DNS HOWTO

Nicolai Langfeldt <janl@math.uio.no>
Traduction française par Mathieu Arnold <arn_mat@club-internet.fr> Version 2.2, du 11 février 1999

Comment devenir un administrateur de DNS à la petite semaine.

Table des matières

1 Préambule

Mots-clés : DNS, bind, bind-4, bind-8, named, dialup, ppp, slip, isdn, Internet, domain, name, hosts, resolving, caching

Ce document fait partie du *Linux Documentation Project*.

1.1 Aspect juridique

1.2 Remerciements et appel aux bonnes volontés

J'aimerais remercier Arnt Gulbrandsen qui a tant souffert en relisant les brouillons de ce document et qui a apporté nombre de suggestions pertinentes. Merci également à tous ceux qui m'ont envoyé leurs suggestions par courrier électronique. Merci beaucoup ! Vous m'aidez vraiment dans ce travail.

Ce document n'est pas destiné à atteindre un jour un état final, alors faites-moi part de vos problèmes ainsi que de vos succès, cela me permettra d'améliorer ce HOWTO. Merci d'envoyer les commentaires et/ou les questions et même l'argent à janl@math.uio.no. Si vous m'envoyez un courrier électronique, merci de *vérifier* que votre adresse de retour est correcte car je reçois *beaucoup* de courrier électronique. Essayez aussi de lire le chapitre 8 (FAQ) avant de m'envoyer un mail. Autre chose je (l'auteur) ne parle qu'anglais et norvégien.

Si vous ne parlez ni l'anglais ni le norvégien, vous pouvez toujours envoyer vos commentaires en français au traducteur (arn_mat@club-internet.fr) qui fera suivre.

Si vous voulez traduire ce HOWTO, prévenez-moi pour que je puisse garder le compte de toutes les langues dans lesquelles il a été traduit :-), de plus, cela me permettra de vous tenir au courant des évolutions de ce HowTo.

1.3 Dédicace

Ce HOWTO est dédié à Anne Line Norheim. Pourtant, elle ne le lira sans doute jamais, ce n'est pas du tout son genre.

2 Introduction.

Ce que ce document est et ce qu'il n'est pas

Le DNS est le *Domain Name System*. C'est l'ensemble des règles utilisées par les machines et les logiciels pour établir, entre autres choses, la correspondance entre les noms de machines et les adresses IP, dont chaque machine sur le net est pourvue. Ce document explique comment définir de telles correspondances à l'aide d'un système Linux. Une correspondance est tout simplement une relation entre deux objets, dans notre cas un nom de machine, comme `ftp.linux.org`, et l'adresse IP de cette machine, `199.249.150.4`.

Le DNS constitue pour le non-initié (vous dans le cas présent ;-) une des parties les plus obscures de l'administration de réseau. Le but de ce HOWTO est d'essayer d'en éclaircir quelques aspects. Ce document explique comment configurer un DNS *simple*. Nous allons commencer avec un serveur de noms qui ne sert qu'à faire cache puis nous continuerons en configurant un serveur DNS primaire pour un domaine. Pour des configurations plus complexes, jetez un coup d'oeil à la section 8 (FAQ) de ce document. Si vous n'y trouvez pas ce que vous cherchez, vous allez alors devoir *lire* la Vraie Documentation. Je reviendrai sur ce en quoi consiste la Vraie Documentation dans le chapitre 9 (final).

Avant de commencer, vous devez configurer votre machine pour être capable de vous connecter par telnet sur d'autres machines mais aussi pouvoir recevoir des connexions sur votre machine. Vous devez aussi être en mesure de vous connecter au réseau par tous les services possibles, et en particulier pouvoir faire `telnet 127.0.0.1`, ce qui revient à vous connecter à votre propre machine (vérifiez tout de suite que ça marche !). Il est aussi nécessaire, pour commencer, que les fichiers `/etc/nnswitch.conf` (ou `/etc/host.conf`), `/etc/resolv.conf` et `/etc/hosts` soient correctement configurés car je n'expliquerai pas ici à quoi ils servent. Si tout cela n'est pas déjà configuré et en état de marche, lisez le NET-3-HOWTO.

Si vous utilisez une connexion SLIP ou PPP, il est indispensable qu'elle fonctionne. Lisez le PPP HOWTO si ce n'est pas le cas.

Quand je dis "votre machine", j'entends la machine sur laquelle vous aller essayer d'installer le DNS, et non pas une autre machine dont vous pourriez vous servir pour accéder au réseau.

Je supposerai par la suite que vous ne vous trouvez pas derrière un firewall qui bloque les requêtes de résolution de nom. Si tel est le cas, vous aurez besoin d'une configuration spéciale. Reportez-vous alors au chapitre 8 (FAQ).

Le service de résolution de nom sous Unix est assuré par un programme appelé `named`. Il fait partie du paquetage "bind", géré par Paul Vixie pour l'Internet Software Consortium. `named` est inclus dans la plupart des distributions de Linux et se trouve le plus souvent installé dans `/usr/sbin/named`. Si vous disposez d'un `named`, vous pouvez vraisemblablement l'utiliser. Si vous n'en avez pas, chargez-en un à partir d'un site FTP Linux ou allez chercher la dernière et meilleure version des sources du programme depuis *ftp.isc.org/isc/bind/src/cur/bind-8/*. Ce HowTo parle de bind version 8. L'ancienne version de ce HowTo, a propos de bind 4 est toujours disponible à *www.math.uio.no/˜janl/DNS/* au cas ou vous auriez bind 4. Si la page man de `named` parle de `named.conf` vous avec bind 8, si elle parle (tout a la fin, dans la section FILES) de `named.boot` vous avez bind 4. Si vous avez bind 4, et si la sécurité fait partie de vos préoccupations, vous devriez vraiment passer à bind 8.

Le service DNS est une base de données à l'échelle du réseau tout entier. Faites donc très attention à ce que vous y introduisez. Si vous y mettez n'importe quoi, vous en retirerez n'importe quoi, et les autres aussi. Conservez votre DNS bien propre, à jour et cohérent et vous verrez qu'il vous offrira le meilleur de lui-même. Apprenez a l'utiliser, l'administrer, le débogger et vous ferez partie de ces administrateurs qui empêchent que le réseau ne s'écroule sous le poids des systèmes mal gérés.

Dans ce document, je dis des choses qui ne sont pas tout à fait vraies (mais qui le sont toujours au moins à moitié). Si je le fais, c'est toujours dans le but de rendre les choses plus simples. Tout marchera (probablement ;-) très bien si vous croyez ce que je vous dis.

Astuce : S'ils existent déjà, faites une copie de sauvegarde de tous les fichiers que je vous demande de modifier. Ainsi, si plus rien ne marche après ce que nous allons faire, vous pourrez toujours revenir au bon vieux temps où tout marchait bien.

3 Un Serveur de Noms qui ne sert que de cache.

Un premier aperçu de la configuration d'un DNS, très utile pour ceux qui utilisent une

connexion en dialup.

Un serveur de noms qui ne sert que de cache trouve la réponse aux requêtes de résolution de nom et se souvient de cette réponse chaque fois qu'on lui posera la même question par la suite. Cela réduira les temps de réponse, surtout si vous avez une connexion plutôt lente.

Vous avez tout d'abord besoin du fichier `/etc/named.conf`. Ce fichier est lu au lancement de `named`. Pour le moment, il ne doit pas contenir autre chose que :

```
// Fichier de config pour un serveur de noms qui ne fait que du cache

options {
        directory "/var/named";

        // Enlever les commentaires peut vous aider si vous avez a passer a
        // travers un firewall et que ça ne marche pas :

        // query-source port 53;
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.in-addr.arpa" {
        type master;
        file "pz/127.0.0";
};
```

TRÈS IMPORTANT : Dans certaines versions de ce document, les fichiers listés comme ci-dessus présentent un certain nombre de caractères espace ou tabulation avant le premier caractère non blanc de la ligne. Ils ne sont pas supposés faire partie du fichier. **Effacez donc tous les caractères blancs** de début de ligne des fichiers que vous copiez-collez à partir de ce HOWTO.

La ligne "`directory`" indique à `named` l'endroit où il doit rechercher ses fichiers. Tous les fichiers dont nous parlerons maintenant auront un chemin relatif relatifs à ce répertoire. Ainsi, `pz` est un sous-répertoire de `/var/named`, c'est a dire `/var/named/pz`. D'après le *Linux Filesystem Standard*, ce répertoire doit être `/var/named`.

On trouve à cet endroit le fichier `/var/named/root.hints`, qui doit ressembler à ceci :

```
; Il se peut qu'il y ait quelques commentaires ici si vous avez déjà ce
; fichier. de toutes façon, ce sont des commentaires, ils ne sont pas
; important du tout.

.                       6D IN NS        G.ROOT-SERVERS.NET.
.                       6D IN NS        J.ROOT-SERVERS.NET.
.                       6D IN NS        K.ROOT-SERVERS.NET.
.                       6D IN NS        L.ROOT-SERVERS.NET.
.                       6D IN NS        M.ROOT-SERVERS.NET.
.                       6D IN NS        A.ROOT-SERVERS.NET.
.                       6D IN NS        H.ROOT-SERVERS.NET.
.                       6D IN NS        B.ROOT-SERVERS.NET.
```

```
.                        6D IN NS       C.ROOT-SERVERS.NET.
.                        6D IN NS       D.ROOT-SERVERS.NET.
.                        6D IN NS       E.ROOT-SERVERS.NET.
.                        6D IN NS       I.ROOT-SERVERS.NET.
.                        6D IN NS       F.ROOT-SERVERS.NET.

G.ROOT-SERVERS.NET.      5w6d16h IN A   192.112.36.4
J.ROOT-SERVERS.NET.      5w6d16h IN A   198.41.0.10
K.ROOT-SERVERS.NET.      5w6d16h IN A   193.0.14.129
L.ROOT-SERVERS.NET.      5w6d16h IN A   198.32.64.12
M.ROOT-SERVERS.NET.      5w6d16h IN A   202.12.27.33
A.ROOT-SERVERS.NET.      5w6d16h IN A   198.41.0.4
H.ROOT-SERVERS.NET.      5w6d16h IN A   128.63.2.53
B.ROOT-SERVERS.NET.      5w6d16h IN A   128.9.0.107
C.ROOT-SERVERS.NET.      5w6d16h IN A   192.33.4.12
D.ROOT-SERVERS.NET.      5w6d16h IN A   128.8.10.90
E.ROOT-SERVERS.NET.      5w6d16h IN A   192.203.230.10
I.ROOT-SERVERS.NET.      5w6d16h IN A   192.36.148.17
F.ROOT-SERVERS.NET.      5w6d16h IN A   192.5.5.241
```

Souvenez-vous bien de ce que j'ai dit pour les caractères blancs en tête de ligne !

Ce fichier donne une description de tous les serveurs de noms du monde qui se trouvent à la racine (au plus haut niveau) de la hiérarchie des serveurs de noms. Il arrive que cette liste change, c'est pourquoi il est essentiel que ce fichier soit maintenu à jour. Reportez-vous à la section 6 (maintenance) pour savoir comment le garder à jour. Le contenu de ce fichier est décrit dans la page de man de `named` mais cette dernière s'adresse plus, à mon humble avis, à ceux qui savent déjà comment fonctionne ce programme.

La section suivante de `named.conf` est la dernière partie. Elle sera expliquée dans un chapitre suivant, pour l'instant, créez un fichier appelé `127.0.0` dans le sous répertoire `pz` :

```
@               IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                                1       ; Serial
                                8H      ; Refresh
                                2H      ; Retry
                                1W      ; Expire
                                1D)     ; Minimum TTL
                        NS      ns.linux.bogus.
1                       PTR     localhost.
```

Après ça, vous avez besoin d'un fichier `/etc/resolv.conf` qui ressemble à peu près à ça :

```
search subdomain.your-domain.edu your-domain.edu
nameserver 127.0.0.1
```

La ligne `search` spécifie dans quels domaines il faudra chercher lorsque vous voudrez vous connecter sur une machine de nom quelconque. La ligne "`nameserver`" indique à quelle adresse votre machine peut contacter un serveur de noms. Si vous voulez indiquer plusieurs serveurs de nom, mettez une ligne "`nameserver`" pour chacun. Dans notre cas, il s'agit de notre propre machine puisque c'est elle qui fait tourner `named`. (Note : `named` ne lit jamais ce fichier, c'est le *résolveur* qui utilise `named` qui le fait).

Voyons sur un exemple à quoi sert ce fichier : si un client cherche à contacter `foo`, on essaye d'abord `foo.subdomain.your-domain.edu` puis `foo.your-domain.edu` et enfin `foo`. Si un client essaye de contacter `sunsite.unc.edu`, on essaye d'abord `sunsite.unc.edu.subdomain.your-domain.edu` (je sais, c'est stupide, mais c'est comme ça) puis `sunsite.unc.edu.your-domain.edu` et enfin `sunsite.unc.edu`. Faites attention à ne pas mettre trop de noms de domaine dans la ligne `search` car cela prend du temps de tous les essayer.

Cet exemple suppose que vous appartenez au domaine `subdomain.your-domain.edu`. Votre machine s'appelle alors certainement `your-machine.subdomain.your-domain.edu`. La ligne search ne doit pas contenir votre TLD (Top Level Domain ; `edu` dans notre cas). Si vous vous connectez fréquemment à des machines dans un autre domaine, vous pouvez rajouter ce domaine dans la ligne search comme ceci :

```
search subdomain.your-domain.edu your-domain.edu other-domain.com
```

et ainsi de suite. Évidemment, il faut appliquer cet exemple à de vrais noms de domaines. Remarquez qu'ici il n'y a pas de point à la fin des noms de domaine. C'est important, notez l'absence de points aux fins des noms de domaines.

Ensuite, suivant votre version de la libc, vous allez devoir modifier soit `/etc/nsswitch.conf`, soit `/etc/host.conf`. Si vous avez déjà `nsswitch.conf`, c'est celui-là que nous allons modifier, sinon ce sera `host.conf`.

/etc/nsswitch.conf

C'est un long fichier qui spécifie où trouver différentes sortes de types de données, dans quel fichier ou quelle base de données. Il contient généralement des commentaires précieux au début, que vous auriez tout intérêt à lire. Ensuite, trouvez la ligne qui commence par "`hosts :`", elle doit ressembler à ceci :

```
hosts: files dns
```

Si il n'y a aucune ligne qui commence par "`hosts :`", mettez celle ci-dessus. Elle dit que les programmes doivent d'abord regarder dans `/etc/hosts` puis demander au DNS en suivant les indications de `resolv.conf`.

/etc/host.conf

Ce fichier contient certainement plusieurs lignes, dont une doit commencer par `order` et ressembler à ça :

```
order hosts,bind
```

Si il n'y a pas de ligne "`order`", il faut en mettre une. Elle indique aux routines de résolution de nom de regarder d'abord dans `/etc/hosts` puis de demander au serveur de noms (que vous avez précisé dans `resolv.conf` comme étant 127.0.0.1).

3.1 Démarrer named.

Après tout ça, il est temps de démarrer `named`. Si vous utilisez une connexion en dialup, commencez par vous connecter. Tapez "`ndc start`" et appuyez sur la touche entrée, sans donner d'options. Si ça ne marche pas, essayez plutôt "`/usr/sbin/ndc start`". Si ça ne marche toujours pas, jetez un coup d'oeil au chapitre 8 (FAQ). Si vous jetez un oeil à votre fichier de messages syslog (souvent appelé `/var/adm/messages`, mais regardez également dans le répertoire `/var/log` ou dans le fichier `syslog`) tout en lançant `named` (faites `tail -f /var/adm/messages`), vous devriez voir quelque chose comme ça :

(les lignes se terminant par \ se continuent sur la ligne suivante)

```
Feb 15 01:26:17 roke named[6091]: starting.  named 8.1.1 Sat Feb 14 \
  00:18:20 MET 1998 ^Ijanl@roke.uio.no:/var/tmp/bind-8.1.1/src/bin/named
Feb 15 01:26:17 roke named[6091]: cache zone "" (IN) loaded (serial 0)
Feb 15 01:26:17 roke named[6091]: master zone "0.0.127.in-addr.arpa" \
  (IN) loaded (serial 1)
Feb 15 01:26:17 roke named[6091]: listening [127.0.0.1].53 (lo)
Feb 15 01:26:17 roke named[6091]: listening [129.240.230.92].53 (ippp0)
```

```
Feb 15 01:26:17 roke named[6091]: Forwarding source address is [0.0.0.0].1040
Feb 15 01:26:17 roke named[6092]: Ready to answer queries.
```

Si il y a un quelconque message d'erreur, `named` donnera le nom du fichier dans lequel se trouve l'erreur (soit `named.conf`, soit root.hints, j'espère :-). Tuez le processus `named` et re-vérifiez ce fichier.

Il est maintenant temps de vérifier votre configuration. Lancez `nslookup` pour regarder le résultat de votre petit travail.

```
$ nslookup
Default Server: localhost
Address: 127.0.0.1

>
```

Si vous obtenez ce message, c'est que ça marche. Nous l'espérons tous. Si vous obtenez quoi que ce soit d'autre, revenez en arrière et vérifiez tout. Chaque fois que vous modifiez le fichier `named.conf`, il vous faut relancer `named` avec la commande `ndc restart`.

Maintenant, vous pouvez entrer une requête. Essayez de contacter une machine proche de vous. `pat.uio.no` est proche de moi, à l'Université d'Oslo :

```
> pat.uio.no
Server: localhost
Address: 127.0.0.1

Name:    pat.uio.no
Address: 129.240.2.50
```

nslookup a demandé à votre `named` de rechercher la machine `pat.uio.no`. Il a ensuite contacté un des serveurs de noms mentionnés dans `root.cache` et a demandé le chemin à suivre. Il peut s'écouler un certain temps avant que vous obteniez le résultat puisqu'il se peut qu'il recherche tous les domaines listés dans `/etc/resolv.conf`.

Si vous réessayez, vous obtiendrez ceci :

```
> pat.uio.no
Server: localhost
Address: 127.0.0.1

Non-authoritative answer :
Name: pat.uio.no
Address: 129.240.2.50
```

Notez cette fois ci l'apparition de la ligne “`Non-authoritative answer :`”. Elle veut dire que `named` n'a pas accédé au réseau pour obtenir la réponse mais a trouvé l'information dans son cache. Cependant, l'information cachée *pourrait* ne plus être à jour. C'est pourquoi vous êtes informé de cette possibilité très improbable par le message “`Non-authoritative answer :`” Quand `nslookup` répond ceci la seconde fois qu'on lui demande un certain hôte, c'est un signe certain que `named` cache bien les informations et que tout marche. Pour sortir de `nslookup`, utilisez la commande “`exit`”.

3.2 Le rendre encore meilleur

Dans les grands réseaux, bien administrés, des universités ou FAI (Fournisseur d'Accès a Internet), vous remarquerez peut-être que les administrateurs réseau ont mis en place une hiérarchie de serveurs DNS ce qui permet de soulager le réseau interne ainsi que le réseau vers l'extérieur. Il n'est pas facile de savoir si vous êtes dans un réseau de ce type. Tout cela n'est pas très important, mais en utilisant le serveur DNS de votre FAI comme "forwarder" vous pouvez rendre les réponses plus rapides et alléger la charge de votre réseau. Avec un modem, la différence peut être sensible. Pour améliorer encore notre exemple, supposons que votre FAI aie deux serveurs de noms qu'il veux vous faire utiliser, ayant pour adresses IP `10.0.0.1` et `10.1.0.1`. Alors, dans votre fichier `named.conf`, dans la section appelée "options" insérez les lignes :

```
forward first;
forwarders {
     10.0.0.1;
     10.1.0.1;
 };
```

Redémarrez votre serveur de noms et testez avec `nslookup`. Cela devrait marcher sans problèmes.

3.3 Félicitations !

Maintenant, vous savez comment configurer un `named` qui sert de cache. Servez-vous une bière, un verre de lait ou tout ce que vous voudrez pour fêter l'événement.

4 Un domaine *simple*

Comment mettre en place votre propre domaine

4.1 Mais avant tout, un brin de théorie

Avant d'entrer *vraiment* dans le vif du sujet, il va falloir que je fasse un brin de théorie avec quand même un petit exemple sur le principe du service DNS. Et il faudra tout lire, car c'est pour votre bien. Vous devriez au moins survoler rapidement cette section. Arrêtez le survol quand vous arrivez à l'endroit où j'explique le contenu du fichier `named.conf`.

Le service DNS est un système organisé de manière hiérarchique, sous forme d'arbre. La racine est désignée par "`.`" et s'appelle "la racine". En dessous de `.` se trouvent un certain nombre de TLD (*Top Level Domains*) ; les plus connus sont `ORG`, `COM`, `EDU`, `NET` et `FR`, mais il y en a beaucoup d'autres. Tout comme un arbre, il a une racine avec des branches qui en partent. Si vous avez des connaissances en informatique fondamentale, vous reconnaîtrez dans le DNS un arbre de recherche, avec des noeuds, des arrêtes et des feuilles.

Lorsque vous recherchez une machine, la question est posée récursivement dans toute la hiérarchie depuis la racine. Lorsque vous voulez trouver l'adresse IP de `prep.ai.mit.edu`, votre DNS doit trouver un serveur de noms pour le domaine `edu`. Votre DNS demande d'abord à un serveur de noms de `.` (il possède déjà les adresses des serveurs pour `.`, elles sont dans le fichier `root.hints`), et le serveur pour `.` donne une liste des serveurs d'`edu`.

Voici un exemple :

```
$ nslookup
Default Server: localhost
Address: 127.0.0.1
```

Interrogeons un serveur situé à la racine.

```
> server c.root-servers.net.
Default Server: c.root-servers.net
Address: 192.33.4.12
```

Positionnons le type de requête (Query Type) à NS (Name Server records).

```
> set q=ns
```

Posons la question à propos de `edu`.

```
> edu.
```

Le `.` terminal est significatif, il indique à `nslookup` que nous interrogeons que `edu` se trouve juste sous `.` (et pas dans l'un de nos sous-domaines, ce qui accélère la recherche).

```
edu     nameserver = A.ROOT-SERVERS.NET
edu     nameserver = H.ROOT-SERVERS.NET
edu     nameserver = B.ROOT-SERVERS.NET
edu     nameserver = C.ROOT-SERVERS.NET
edu     nameserver = D.ROOT-SERVERS.NET
edu     nameserver = E.ROOT-SERVERS.NET
edu     nameserver = I.ROOT-SERVERS.NET
edu     nameserver = F.ROOT-SERVERS.NET
edu     nameserver = G.ROOT-SERVERS.NET
A.ROOT-SERVERS.NET      internet address = 198.41.0.4
H.ROOT-SERVERS.NET      internet address = 128.63.2.53
B.ROOT-SERVERS.NET      internet address = 128.9.0.107
C.ROOT-SERVERS.NET      internet address = 192.33.4.12
D.ROOT-SERVERS.NET      internet address = 128.8.10.90
E.ROOT-SERVERS.NET      internet address = 192.203.230.10
I.ROOT-SERVERS.NET      internet address = 192.36.148.17
F.ROOT-SERVERS.NET      internet address = 192.5.5.241
G.ROOT-SERVERS.NET      internet address = 192.112.36.4
```

Nous apprenons ainsi que tous les serveurs `ROOT-SERVERS.NET` servent le domaine `edu.` ; nous pouvons donc continuer en les interrogeant tous. Nous continuerons en interrogeant `C`. Maintenant, nous voulons savoir qui sert le niveau suivant du nom de domaine : `mit.edu.` :

```
> mit.edu.
Server:  c.root-servers.net
Address:  192.33.4.12

Non-authoritative answer:
mit.edu nameserver = STRAWB.mit.edu
mit.edu nameserver = W20NS.mit.edu
mit.edu nameserver = BITSY.mit.edu

Authoritative answers can be found from:
STRAWB.mit.edu  internet address = 18.71.0.151
W20NS.mit.edu   internet address = 18.70.0.160
BITSY.mit.edu   internet address = 18.72.0.3
```

`strawb`, `w20ns` et `bitsy` servent tous le domaine `mit`, prenons-en un au hasard et posons-lui la question au sujet d'un domaine encore plus précis : `ai.mit.edu` :

```
> server W20NS.mit.edu.
```

On ne distingue pas majuscules et minuscules pour les noms de domaine, et comme j'utilise ma souris pour faire du copier-coller, vous lisez les choses dans ce document telles qu'elles apparaissent sur mon écran.

```
Server:  W20NS.mit.edu
Address:  18.70.0.160

> ai.mit.edu.
Server:  W20NS.mit.edu
Address:  18.70.0.160

Non-authoritative answer:
ai.mit.edu      nameserver = ALPHA-BITS.AI.MIT.EDU
ai.mit.edu      nameserver = GRAPE-NUTS.AI.MIT.EDU
ai.mit.edu      nameserver = TRIX.AI.MIT.EDU
ai.mit.edu      nameserver = MUESLI.AI.MIT.EDU
ai.mit.edu      nameserver = LIFE.AI.MIT.EDU
ai.mit.edu      nameserver = BEET-CHEX.AI.MIT.EDU
ai.mit.edu      nameserver = MINI-WHEATS.AI.MIT.EDU
ai.mit.edu      nameserver = COUNT-CHOCULA.AI.MIT.EDU
ai.mit.edu      nameserver = MINTAKA.LCS.MIT.EDU

Authoritative answers can be found from:
AI.MIT.EDU      nameserver = ALPHA-BITS.AI.MIT.EDU
AI.MIT.EDU      nameserver = GRAPE-NUTS.AI.MIT.EDU
AI.MIT.EDU      nameserver = TRIX.AI.MIT.EDU
AI.MIT.EDU      nameserver = MUESLI.AI.MIT.EDU
AI.MIT.EDU      nameserver = LIFE.AI.MIT.EDU
AI.MIT.EDU      nameserver = BEET-CHEX.AI.MIT.EDU
AI.MIT.EDU      nameserver = MINI-WHEATS.AI.MIT.EDU
AI.MIT.EDU      nameserver = COUNT-CHOCULA.AI.MIT.EDU
AI.MIT.EDU      nameserver = MINTAKA.LCS.MIT.EDU
ALPHA-BITS.AI.MIT.EDU   internet address = 128.52.32.5
GRAPE-NUTS.AI.MIT.EDU   internet address = 128.52.36.4
TRIX.AI.MIT.EDU internet address = 128.52.37.6
MUESLI.AI.MIT.EDU       internet address = 128.52.39.7
LIFE.AI.MIT.EDU internet address = 128.52.32.80
BEET-CHEX.AI.MIT.EDU    internet address = 128.52.32.22
MINI-WHEATS.AI.MIT.EDU  internet address = 128.52.54.11
COUNT-CHOCULA.AI.MIT.EDU        internet address = 128.52.38.22
MINTAKA.LCS.MIT.EDU     internet address = 18.26.0.36
```

Ainsi, `muesli.ai.mit.edu` est un serveur de noms pour le domaine `ai.mit.edu` :

```
> server MUESLI.AI.MIT.EDU
Default Server:  MUESLI.AI.MIT.EDU
Address:  128.52.39.7
```

Changeons le type de requête. Nous avons réussi à trouver le serveur de noms, nous allons maintenant demander tout ce que `muesli` sait sur le domaine `prep.ai.mit.edu`.

```
> set q=any
> prep.ai.mit.edu.
Server:  MUESLI.AI.MIT.EDU
Address:  128.52.39.7

prep.ai.mit.edu CPU = dec/decstation-5000.25    OS = unix
```

```
prep.ai.mit.edu
        inet address = 18.159.0.42, protocol = tcp
          ftp  telnet  smtp  finger
prep.ai.mit.edu preference = 1, mail exchanger = gnu-life.ai.mit.edu
prep.ai.mit.edu internet address = 18.159.0.42
ai.mit.edu      nameserver = beet-chex.ai.mit.edu
ai.mit.edu      nameserver = alpha-bits.ai.mit.edu
ai.mit.edu      nameserver = mini-wheats.ai.mit.edu
ai.mit.edu      nameserver = trix.ai.mit.edu
ai.mit.edu      nameserver = muesli.ai.mit.edu
ai.mit.edu      nameserver = count-chocula.ai.mit.edu
ai.mit.edu      nameserver = mintaka.lcs.mit.edu
ai.mit.edu      nameserver = life.ai.mit.edu
gnu-life.ai.mit.edu     internet address = 128.52.32.60
beet-chex.ai.mit.edu    internet address = 128.52.32.22
alpha-bits.ai.mit.edu   internet address = 128.52.32.5
mini-wheats.ai.mit.edu  internet address = 128.52.54.11
trix.ai.mit.edu internet address = 128.52.37.6
muesli.ai.mit.edu       internet address = 128.52.39.7
count-chocula.ai.mit.edu        internet address = 128.52.38.22
mintaka.lcs.mit.edu     internet address = 18.26.0.36
life.ai.mit.edu internet address = 128.52.32.80
```

En commençant à partir de `.`, nous avons successivement trouvé les serveurs de noms des différents niveaux du nom de domaine. Si vous aviez utilisé votre propre serveur DNS à la place de tous ces autres serveurs, votre `named` aurait, bien sûr, caché toutes ces informations et il n'aurait plus eu besoin de les redemander pendant un certain temps.

Si l'on revient a l'analogie avec les arbres, chaque "`.`" dans le nom est un embranchement. Et chaque nom entre deux `.` est une branche de l'arbre.

Grimpons ensemble dans l'arbre en prenant le nom que nous voulons (`prep.ai.mit.edu`). On part de la racine (`.`), on regarde ensuite dans quelle branche grimper, dans notre cas, `edu`. Dès qu'on l'a trouvée, on y grimpe en passant par le serveur qui connaît cette partie du nom. Ensuite, assis sur la branche `edu`, on cherche la branche `mit` (le nom combiné est `mit.edu`), puis la branche `ai.mit.edu`. Maintenant, on est sur le bon serveur, au bon embranchement. La dernière partie est de trouver `prep.ai.mit.edu`, ce qui est très simple. En informatique fondamentale, on appelle `prep` une *feuille* de l'arbre.

Un domaine dont on parle beaucoup moins, mais qui n'en est pas moins important, est `in-addr.arpa`. Ce domaine trouve sa place dans la hiérarchie des noms de domaine comme un domaine "normal". `in-addr.arpa` nous sert à obtenir le nom d'hôte connaissant l'adresse IP d'une machine. Une chose très importante ici est de bien remarquer que les adresses IP sont notées en sens inverse à l'intérieur du domaine `in-addr.arpa`. Si vous avez l'adresse d'une machine : `192.128.52.43`, `named` procède exactement comme dans l'exemple de `prep.ai.mit.edu` : il trouve les serveurs pour `in-addr.arpa.`, trouve les serveurs pour `192.in-addr.arpa.`, trouve les serveurs pour `128.192.in-addr.arpa.`, et finalement trouve les serveurs pour `52.128.192.in-addr.arpa.` . On obtient bien ainsi l'information liée à `43.52.128.192.in-addr.arpa.` Malin, n'est ce pas ? (dites oui). En fait, la résolution de noms inverse est assez difficile à admettre les premières années.

À vrai dire, je vous ai menti. Le service DNS ne marche pas vraiment comme ça. Mais ce que je vous ai dit est suffisamment proche de la réalité.

4.2 Notre propre domaine

Maintenant, nous en sommes à définir notre propre domaine bien à nous. Nous allons créer le domaine `linux.bogus` et y déclarer quelques machines. C'est un nom de domaine totalement factice, afin d'être sûr de ne déranger personne dans le Vaste Monde.

Encore une chose avant de commencer. Tous les caractères ne sont pas admis dans les noms de machines. On ne doit utiliser que les caractères de l'alphabet anglais (a-z), les nombres (0-9) et le tiret "-". Utilisez ces caractères, majuscules et minuscules sont confondues, donc `pat.uio.no` est identique à `Pat.UiO.No`.

En fait, nous avons déjà commencé à créer notre propre domaine avec cette ligne dans `named.conf` :

```
zone "0.0.127.in-addr.arpa" {
        type master;
        file "pz/127.0.0";
};
```

Notez bien l'absence de "`.`" à la fin des noms de domaine de ce fichier. Elle signifie que nous allons définir la zone `0.0.127.in-addr.arpa`, que nous sommes son serveur principal et que tout est stocké dans un fichier appelé `pz/127.0.0`. On a déjà vu ce fichier, il se présente comme ceci :

```
@               IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                                1       ; Serial
                                8H      ; Refresh
                                2H      ; Retry
                                1W      ; Expire
                                1D)     ; Minimum TTL
                        NS      ns.linux.bogus.
1                       PTR     localhost.
```

Notez bien le "`.`" à la fin de tous les noms de domaine complets de ce fichier, contrairement au fichier `named.boot`. Certaines personnes aiment commencer chaque fichier définissant une zone par une directive `$ORIGIN`, mais en fait c'est superflu. L'origine (l'emplacement dans la hiérarchie du service DNS) d'un fichier de zone est indiquée dans la zone section du fichier `named.conf`. Dans notre cas, c'est `0.0.127.in-addr.arpa`.

Ce "fichier de zone" ("zone file"), contient 3 "resource records" (RRs) : un SOA RR, un NS RR et un PTR RR. SOA est l'abréviation de "Start Of Authority" (Origine de l'Autorité). Le "@" est une notation spéciale qui désigne l'origine. Et comme la colonne "domain" de ce fichier donne `0.0.127.in-addr.arpa`, la première ligne signifie donc :

```
0.0.127.IN-ADDR.ARPA. IN SOA ...
```

NS est le "resource records" pour le serveur de noms (NS = Name Server), Il n'y a pas de @ au début de la ligne, il est implicite, puisque la ligne d'avant commence avec un "@". Alors, faites-vous une fleur en omettant ce caractère. Donc, la ligne NS peut aussi s'écrire comme suit :

```
0.0.127.in-addr.arpa.    IN      NS      ns.linux.bogus
```

Elle dit au service DNS quelle machine est le serveur de noms pour le domaine `0.0.127.in-addr.arpa`, c'est `ns.linux.bogus`. `ns` est le nom habituel des serveurs de noms, tout comme `www.` pour les serveurs Web, mais c'est simplement une habitude, on peut choisir n'importe quel nom.

Et finalement le PTR dit que l'adresse 1 dans le sous réseau `0.0.127.in-addr.arpa`, donc 127.0.0.1 est appelé `localhost`.

Le champ SOA est le préambule de *tous* les fichiers de zone, et il doit y en avoir exactement un dans chaque fichier de zone. Ce champ SOA décrit la zone, son origine (une machine appelée `ns.linux.bogus`), qui est responsable de son contenu (`hostmaster@linux.bogus`, vous devriez mettre votre adresse email à cet endroit), de quelle version du fichier de zone il s'agit (serial : 1), et quelques autres paramètres pour le cache et les serveurs DNS secondaires. Quant aux champs restants (*refresh*, *retry*, *expire* et *minimum*) utilisez les valeurs données dans ce HOWTO et tout se passera certainement très bien.

Maintenant, relancez votre `named` (avec la commande `ndc restart`) et utilisez `nslookup` pour regarder le résultat :

```
$ nslookup

Default Server:  localhost
Address:  127.0.0.1

> 127.0.0.1
Server:  localhost
Address:  127.0.0.1

Name:    localhost
Address:  127.0.0.1
```

Tout va bien, on arrive à obtenir `localhost` à partir de 127.0.0.1. Maintenant, pour le sujet qui nous préoccupe, le domaine `linux.bogus`, insérez une nouvelle zone dans le fichier `named.conf` :

```
zone "linux.bogus" {
        notify no;
        type master;
        file "pz/linux.bogus";
};
```

Notez qu'encore une fois il n'y a pas de "`.`" à la fin des noms de domaine dans le fichier `named.conf`.

Dans le fichier de zone `linux.bogus`, nous allons mettre quelques données totalement factices :

```
;
; Zone file for linux.bogus
;
; The full zone file
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; serial, todays date + todays serial #
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        1W              ; expire, seconds
                        1D )            ; minimum, seconds
;
                NS      ns              ; Inet Address of name server
                MX      10 mail.linux.bogus     ; Primary Mail Exchanger
                MX      20 mail.friend.bogus.   ; Secondary Mail Exchanger
;
localhost       A       127.0.0.1
ns              A       192.168.196.2
mail            A       192.168.196.4
```

Il y a deux choses à noter à propos du champ SOA. `ns.linux.bogus` *doit absolument* être une vraie machine possédant un champ A. Il n'est pas légal d'avoir un champ CNAME pour la machine mentionnée dans le champ SOA. Il n'est pas nécessaire que son nom soit "ns", ce peut être tout autre nom valide. La deuxième chose à noter c'est que `hostmaster.linux.bogus` doit se lire comme `hostmaster@linux.bogus`. Ce doit être un alias de mail, ou une véritable boîte aux lettres électronique, et la personne qui maintient le DNS doit la lire régulièrement. Tous les mails concernant l'administration du domaine seront envoyés à cette adresse. Il n'est pas obligatoire que le nom soit "hostmaster", vous pouvez mettre votre adresse e-mail personnelle, mais il serait bon que l'adresse "hostmaster" fonctionne aussi.

Il y a un nouveau RR (Resource Record) dans ce fichier, c'est le MX, pour Mail eXchanger. Il indique aux systèmes de gestion du courrier électronique à quelle machine envoyer le mail adressé à `someone@linux.bogus`, dans notre cas à `mail.linux.bogus` ou `mail.friend.bogus`. Le nombre devant chaque machine est sa priorité vis-à-vis du champ MX, le RR avec le numéro le plus faible (10) correspond à la machine à laquelle le courrier doit être adressé en priorité. En cas d'échec, il peut être adressé à la machine qui a le numéro de priorité immédiatement supérieur, c'est-à-dire `mail.friend.bogus` qui a une priorité de 20 dans notre cas.

Relancez `named` en tapant `ndc restart`. Examinons le résultat avec nslookup :

```
$ nslookup
> set q=any
> linux.bogus
Server:  localhost
Address:  127.0.0.1

linux.bogus
        origin = ns.linux.bogus
        mail addr = hostmaster.linux.bogus
        serial = 199802151
        refresh = 28800 (8 hours)
        retry   = 7200 (2 hours)
        expire  = 604800 (7 days)
        minimum ttl = 86400 (1 day)
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     preference = 10, mail exchanger = mail.linux.bogus.linux.bogus
linux.bogus     preference = 20, mail exchanger = mail.friend.bogus
linux.bogus     nameserver = ns.linux.bogus
ns.linux.bogus  internet address = 192.168.196.2
mail.linux.bogus        internet address = 192.168.196.4
```

Un examen approfondi vous montrera qu'il y a un bug. En effet, la ligne

```
linux.bogus preference = 10, mail exchanger = mail.linux.bogus.linux.bogus
```

est entièrement fausse. Il devrait y avoir

```
linux.bogus preference = 10, mail exchanger = mail.linux.bogus
```

J'ai fait cette erreur délibérément, pour voir si vous suiviez :-) En regardant dans le fichier de zone, nous trouvons que dans la ligne

```
@ MX 10 mail.linux.bogus ; Primary Mail Exchanger
```

il manque un point. Ou il y a un "linux.bogus" de trop. Si, dans un fichier de zone, un nom de machine ne se termine pas par un point, l'origine est ajoutée au nom de la machine. Ainsi, une des deux formes :

```
                MX      10 mail.linux.bogus.    ; Primary Mail Exchanger
```

ou

```
                MX      10 mail                 ; Primary Mail Exchanger
```

est correcte. Je préfère la deuxième forme parce qu'il y a moins de caractères à taper. Certains approuveront, d'autres non. Dans un fichier de zone, le nom de domaine doit ou bien être écrit et terminé par un point, ou bien ne pas être inclus du tout. Dans le dernier cas, le nom de domaine par défaut est l'origine.

Il faut que j'insiste sur le point suivant : dans le fichier `named.conf`, il ne doit *pas* y avoir de "`.`" après les noms de domaines. Vous ne pouvez pas vous imaginer les ravages qui ont été causés pas des "`.`" en trop ou en moins.

Cela étant dit, voici le nouveau fichier de zone, avec quelques informations supplémentaires :

```
;
; Zone file for linux.bogus
;
; The full zone file
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; serial, todays date + todays serial #
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        1W              ; expire, seconds
                        1D )            ; minimum, seconds
;
                TXT     "Linux.Bogus, your DNS consultants"
                NS      ns              ; Inet Address of name server
                NS      ns.friend.bogus.
                MX      10 mail         ; Primary Mail Exchanger
                MX      20 mail.friend.bogus. ; Secondary Mail Exchanger

localhost       A       127.0.0.1

gw              A       192.168.196.1
                HINFO   "Cisco" "IOS"
                TXT     "The router"

ns              A       192.168.196.2
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "Pentium" "Linux 2.0"
www             CNAME   ns

donald          A       192.168.196.3
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "i486"  "Linux 2.0"
                TXT     "DEK"

mail            A       192.168.196.4
```

```
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "386sx" "Linux 1.2"

ftp             A       192.168.196.5
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "P6" "Linux 2.1.86"
```

Il y a un certain nombre de nouveaux RR que nous allons passer en revue : HINFO (Host INFOrmation), qui est en deux parties, et c'est une bonne habitude à prendre que d'encadrer chacune de guillemets. La première partie est la description matérielle ou le type de processeur de la machine tandis que la deuxième partie décrit le logiciel utilisé ou le système d'exploitation de la machine. ns a pour processeur un Pentium et tourne sous Linux 2.0. Le champ CNAME (Canonical NAME) sert à donner plusieurs noms à la même machine. Par conséquent, `www` est un alias de `ns`.

L'utilisation des champs CNAME est assez controversée. Mais il est sage de suivre la règle selon laquelle un champ MX, CNAME ou SOA ne doit *jamais* se référer à un champ CNAME, toujours se référer à un champ A, il est donc préférable de ne pas avoir :

```
foobar          CNAME   www                     ; NON !
```

En revanche, ceci est correct :

```
foobar          CNAME   ns                      ; Oui !
```

Il est aussi important de noter qu'un CNAME n'est pas un nom d'hôte légal pour une adresse de courrier électronique. `webmaster@www.linux.bogus` est une adresse de mail illégale avec la configuration ci-dessus. Vous pouvez être sûrs qu'il y a un certain nombre d'administrateurs système dans le Vaste Monde qui sont très à cheval sur cette règle, même si avec un CNAME ça marche pour vous. Une façon de contourner le problème est d'utiliser des champs A (et peut-être d'autres, comme un champ MX par exemple) à la place :

```
www             A       192.168.196.2
```

Un certain nombre de gourous-du-bind recommandent de ne *pas* utiliser de CNAME. Mais les discussions sur le pour et le contre sortent du cadre de ce HOWTO.

Mais comme vous le voyez, ce HowTo ainsi que beaucoup de serveurs ne suivent pas cette règle.

Chargez la nouvelle base de données en lançant `ndc reload`, ce qui forcera `named` à relire ses fichiers de configuration.

```
$ nslookup
Default Server:  localhost
Address:  127.0.0.1

> ls -d linux.bogus
```

Ceci veut dire que l'on souhaite que tous les champs soient affichés.

```
[localhost]
$ORIGIN linux.bogus.
@                       1D IN SOA       ns hostmaster (
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
```

```
                                    1W              ; expiry
                                    1D )            ; minimum

                        1D IN NS        ns
                        1D IN NS        ns.friend.bogus.
                        1D IN TXT       "Linux.Bogus, your DNS consultants"
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
gw                      1D IN A         192.168.196.1
                        1D IN HINFO     "Cisco" "IOS"
                        1D IN TXT       "The router"
mail                    1D IN A         192.168.196.4
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "386sx" "Linux 1.0.9"
localhost               1D IN A         127.0.0.1
www                     1D IN CNAME     ns
donald                  1D IN A         192.168.196.3
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "i486" "Linux 1.2"
                        1D IN TXT       "DEK"
ftp                     1D IN A         192.168.196.5
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "P6" "Linux 1.3.59"
ns                      1D IN A         192.168.196.2
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "Pentium" "Linux 1.2"
```

Tout va bien. Regardons ce qu'il dit pour `www` tout seul :

```
> set q=any
> www.linux.bogus.
Server:  localhost
Address:  127.0.0.1

www.linux.bogus canonical name = ns.linux.bogus
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     nameserver = ns.friend.bogus
ns.linux.bogus  internet address = 192.168.196.2
```

En d'autres termes, le vrai nom de `www.linux.bogus` est `ns.linux.bogus`, et vous avez en plus quelques informations à propos de ns, en fait, suffisamment pour vous y connecter si vous étiez un programme.

Bon, on a fait la moitié du boulot.

4.3 La zone inversée

Ça y est, les programmes peuvent convertir les noms de `linux.bogus` en adresses auxquelles ils peuvent se connecter. Maintenant, on a besoin d'une zone inversée pour que l'on puisse retrouver le DNS à partir de l'adresse. Ce nom est utilisé par différents types de serveurs (FTP, IRC, WWW et autres) pour décider s'ils vont discuter avec vous ou non, et s'ils le font, quelle priorité ils vont vous donner. Pour un accès complet aux services sur Internet, la zone inversée est indispensable.

Mettez ça dans votre `named.conf`

```
zone "196.168.192.in-addr.arpa" {
        notify no;
        type master;
        file "pz/192.168.196";
};
```

C'est exactement comme pour le `0.0.127.in-addr.arpa` et le contenu est similaire :

```
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151 ; Serial, todays date + todays serial
                        8H      ; Refresh
                        2H      ; Retry
                        1W      ; Expire
                        1D)     ; Minimum TTL
                NS      ns.linux.bogus.

1               PTR     gw.linux.bogus.
2               PTR     ns.linux.bogus.
3               PTR     donald.linux.bogus.
4               PTR     mail.linux.bogus.
5               PTR     ftp.linux.bogus.
```

Redémarrez votre `named` (`ndc restart`) et examinez votre travail avec nslookup :

```
> 192.168.196.4
Server:  localhost
Address:  127.0.0.1

Name:    mail.linux.bogus
Address:  192.168.196.4
```

On dirait que c'est bon, on va regarder en détails pour s'en assurer :

```
> ls -d 196.168.192.in-addr.arpa
[localhost]
$ORIGIN 196.168.192.in-addr.arpa.
@                       1D IN SOA       ns.linux.bogus. hostmaster.linux.bogus. (
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

                        1D IN NS        ns.linux.bogus.
1                       1D IN PTR       gw.linux.bogus.
2                       1D IN PTR       ns.linux.bogus.
3                       1D IN PTR       donald.linux.bogus.
4                       1D IN PTR       mail.linux.bogus.
5                       1D IN PTR       ftp.linux.bogus.
@                       1D IN SOA       ns.linux.bogus. hostmaster.linux.bogus. (
```

```
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
                                        1W              ; expiry
                                        1D )            ; minimum
```

Pas mal ! Si ce que vous donne nslookup ne ressemble pas a ça, allez a la pêche aux messages d'erreur dans votre syslog. J'ai expliqué comment faire au tout début du chapitre.

4.4 Précautions d'usage

Je devrais maintenant faire quelques remarques. Les adresses IP utilisées dans les exemples précédents sont prises dans le bloc des “réseaux privés”, c'est à dire des adresses qui ne doivent pas être utilisées publiquement sur Internet. Donc, il est sage de les avoir utilisées dans un exemple d'un HowTo. La deuxième chose est la ligne `notify no ;`. Elle demande à `named` de ne pas informer ses serveur secondaires (les esclaves) quand l'un de ses fichiers de zone a été mis à jour. Depuis Bind-8 `named` peut informer les autres serveurs listés dans ses champs NS dans le fichier zone, quand une zone est mise a jour. C'est pratique pour une utilisation normale, mais pour des expériences privées cette fonctionnalité doit être mise hors service, on ne va quand même pas polluer Internet avec nos expériences, non ?

Bien sûr, ce domaine est très factice, tout comme le sont ses adresses. C'est peut-être un peu déroutant pour vous. Un vrai exemple tiré d'un vrai domaine vous attend au grand chapitre suivant.

4.5 Pourquoi est-ce que les lookup inversés ne marchent pas ?

Il y a quelques trucs qui sont normalement évités avec les lookups qui arrivent souvent quand on met en place des zones inversés. Avant de continuer, vous avez besoin d'avoir des lookups qui marchent sur vos propres serveurs de noms. Si ce n'est pas le cas, revenez en arrière et réparez-le avant de continuer.

Je parlerais des deux problèmes de lookups inversés qui sont vu de l'extérieur de votre réseau :

4.5.1 La zone inverse n'est pas déléguée.

Quand vous demandez à un fournisseur d'accès quelques adresses IP ainsi qu'un nom de domaine, le nom de domaine vous est normalement délégué. La délégation consiste en un champ `NS` qui vous aide a passer d'un serveur à l'autre comme je l'ai expliqué dans le brin de théorie qui précède. Vous l'avez lu, n'est-ce pas ? Si votre zone inversée ne marche pas, retournez y et lisez-le. Maintenant.

La zone inversée a elle aussi besoin d'être déléguée. Si vous avez le réseau `192.168.196` avec le domaine `linux.bogus` de votre fournisseur, il devra mettre des champs `NS` pour votre zone inversée aussi bien que pour votre zone directe. Si vous remontez la chaîne à partir de `in-addr.arpa` vous trouverez un trou quelque part. Très certainement au niveau de votre fournisseur. Après avoir trouvé le trou dans la chaîne, contactez votre fournisseur et demandez-lui de corriger l'erreur.

4.5.2 Vous avez un sous-réseau sans classe

C'est un sujet plutôt pointu, mais les sous réseaux sans classe sont très répandus de nos jours et vous en aurez très certainement un si vous n'êtes pas une entreprise assez grande.

Un sous-réseau sans classe est ce qui sauve Internet de nos jours. Il y a quelques années, il y avait vraiment beaucoup de discussions sur la raréfaction des adresses IP. Les personnes intelligentes de l'IETF (Internet Engineering Task Force, ceux qui maintiennent Internet en état de marche) se sont penchées sur cet épineux problème et ont trouvé une solution. A un certain prix. Le prix est que vous aurez moins qu'un sous réseau de classe “C” et que certaines choses ne marcheront certainement plus. Allez voir *Ask Mr DNS* (c'est en anglais) pour plus d'explications.

Vous l'avez lu ? Comme je ne vais pas l'expliquer, s'il vous plaît, allez le lire.

La première partie du problème est que votre FAI doit comprendre la technique décrite par *Mr DNS*. Tous les petits FAI ne le comprennent pas. S'ils n'ont pas bien compris, vous allez avoir à leur expliquer et à insister. Mais assurez-vous de comprendre vous-même en premier lieu ;-). Ils mettrons ensuite une jolie zone inversée sur leurs serveurs que vous pourrez examiner pour savoir si elle est correcte avec `nslookup`.

La deuxième et dernière partie du problème est que vous devez en comprendre la technique. Si vous n'êtes pas certain, revenez en arrière et relisez ce document. Ensuite, vous pourrez mettre en place une zone inversée sans classe comme le décrit *Mr DNS*.

Il y a une autre difficulté qui pointe son nez ici. Les vieux résolveurs *ne seront pas* capable de suivre les champs `CNAME` dans la chaîne de résolution et n'arriveront pas a résoudre l'IP de votre machine. Cela peut entraîner l'assignation d'une mauvaise classe, la non-résolution ou quelque chose dans ce goût-là. Si vous butez sur ce genre de problème, la seule solution (que je connaisse) est de demander à votre FAI d'insérer vos champs `PTR` dans ses fichiers de zone sans classe plutôt que des champs `CNAME`.

Certains FAI vous proposeront d'autre méthodes pour gérer cela, comme des formulaires web où vous pourrez entrer vos zones inversées, ou d'autre systèmes automatisés.

5 Un exemple tiré d'un domaine réel

Où nous allons enfin voir de *vrais* fichiers de zone

Certains utilisateurs ont suggéré que je mette un vrai exemple d'un domaine qui marche dans la réalité car mon explication sur la différence entre un vrai domaine et l'exemple bidon ci-dessus n'était pas très claire.

J'utilise cet exemple avec la permission de David Bullock de LAND-5. Ces fichiers étaient à jour le 24 Septembre 96, et ont été modifiée pour être utilisés avec les restrictions de bind 8 et quelques extensions de mon cru. Par conséquent, ils peuvent donc différer de ce que vous pouvez trouver en questionnant les serveurs de nom de LAND-5 aujourd'hui.

Voici les sections pour les deux zones inversées nécessaires : le réseau `127.0.0`, ainsi que le sous-réseau LAND-5 `206.6.177`. Et une ligne primary pour la forward zone `land-5.com`. Notez aussi qu'au lieu de mettre les fichiers dans le répertoire `pz` comme dans ce HowTo, il les met dans le répertoire `zone`.

5.1 /etc/named.conf (ou /var/named/named.conf)

```
// Boot file for LAND-5 name server

options {
        directory "/var/named";
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.in-addr.arpa" {
```

```
        type master;
        file "zone/127.0.0";
};

zone "land-5.com" {
        type master;
        file "zone/land-5.com";
};

zone "177.6.206.in-addr.arpa" {
        type master;
        file "zone/206.6.177";
};
```

Si vous mettez ça dans votre `named.conf` pour jouer avec, **PAR PITIÉ** mettez aussi le "`notify no ;`" dans les zones des deux `land-5.com` pour éviter les accidents.

5.2 /var/named/root.hints

Souvenez-vous que le contenu de ce fichier peut changer, et celui donné ici est assez vieux. Vous feriez mieux d'utiliser un fichier plus récent, produit par le programme `dig`.

```
; <<>> DiG 8.1 <<>> @A.ROOT-SERVERS.NET.
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10
;; flags: qr aa rd; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 13
;; QUERY SECTION:
;;      ., type = NS, class = IN

;; ANSWER SECTION:
.                       6D IN NS        G.ROOT-SERVERS.NET.
.                       6D IN NS        J.ROOT-SERVERS.NET.
.                       6D IN NS        K.ROOT-SERVERS.NET.
.                       6D IN NS        L.ROOT-SERVERS.NET.
.                       6D IN NS        M.ROOT-SERVERS.NET.
.                       6D IN NS        A.ROOT-SERVERS.NET.
.                       6D IN NS        H.ROOT-SERVERS.NET.
.                       6D IN NS        B.ROOT-SERVERS.NET.
.                       6D IN NS        C.ROOT-SERVERS.NET.
.                       6D IN NS        D.ROOT-SERVERS.NET.
.                       6D IN NS        E.ROOT-SERVERS.NET.
.                       6D IN NS        I.ROOT-SERVERS.NET.
.                       6D IN NS        F.ROOT-SERVERS.NET.

;; ADDITIONAL SECTION:
G.ROOT-SERVERS.NET.     5w6d16h IN A    192.112.36.4
J.ROOT-SERVERS.NET.     5w6d16h IN A    198.41.0.10
K.ROOT-SERVERS.NET.     5w6d16h IN A    193.0.14.129
```

```
L.ROOT-SERVERS.NET.     5w6d16h IN A    198.32.64.12
M.ROOT-SERVERS.NET.     5w6d16h IN A    202.12.27.33
A.ROOT-SERVERS.NET.     5w6d16h IN A    198.41.0.4
H.ROOT-SERVERS.NET.     5w6d16h IN A    128.63.2.53
B.ROOT-SERVERS.NET.     5w6d16h IN A    128.9.0.107
C.ROOT-SERVERS.NET.     5w6d16h IN A    192.33.4.12
D.ROOT-SERVERS.NET.     5w6d16h IN A    128.8.10.90
E.ROOT-SERVERS.NET.     5w6d16h IN A    192.203.230.10
I.ROOT-SERVERS.NET.     5w6d16h IN A    192.36.148.17
F.ROOT-SERVERS.NET.     5w6d16h IN A    192.5.5.241

;; Total query time: 215 msec
;; FROM: roke.uio.no to SERVER: A.ROOT-SERVERS.NET.  198.41.0.4
;; WHEN: Sun Feb 15 01:22:51 1998
;; MSG SIZE  sent: 17  rcvd: 436
```

5.3 /var/named/zone/127.0.0

Nous y avons mis juste l'essentiel, le champ SOA obligatoire, et un champ qui établit la correspondance entre 127.0.0.1 et `localhost`. Ils sont tous les deux indispensables. Rien d'autre ne doit figurer dans ce fichier. Il ne sera probablement jamais nécessaire de le mettre à jour, à moins que l'adresse du serveur de noms ou de hostmaster ne change.

```
@               IN      SOA     land-5.com. root.land-5.com. (
                                199609203       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; Minimum TTL
                        NS      land-5.com.

1                       PTR     localhost.
```

5.4 /var/named/zone/land-5.com

Nous trouvons ici le classique et obligatoire champ SOA ainsi que les champs NS. Nous pouvons voir qu'il y a un serveur de noms secondaire `ns2.psi.net`. C'est comme ça que tout le monde devrait faire : *toujours* avoir un serveur secondaire sur un site distant pour faire des sauvegardes. Nous voyons également que le serveur primaire est `land-5`, qui assure tous les services, et que l'administrateur a utilisé des CNAME pour faire ça (il aurait pu utiliser des champs A).

Comme vous pouvez voir d'après le champ SOA, le fichier de zone a son origine en `land-5.com`, la personne à contacter est `root@land-5.com`. `hostmaster` est une autre adresse souvent utilisée pour la personne à contacter. Le numéro de série est au format obligatoire aaaammjj, avec le numéro de série dans la journée ajouté à la fin ; il s'agit certainement de la sixième version du fichier de zone pour la journée du 20 septembre 1996. N'oubliez-pas que le numéro de série doit *obligatoirement* augmenter avec le temps, ici il n'y a qu'un chiffre pour le numéro de série dans la journée, si bien qu'après 9 modifications il faudra attendre le lendemain pour modifier le fichier à nouveau. On peut aussi utiliser deux chiffres au lieu d'un seul.

```
@       IN      SOA     land-5.com. root.land-5.com. (
                        199609206       ; serial, todays date + todays serial #
```

```
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        1W              ; expire, seconds
                        1D )            ; minimum, seconds
                NS      land-5.com.
                NS      ns2.psi.net.
                MX      10 land-5.com.  ; Primary Mail Exchanger
                TXT     "LAND-5 Corporation"

localhost       A       127.0.0.1

router          A       206.6.177.1

land-5.com.     A       206.6.177.2
ns              A       206.6.177.3
www             A       207.159.141.192

ftp             CNAME   land-5.com.
mail            CNAME   land-5.com.
news            CNAME   land-5.com.

funn            A       206.6.177.2

;
;       Workstations
;
ws-177200       A       206.6.177.200
                MX      10 land-5.com.   ; Primary Mail Host
ws-177201       A       206.6.177.201
                MX      10 land-5.com.   ; Primary Mail Host
ws-177202       A       206.6.177.202
                MX      10 land-5.com.   ; Primary Mail Host
ws-177203       A       206.6.177.203
                MX      10 land-5.com.   ; Primary Mail Host
ws-177204       A       206.6.177.204
                MX      10 land-5.com.   ; Primary Mail Host
ws-177205       A       206.6.177.205
                MX      10 land-5.com.   ; Primary Mail Host
; {Ici 245 lignes ont été effacées}
ws-177250       A       206.6.177.250
                MX      10 land-5.com.   ; Primary Mail Host
ws-177251       A       206.6.177.251
                MX      10 land-5.com.   ; Primary Mail Host
ws-177252       A       206.6.177.252
                MX      10 land-5.com.   ; Primary Mail Host
ws-177253       A       206.6.177.253
                MX      10 land-5.com.   ; Primary Mail Host
ws-177254       A       206.6.177.254
                MX      10 land-5.com.   ; Primary Mail Host
```

Si vous examinez le serveur de noms de land-5, vous allez voir que les noms sont de la forme `ws_`*nombre*.

Depuis les dernières versions de `bind 4`, `named` fait plus attention aux caractères placés dans les noms de domaines. Cela ne marcherait pas du tout avec bind-8, c'est pour ça que j'ai remplacé les "_" (souligné) par des "-" (tiret) pour l'exemple dans ce HowTo.

Une autre chose qu'il faut noter est que les stations de travail n'ont pas de nom personnel, mais plutôt un préfixe suivit des deux derniers morceaux de leur adresse IP. Utiliser une telle convention simplifie grandement la maintenance, mais c'est un peu impersonnel, et ça peut agacer vos clients.

Nous voyons aussi que `funn.land-5.com` est un alias pour `land-5.com`, mais en utilisant un enregistrement `A`, pas un `CNAME`. C'est une bonne chose comme on l'a noté plus haut.

5.5 /var/named/zone/206.6.177

Les commentaires se trouvent juste après le fichier.

```
@               IN      SOA     land-5.com. root.land-5.com. (
                                199609206       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; Minimum TTL
                        NS      land-5.com.
                        NS      ns2.psi.net.
;
;       Servers
;
1       PTR     router.land-5.com.
2       PTR     land-5.com.
2       PTR     funn.land-5.com.
;
;       Workstations
;
200     PTR     ws-177200.land-5.com.
201     PTR     ws-177201.land-5.com.
202     PTR     ws-177202.land-5.com.
203     PTR     ws-177203.land-5.com.
204     PTR     ws-177204.land-5.com.
205     PTR     ws-177205.land-5.com.
; {Ici 245 lignes ont été effacées}
250     PTR     ws-177250.land-5.com.
251     PTR     ws-177251.land-5.com.
252     PTR     ws-177252.land-5.com.
253     PTR     ws-177253.land-5.com.
254     PTR     ws-177254.land-5.com.
```

La zone inverse est la partie de la configuration qui semble poser le plus de problèmes. Elle est utilisée pour trouver le nom d'hôte d'une machine, connaissant son adresse IP. Exemple : vous êtes un serveur IRC et vous acceptez des connexions provenant de clients IRC. Cependant, comme vous êtes un serveur IRC norvégien, vous ne voulez accepter que les connexions venant de Norvège ou des autres pays scandinaves. Ainsi, lorsqu'un client se connecte chez vous, la bibliothèque C peut vous dire quelle est l'adresse IP du client, puisque cette dernière se trouve dans tous les paquets qui traversent le réseau. Ensuite, vous pouvez appeler une fonction connue sous le nom de `gethostbyaddr` qui va rechercher le nom d'une machine connaissant son adresse IP. gethostbyaddr va poser la question à un serveur de noms, qui va alors faire une recherche de la machine dans le DNS. Supposons que la connexion du client se fasse depuis ws_177200.land-5.com. L'adresse IP que la bibliothèque C fournit au serveur IRC est 206.6.177.200. Pour retrouver le nom de cette machine, il nous faut trouver 200.177.6.206.in-addr.arpa. Le serveur de noms va donc d'abord trouver les serveurs arpa., puis les serveurs in-addr.arpa., poursuivre la recherche inverse par 206, puis 6 et finalement trouver le serveur pour la zone 177.6.206.in-addr.arpa à LAND-5. C'est ce dernier qui lui dira que pour 200.177.6.206.in-addr.arpa nous avons un champ "PTR ws_177200.land-5.com", ce qui veut dire que le nom qui va avec 206.6.177.200 est ws_177200.land-5.com. Tout comme l'explication de la résolution de prep.ai.mit.edu, ce scénario est un peu idéalisé.

Revenons à l'exemple du serveur IRC. Le serveur n'accepte que les connexions venant des pays scandinaves, c'est-à-dire *.no, *.se, *.dk. Le nom ws_177200.land-5.com ne correspond évidemment pas, et le serveur va donc refuser la connexion. Si il n'existait *pas* de résolution inverse de 206.2.177.200 au travers de la zone in-addr.arpa, le serveur aurait été tout à fait incapable de trouver le nom, et aurait dû se contenter de comparer 206.6.177.200 à *.no, *.se et *.dk, dont aucun ne correspond.

Certaines personnes vous diront que la résolution de noms inverse n'est importante que pour les serveurs, ou pas importante du tout. Pas tant que ça : beaucoup de serveurs ftp, news, irc ou même certains http (Web) n'acceptent *pas* les connexions venant de machines dont ils ne peuvent retrouver le nom. C'est pourquoi la résolution de noms inverse pour les machines est *obligatoire*.

6 Maintenance

Garder votre DNS en état de marche

En plus des tâches normales, il y a une tâche de maintenance spéciale à effectuer sur les serveurs de nom. Il s'agit de garder le fichier root.hints à jour. La façon la plus simple de le faire est d'utiliser dig. Lancez d'abord dig sans argument, vous obtiendrez le fichier root.cache de votre propre serveur. Posez alors la même question à un des serveurs de cette liste avec la commande dig @rootserver. Vous remarquerez que ce que vous obtenez ressemble énormément à un fichier root.hints, avec quelques chiffres en plus. Ces chiffres supplémentaires sont inoffensifs. Sauvez-le dans un fichier (dig . @e.root-servers.net >root.hints.new) et remplacez l'ancien fichier root.hints avec.

N'oubliez pas de relancer named après avoir remplacé ce fichier.

Al Longyear m'a envoyé ce script, qui peut être lancé automatiquement pour mettre à jour named.hints. Lancez-le automatiquement à partir de la crontab et vous pourrez oublier qu'il existe. Ce script suppose que l'alias de mail 'hostmaster' existe. Il faudra sans doute modifier ce fichier pour qu'il fonctionne chez vous.

```
#!/bin/sh
#
# Met a jours les informations du cache du serveur de noms chaque mois.
# Ce script est lancé automatiquement par un cron.
#
# Original par Al Longyear
# Mis a jour pour Bind 8 par Nicolai Langfeldt
# Plusieurs erreurs découvertes par David A. Ranch
# Test avec un ping suggéré par Martin Foster
#
(
 echo "To: hostmaster <hostmaster>"
 echo "From: system <root>"
 echo "Subject: Mise a jour automatique du fichier root.hints"
```

```
 echo

 PATH=/sbin:/usr/sbin:/bin:/usr/bin:
 export PATH
 cd /var/named

 # Sommes nous connectés ? Pingons un serveur de notre FAI
 case `ping -qnc some.machine.net` in
   *'100% packet loss'*)
        echo "PAS de connexion réseau. root.hints NON mis à jour"
        echo
        exit 0
        ;;
 esac

 dig @rs.internic.net . ns >root.hints.new 2>&1

 case `cat root.hints.new` in
   *NOERROR*)
        # Ca a marché
        :;;
   *)
        echo "La mise a jour de root.hints a ECHOUE."
        echo "Voici la sortie de dig :"
        echo
        cat root.hints.new
        exit 0
        ;;
 esac

 echo "Le fichier root.hints a été mis a jour et contient les informations suivantes :"
 echo
 cat root.hints.new

 chown root.root root.hints.new
 chmod 444 root.hints.new
 rm -f root.hints.old
 mv root.hints root.hints.old
 mv root.hints.new root.hints
 ndc restart
 echo
 echo "Le serveur de noms a été redémarré, de cette manière, la mise a jour est complète."
 echo "L'ancien root.hints s'appelle maintenant /var/named/root.hints.old."
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

Certains d'entre vous ont remarqués que le fichier `root.hints` est aussi disponible via ftp depuis l'Internic. S'il vous plaît, n'utilisez pas le ftp pour mettre à jour le `root.hints`, la méthode ci dessus est bien meilleur du point de vue de la nettiquette et de l'Internic.

7 Passer de la version 4 à la version 8

Cette section était au départ sur l'utilisation de bind-8 écrite par David E. Smith (dave@bureau42.ml.org). Je l'ai éditée pour refléter le nouveau nom de la section.

Il n'y a pas grand chose à faire, sinon, utiliser `named.conf` au lieu du `named.boot`, or bind-8 est distribué avec un script perl pour convertir un `named.boot` en `named.conf`. Exemple de `named.boot` (vieux) pour un serveur qui ne sert que de cache :

```
directory /var/named
cache   .                                               root.hints
primary 0.0.127.IN-ADDR.ARPA                            127.0.0.zone
primary localhost                                       localhost.zone
```

Depuis la ligne de commande, et depuis le répertoire `bind8/src/bin/named` (au cas où vous avez récupéré les sources ; si vous avez eu un paquetage binaire, le script se balade certainement dans le coin), tapez :

```
./named-bootconf.pl < named.boot > named.conf
```

qui crée un nouveau `named.conf` :

```
// generated by named-bootconf.pl

options {
        directory "/var/named";
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.IN-ADDR.ARPA" {
        type master;
        file "127.0.0.zone";
};

zone "localhost" {
        type master;
        file "localhost.zone";
};
```

Ça marche pour tout ce qui pouvait aller dans un `named.boot`, mais, il ne met pas toutes les nouveautés que bind-8 permet. Voici une version plus complète d'un `named.conf` qui fait la même chose, mais d'une façon plus efficace :

```
// Voici le fichier de configuration de named (pour BIND 8.1 et ultérieur).
// Il devrait normalement être installé dans /etc/named.conf.
// Le seul changement fait dans le named.conf d'origine (a part ce commentaire
// :) est que la ligne directory a été décommentée, car j'ai déjà les fichiers
// de zone dans /var/named.
```

```
options {
        directory "/var/named";
        datasize 20M;
};

zone "localhost" IN {
        type master;
        file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "127.0.0.zone";
};

zone "." IN {
        type hint;
        file "root.hints";
};
```

Dans le répertoire `bind8/src/bin/named/test` de la distribution de bind8, vous trouverez tout ça, ainsi que des fichiers de zone que la majorité peuvent prendre et utiliser instantanément.

Les formats des fichiers de zone et du `root.hints` sont les mêmes, tout comme les commandes qui les mettent à jour.

8 Questions et Réponses

Dans cette section, je passe en revue quelques-unes des questions les plus fréquemment posées à propos du DNS et de ce HOWTO. Et je donne même les réponses ;-) Merci de bien lire cette section avant de m'écrire.

1. Mon `named` me réclame un fichier `named.boot`

   Vous vous êtes trompés de HowTo. allez voir l'ancienne version de ce HowTo, celle qui parle de bind 4, à *www.math.uio.no/~janl/DNS/*

2. Question : Comment utiliser un DNS si l'on se trouve derrière un firewall ?

   Voici un indice : `forward only ;`. Vous aurez probablement aussi besoin de mettre :

```
query-source port 53;
```

   dans la partie “options” de votre `named.conf` comme l'exemple 3 le suggère 3 (serveur qui ne fait que du cache).

3. Question : Comment dire à un DNS qu'il doit faire une rotation entre un certain nombre d'adresses pour un service donné, par exemple si l'on veut obtenir équilibrer la charge de `www.busy.com` entre plusieurs machines ?

   Créez plusieurs champs **A** pour www.busy.com et utilisez bind 4.9.3 ou une version plus récente, qui supporte les réponses à scrutation circulaire. Cela ne marchera *pas* avec des versions de bind antérieures.

4. Je veux mettre en place un serveur DNS sur un Intranet (fermé). Comment faire ?

   Effacez rageusement le fichier `root.hints` et créez seulement les fichiers de zone. Cela veut aussi dire que vous n'aurez pas à créer des nouveaux fichiers hints tout le temps.

5. Comment mettre en place un serveur secondaire ?

   Si le serveur primaire a pour adresse 127.0.0.1, mettez une ligne comme celle-ci dans le fichier `named`.conf du serveur secondaire :

```
zone "linux.bogus" {
      type slave;
      file "sz/linux.bogus";
      masters { 127.0.0.1; };
};
```

   Vous pouvez mettre plusieurs serveurs maîtres, ajoutez les sur la ligne masters en les séparant par un " ;" (point-virgule)

6. Je veux faire tourner bind lorsque je suis déconnecté du réseau

   Il y a trois trucs a savoir :

   – J'ai reçu le mail suivant de Ian Clark <ic@deakin.edu.au>, où il explique comment il fait ça :

```
Ici, je fais tourner named sur la machine qui fait du "Masquerading". J'ai
deux fichiers root.cache, un qui s'appelle root.cache.real et qui contient les
vrais noms des serveurs root, et l'autre qui s'appelle root.cache.fake qui
contient ceci~:

--------------
; root.hints.fake
; Ce fichier ne contient pas d'informations
--------------

Quand je me déconnecte, je copie le fichier root.hints.fake vers root.hints et
je relance named.

Quand je me connecte, je copie root.hints.real et je relance named.

Ces deux manoeuvres sont faites, respectivement, à partir de ip-down et ip-up.

Lorsque je suis déconnecté, named rajoute ceci au fichier messages après la
première requête concernant un nom de domaine qu'il ne connaît pas~:

Jan 28 20:10:11 hazchem named[10147]: No root nameserver for class IN

Ce qui n'est pas très gênant.

Ça marche très bien dans mon cas.  Je peux utiliser le serveur de noms pour
les machines locales lorsque je suis déconnecté du Net en évitant les délais
introduits par les timeout liés à la recherche des noms de domaine extérieurs.
Et lorsque je suis connecté au Net, les requêtes concernant les noms de
domaines extérieurs marchent normalement.
```

   – J'ai aussi reçu des informations sur la façon dont bind interagit avec NFS et le portmapper sur une machine qui est le plus souvent déconnectée de la part de Karl-Max Wanger :

```
J'ai pris l'habitude d'utiliser named sur toutes mes machines qui sont
seulement connectées à Internet de façons occasionnelles grâce à un modem. Le
serveur de noms n'agit qu'en tant que cache, il n'a aucune zone d'autorité et
demande tout aux serveurs du fichier root.cache.  Comme d'habitude avec une
Slackware, named est démarré avant nfsd et mountd.
```

```
Avec l'une de mes machines (un portable Libretto 30), j'ai eu le problème
suivant~: de temps en temps, je pouvais monter ses disques depuis un autre
système connecté sur mon LAN local, mais la plupart du temps, ça ne marchait
pas. Il se passait la même chose que ce soit en utilisant PLIP, une carte
Ethernet PCMCIA ou PPP avec une interface série.

Après quelques temps de réflexions et d'expériences, j'ai découvert que named
empêchait nfsd et mountd de s'enregistrer avec portmapper au démarrage (Je
démarre ces démons au boot d'habitude).  Le fait de lancer named après nfsd
et mountd éliminait ce problème complètement

Comme il n'y a pas de désavantages à modifier ainsi la séquence de boot de
cette façon, j'encourage tout le monde à en faire de même pour éviter des
problèmes potentiels.
```

– Enfin, il y a quelques informations sur le sujet chez *Ask Mr DNS*. C'est a propos de bind 4, vous aurez donc à l'adapter pour que cela fonctionne avec bind 8.

7. Où le serveur de noms qui fait que du cache stocke-t-il son cache ? Puis-je contrôler la taille de ce cache ?

   Le cache est entièrement stocké en mémoire, il n'est *pas* écrit sur le disque. Chaque fois que vous tuez `named`, le cache est perdu. Il n'y a *aucun* moyen de contrôler le cache. `named` gère le cache selon quelques règles simples, et c'est tout. Vous ne pouvez pas contrôler le cache ou sa taille en aucune manière. Si vous voulez vraiment le faire, vous pouvez le faire en bricolant le code de `named`. Mais ce n'est pas recommandé.

8. Est-ce que `named` sauvegarde le contenu du cache entre deux redémarrage ? Puis-je le forcer à le faire ?

   Non, `named` ne sauve *pas* la contenu du cache lorsqu'il meurt. Cela signifie que le cache est reconstruit à partir de zéro chaque fois que vous tuez puis relancez `named`. Il n'y a *aucun* moyen de forcer `named` à sauvegarder le contenu du cache dans un fichier. Si vous voulez vraiment le faire, vous pouvez le faire en bricolant le code de `named`. Mais, encore une fois, ce n'est pas recommandé.

9. Comment je fais pour obtenir un domaine ? Je veux mettre en place mon domaine appelé (par exemple) `linux-rulez.net`. Comment puis-je me faire assigner ce domaine ?

   Contactez votre FAI. Ils seront en mesure de vous aider pour tout ça. Notez toutefois que vous aurez certainement à payer quelque chose.

9 Comment devenir un administrateur DNS de haut vol

Documentation et outils

La Vraie Documentation existe. En ligne et imprimée. Il faut absolument la lire si vous voulez devenir un administrateur DNS du plus haut niveau. Pour ce qui est de la documentation imprimée, le livre standard est *DNS and BIND* de C. Liu et P. Albitz chez O'Reilly & Associates, Sebastopol, CA, ISBN 0-937175-82-X. Je l'ai lu, c'est excellent, la deuxième édition est basée sur bind 4, la troisième sur bind 8. Il y a aussi un chapitre sur le DNS dans *TCP/IP Network Administration*, de Craig Hunt chez O'Reilly..., ISBN 0-937175-82-X. Un autre passage obligé pour une Bonne administration de DNS (ou Bonne n'importe quoi, d'ailleurs) est *Zen and the Art of Motorcycle Maintenance* by Robert M. Pirsig :-) Disponible sous la référence ISBN 0688052304 entre autres.

En ligne, vous trouverez des trucs sur *DNS Resources Directory*, *www.isc.org/bind.html*; Une FAQ, un manuel de référence (BOG ; Bind Operations Guide) aussi bien que des papiers, des descriptions de protocoles et des trucs sur le service DNS (ces documents, ainsi que la majorité, sinon la totalité des RFC mentionnées ci-dessous font partie de la distribution de bind). Je n'ai pas lu la plupart de ces trucs-là, c'est pourquoi je ne suis pas un Grand Administrateur de DNS. Arnt Gulbrandsen, à l'inverse, a lu le BOG et n'en dit que du bien :-). Le newsgroup *comp.protocols.tcp-ip.domains* parle de DNS. En complément, il y a un certain nombre de RFC sur le DNS, les plus importantes sont certainement celles-ci :

RFC 2052

A. Gulbrandsen, P. Vixie, *A DNS RR for specifying the location of services (DNS SRV)*, October 1996

RFC 1918

Y. Rekhter, R. Moskowitz, D. Karrenberg, G. de Groot, E. Lear, *Address Allocation for Private Internets*, 02/29/1996.

RFC 1912

D. Barr, *Common DNS Operational and Configuration Errors*, 02/28/1996.

RFC 1912 Errors

B. Barr *Errors in RFC 1912*, this is available at *www.cis.ohio-state.edu/~barr/rfc1912-errors.html*

RFC 1713

A. Romao, *Tools for DNS debugging*, 11/03/1994.

RFC 1712

C. Farrell, M. Schulze, S. Pleitner, D. Baldoni, *DNS Encoding of Geographical Location*, 11/01/1994.

RFC 1183

R. Ullmann, P. Mockapetris, L. Mamakos, C. Everhart, *New DNS RR Definitions*, 10/08/1990.

RFC 1035

P. Mockapetris, *Domain names - implementation and specification*, 11/01/1987.

RFC 1034

P. Mockapetris, *Domain names - concepts and facilities*, 11/01/1987.

RFC 1033

M. Lottor, *Domain administrators operations guide*, 11/01/1987.

RFC 1032

M. Stahl, *Domain administrators guide*, 11/01/1987.

RFC 974

C. Partridge, *Mail routing and the domain system*, 01/01/1986.