p1** et **p2** doivent pointer au même nom **scale.perf.netas** :

```
1 IN PTR scale.perf.netas.
2 IN PTR scale.perf.netas.
```

4.7 Configuration d'un nouveau sous-domaine

Les machines **a1** et **a2** proposent des sites web d'administration. Nous allons les inclure dans un nouveau sous-domaine prévu à cet effet.

53) Configurer une nouveau sous-domaine **admin.netas** contenant les machines **a1** (alias **dash.admin.netas**) et **a2** (alias **ela.admin.netas**). La zone doit être primaire sur **dns1** et secondaire sur **dns2**. N'oubliez pas de configurer la zone inverse relative.

54) Vérifier le bon fonctionnement de votre configuration dans les navigateurs de **client1** et **client2**.

5 Mise en place d'une attaque de type *DNS cache poisoning* (bonus)

Principe

Il est possible d'empoisonner le cache DNS par une attaque de type *spoofing*. Nous allons effectuer cette attaque en faisant croire à *client1* que le site qu'elle souhaite contacter est en fait hébergé sur une machine attaquante masquée sur l'IP *192.168.0.42*. L'attaque sera effectuée depuis *client2*.

5.1 A l'abordage !

55) Il est tout d'abord nécessaire d'écrire sur *client2* un fichier de correspondance DNS respectant la syntaxe suivante.

```
<@IP pirate> <expression d'un nom de domaine>

Exemple:
192.168.0.42 google.fr
192.168.0.42 *.google.fr
192.168.0.42 u-bordeaux.fr
192.168.0.42 *.u-bordeaux.fr
192.168.0.42 *.cat
```

56) Effectuez la procédure d'*APR Spoofing* depuis *client2* pour faire croire à *client1* que son serveur DNS est *client2*. N'oubliez pas d'activer l'*IP forwarding*.

57) Activez la procédure de *DNS Spoofing* depuis *client2*.

```
# dnsspoof -i <iface> -f <file>
```

58) Essayez de vous connecter à un site présent dans votre fichier de correspondances depuis *client1*.

59) Vous constatez que *client1* pense être sur un site qui n'est en fait pas le bon. Nous avons donc réussi.

60) Stoppez `dnsspoof` ainsi qu'`arpspoof`.

6 Fin

61) Éteindre chaque machine correctement à l'aide de la commande `halt`. Vous pouvez ensuite sauvegarder votre session à l'aide de la commande `save()` et quitter l'environnement avec la commande `quit()` dans le terminal principal.

Vincent Autefage
vincent.autefage@u-bordeaux.fr