DNS HOWTO

Nicolai Langfeldt janl@math.uio.no Version 1.4.2 du 1er Janvier 1998

Comment devenir un administrateur de DNS à la petite semaine. Traduction française par Julien Vayssière (Julien.Vayssiere@sophia.inria.fr)

Table des matières

1 Préambule **2**

1.1 Aspect juridique 2

1.2 Remerciements et appel aux bonnes volontés 2

1.3 Dédicace 2

2 Introduction. **2**

3 Un Serveur de Nom qui ne fait que cache. **3**

4 Un domaine *simple* **7**

5 Un exemple tiré d'un domaine réel **18**

5.1 /etc/named.boot (ou /var/named/named.boot) 18

5.2 /var/named/root.cache 18

5.3 /var/named/zone/127.0.0 20

5.4 /var/named/zone/land-5.com 20

5.5 /var/named/zone/206 22

6 Maintenance **23**

7 Configuration automatique pour les connexions en dialup **24**

8 FAQ **26**

9 Comment devenir un administrateur DNS de la plus haute volée **28**

1 Préambule

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

1.1 Aspect juridique

(C)opyright 1995 Nicolai Langfeldt. Ne modifiez pas ce document sans en modifier le message de copyright en conséquence. Vous pouvez distribuer ce document librement sous réserve de conserver le message de copyright.

1.2 Remerciements et appel aux bonnes volontés

J'aimerais remercier Arnt Gulbrandsen qui a relu le brouillon de ce document un nombre incalculable de fois et a apporté nombre de suggestions pertinentes. Merci également à tous ceux qui m'ont envoyé leurs suggestions par courrier électronique, merci aussi pour tous les billets de banque que je reçois. Merci beaucoup ! Vous m'aidez vraiment dans ce travail.

Ce document n'est pas destiné à atteindre un jour un état final, alors mailez-moi vos problèmes ainsi que le récit de vos réussites, cela me permettra d'améliorer ce HOWTO. Merci d'envoyer l'argent, les commentaires et/ou les questions 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 avant de m'envoyer un mail.

Si vous voulez traduire ce HOWTO, prévenez-moi pour que je puisse garder le compte de toutes les langues dans lesquelles j'ai été traduit :-).

1.3 Dédicace

Ce HOWTO est dédié à Anne Line Norheim, bien qu'elle ne le lise sans doute jamais. Il ne s'agit pas du tout du genre de fille auquel vous pensez.

2 Introduction.

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

Pour ceux qui sont entièrement nouveaux dans ce domaine, le DNS est le Domain Name System. C'est l'ensemble des règles utilisées par les machines et les logiciels pour établir la correspondance entre les noms de machines et les adresses IP. Ce document explique comment définir de telles correspondances à l'aide d'un système Linux. Une correspondance est tout simplement une association entre deux choses, 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 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.

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/host.conf` (ou `/etc/nnswitch.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 networking/NET-2 HOWTO pour savoir comment faire.

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.

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.vix.com:/pub/bind, soit dans le repertoire release soit dans testing, suivant ce qui correspond le mieux à votre manière de vivre.

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, et les autres avec, en retirerez n'importe quoi. 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 debugger 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, j'énonce deux ou trois 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 : Si 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 Nom qui ne fait que cache.

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

Un serveur de nom qui ne fait que cache sert à trouver 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.

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

```
;  Boot file for caching name server
;
directory /var/named
;
; type          domain                        source file or host
cache           .                             root.cache
primary         0.0.127.in-addr.arpa          pz/127.0.0
```

TRES 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 seront relatifs à cet endroit. D'après le *Linux Filesystem Standard*, ce répertoire doit être `/var/named`. Ainsi, `pz` est un sous-répertoire de `/var/named`, i-e `/var/named/pz`.

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

```
 .       518400  NS      D.ROOT-SERVERS.NET.
 .       518400  NS      E.ROOT-SERVERS.NET.
 .       518400  NS      I.ROOT-SERVERS.NET.
 .       518400  NS      F.ROOT-SERVERS.NET.
 .       518400  NS      G.ROOT-SERVERS.NET.
 .       518400  NS      A.ROOT-SERVERS.NET.
 .       518400  NS      H.ROOT-SERVERS.NET.
 .       518400  NS      B.ROOT-SERVERS.NET.
 .       518400  NS      C.ROOT-SERVERS.NET.
;
D.ROOT-SERVERS.NET.     3600000 A       128.8.10.90
E.ROOT-SERVERS.NET.     3600000 A       192.203.230.10
I.ROOT-SERVERS.NET.     3600000 A       192.36.148.17
F.ROOT-SERVERS.NET.     3600000 A       192.5.5.241
G.ROOT-SERVERS.NET.     3600000 A       192.112.36.4
A.ROOT-SERVERS.NET.     3600000 A       198.41.0.4
H.ROOT-SERVERS.NET.     3600000 A       128.63.2.53
B.ROOT-SERVERS.NET.     3600000 A       128.9.0.107
C.ROOT-SERVERS.NET.     3600000 A       192.33.4.12
```

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 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 named.

La ligne suivante de `named.boot` est la ligne `primary`. J'expliquerai son rôle dans un chapitre ultérieur. Pour l'instant, il s'agit simplement d'un fichier appelé `127.0.0` situé dans le repertoire `pz` :

```
@               IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                                1       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; 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 resolver qui utilise named qui le fait).

Illustrons ce 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. Evidemment, 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.

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 tout de suite. 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). Ces deux derniers fichiers sont documentés dans la page de man resolv(8) (tapez '`man 8 resolv`') dans la plupart des distributions de Linux. Cette page de man est, à mon humble avis, largement lisible et tout le monde, à commencer par les administrateurs de DNS, devrait l'avoir lue. Faites-le maintenant. Si vous vous dites « je le ferai plus tard », vous n'y reviendrez jamais.

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 ca ne démarre pas, essayez plutot '`/usr/sbin/ndc start`'. Si ça ne marche toujours pas, jetez un coup d'oeil au chapitre 8. Maintenant, vous pouvez tester votre configuration. 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 devrez voir quelque chose comme ça :

```
Jun 30 21:50:55 roke named[2258]: starting.  named 4.9.4-REL
Sun Jun 30 21:29:0 3 MET DST 1996
janl@roke.slip.ifi.uio.no:/var/tmp/bind/named Jun 30 21:50:55 roke
named[2258]: cache zone "" loaded (serial 0) Jun 30 21:50:55 roke
named[2258]: primary zone "0.0.127.in-addr.arpa" loaded (serial 1)
```

Si il y a un quelconque message d'erreur, named donnera le nom du fichier dans lequel se trouve l'erreur (soit named.boot, soit root.cache, j'espère :-). Tuez le processus named et retournez vérifier ces fichiers.

Il est maintenant temps de lancer nslookup pour regarder le résultat de votre petit travail manuel

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

>
```

Si vous obtenez ceci, 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.boot`, 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 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 ce (très faible) danger 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`'.

Si vous avez une connexion en dialup (ppp ou slip), lisez le chapitre 7, il y a des conseils pour vous.

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

4 Un domaine *simple*

Comment mettre en place votre propre domaine

Avant d'entrer *vraiment* dans le vif du sujet, il va falloir que je vous fasse un brin de théorie sur le principe du service DNS. Et il va falloir que vous le lisiez, parce que c'est pour votre bien. Si vous ne « voulez » pas, vous devriez au moins le survoler rapidement. Arrêtez le survol quand vous en arrivez au point où j'explique ce qu'il faut mettre dans le fichier `named.boot`.

Le service DNS est un système organisé de manière hiérarchique. Le sommet est désigné par « . » et se prononce « root ». En dessous de . se trouvent un certain nombre de TLD (Top Level Domains), dont les plus connus sont ORG, COM, EDU et NET, mais il y en a beaucoup d'autres. Lorsque vous voulez trouver l'adresse IP de `prep.ai.mit.edu`, votre DNS doit trouver un serveur de noms qui serve le domaine edu. Votre DNS demande d'abord à un serveur de noms de . (il possède déjà les adresses des serveurs de ., c'est ce à quoi sert le fichier `root.cache`), et le serveur de . donne une liste des serveurs d'edu.

En voici une illustration:

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

Interrogeons un serveur situé à la racine (root).

```
> 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 au serveur que nous interrogeons que edu se trouve juste sous ., ce qui permet de faciliter un peu 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
```

Ceci nous dit que les serveurs `*.root-servers.net` servent le domaine `edu.`, nous pouvons donc continuer 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
```

`steawb`, `w20ns` et `bitsy` servent le domaine `mit`, prenons-en un au hasard et posons-lui la question au sujet de `ai.mit.edu`:

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

Les noms de domaine ne sont pas sensibles à la différence entre lettres minuscules et majuscules, mais 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 = WHEATIES.AI.MIT.EDU
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

Authoritative answers can be found from:
AI.MIT.EDU      nameserver = WHEATIES.AI.MIT.EDU
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
WHEATIES.AI.MIT.EDU     internet address = 128.52.32.13
```

```
WHEATIES.AI.MIT.EDU     internet address = 128.52.35.13
ALPHA-BITS.AI.MIT.EDU   internet address = 128.52.32.5
ALPHA-BITS.AI.MIT.EDU   internet address = 128.52.37.5
GRAPE-NUTS.AI.MIT.EDU   internet address = 128.52.32.4
GRAPE-NUTS.AI.MIT.EDU   internet address = 128.52.36.4
TRIX.AI.MIT.EDU internet address = 128.52.32.6
TRIX.AI.MIT.EDU internet address = 128.52.38.6
MUESLI.AI.MIT.EDU       internet address = 128.52.32.7
MUESLI.AI.MIT.EDU       internet address = 128.52.39.7
```

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

```
> server WHEATIES.AI.MIT.EDU.
Default Server:  WHEATIES.AI.MIT.EDU
Addresses:  128.52.32.13, 128.52.35.13
```

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

```
> set q=any
> prep.ai.mit.edu.
Server:  WHEATIES.AI.MIT.EDU
Addresses:  128.52.32.13, 128.52.35.13

prep.ai.mit.edu CPU = dec/decstation-5000.25    OS = unix
prep.ai.mit.edu
        inet address = 18.159.0.42, protocol = tcp
         #21 #23 #25 #79
prep.ai.mit.edu preference = 1, mail exchanger = life.ai.mit.edu
prep.ai.mit.edu internet address = 18.159.0.42
ai.mit.edu      nameserver = alpha-bits.ai.mit.edu
ai.mit.edu      nameserver = wheaties.ai.mit.edu
ai.mit.edu      nameserver = grape-nuts.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 = life.ai.mit.edu
ai.mit.edu      nameserver = mintaka.lcs.mit.edu
life.ai.mit.edu internet address = 128.52.32.80
alpha-bits.ai.mit.edu   internet address = 128.52.32.5
wheaties.ai.mit.edu     internet address = 128.52.35.13
wheaties.ai.mit.edu     internet address = 128.52.32.13
grape-nuts.ai.mit.edu   internet address = 128.52.36.4
grape-nuts.ai.mit.edu   internet address = 128.52.32.4
mini-wheats.ai.mit.edu  internet address = 128.52.32.11
mini-wheats.ai.mit.edu  internet address = 128.52.54.11
mintaka.lcs.mit.edu     internet address = 18.26.0.36
```

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.

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 deux premières années.

Pour dire vrai, je viens de vous mentir. Le service DNS ne marche pas vraiment comme je vous l'ai exposé. Mais ça en est suffisamment proche.

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.

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

```
primary 0.0.127.in-addr.arpa pz/127.0.0
```

Notez bien l'absence de '.' à la fin des noms de domaine de ce fichier. La première ligne désigne le fichier `pz/127.0.0` comme étant celui qui définit le domaine `0.0.127.in-addr.arpa`. Nous avons déjà configuré ce fichier, il se présente comme ceci :

```
@               IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                                1       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; 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, au contraire du fichier `named.boot` dont nous parlions un peu plus haut. 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 colonne 'domain' du fichier `named.boot`. 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'abbréviation de 'Start Of Authority' (Origine d'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 dit au service DNS quelle est la machine qui est serveur de noms pour ce domaine. Enfin, le champ PTR dit que 1 (c'est-à-dire 1.0.0.127.IN-ADDR.ARPA, c'est-à-dire 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, et ce doit être le tout premier champ du fichier.

Ce champ SOA décrit la zone, son origine (une machine appelée `linux.bogus`), qui est responsable de son contenu (`hostmaster@linux.bogus`), de quelle version du fichier de zone il s'agit (serial : 1), et quelques autres choses qui ont à voir avec le cache et les serveurs DNS secondaires. Pour les champs restants, refresh, retry, expire et minimum, utilisez les valeurs données dans ce HOWTO et tout se passera certainement très bien.

Le champ NS nous dit qui fait serveur DNS pour `0.0.127.in-addr.arpa`, il s'agit de `ns.linux.bogus`. Le champ PTR nous indique que `1.0.0.127.in-addr.arpa` (c'est-à-dire 127.0.0.1), est connu sous le nom `localhost`.

Maintenant, relancez votre named (la commande est `ndc restart`) et utilisez nslookup pour regarder le résultat de ce que vous avez fait :

```
$ 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 ligne primary dans `named.boot` :

```
primary linux.bogus 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.boot`.

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

```
;
; Zone file for linux.bogus
;
; Mandatory minimum for a working domain
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199511301       ; serial, todays date + todays serial #
                        28800           ; refresh, seconds
                        7200            ; retry, seconds
                        3600000         ; expire, seconds
                        86400 )         ; minimum, seconds
                NS      ns.linux.bogus.
                NS      ns.friend.bogus.
                MX      10 mail.linux.bogus   ; Primary Mail Exchanger
                MX      20 mail.friend.bogus. ; Secondary Mail Exchanger

localhost       A       127.0.0.1
ns              A       127.0.0.2
mail            A       127.0.0.4
```

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', ce peut être toute autre adresse mail légale, mais il faut dans ce cas 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 vers laquelle le courrier doit être adressé en priorité. En cas d'échec, le courrier 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.

Relançez 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 = linux.bogus
        mail addr = hostmaster.linux.bogus
        serial = 199511301
        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     nameserver = ns.friend.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
linux.bogus     nameserver = ns.friend.bogus
ns.linux.bogus  internet address = 127.0.0.2
mail.linux.bogus        internet address = 127.0.0.4
```

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 délibérement, c'était juste 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 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 à taper. Dans un fichier de zone, le nom de domaine doit soit être écrit et terminé par un point, ou ne pas être inclus du tout. Dans ce dernier cas, le nom de domaine par défaut est l'origine.

Il faut que j'insiste sur le point suivant : dans le fichier named.boot, il ne doit *pas* y avoir de points après les noms de domaines. Vous ne pouvez pas vous imaginer combien de fois un '.' en trop ou en moins a tout foutu en l'air et a causé de graves prises de tête aux gens.

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

```
;
; Zone file for linux.bogus
;
; Mandatory minimum for a working domain
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199511301       ; serial, todays date + todays
serial #
                        28800           ; refresh, seconds
                        7200            ; retry, seconds
                        604800          ; expire, seconds
                        86400 )         ; minimum, seconds

                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
ns              A       127.0.0.2
mail            A       127.0.0.4
;
; Extras
;
@               TXT     "Linux.Bogus, your DNS consultants"

ns              MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "Pentium" "Linux 1.2"
                TXT     "RMS"
richard         CNAME   ns
www             CNAME   ns

donald          A       127.0.0.3
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "i486"  "Linux 1.2"
                TXT     "DEK"

mail            MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "386sx" "Linux 1.0.9"
```

```
ftp             A       127.0.0.5
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "P6" "Linux 1.3.59"
```

Vous préfererez certainement changer les trois premiers champs A de place afin qu'ils se trouvent à côté de leurs autres champs, au lieu d'être en tête du fichier.

Il y a un certain nombre de nouveaux RR que nous allons passer en revue : HINFO (Host INFOrmation), et c'est une bonne habitude à prendre que d'encadrer chaque partie de guillemets. La première partie est la descrition 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 1.2. Le champ TXT est libre et vous pouvez l'utiliser pour tout ce qui vous passera par la tête. CNAME (Canonical NAME) est un champ qui permet de donner à chaque machine plusieurs noms. Ainsi, richard et www sont des alias pour ns. Il est important de noter que les champs MX, CNAME et SOA ne doivent *jamais* mentionner un CNAME, ils ne doivent mentionner que des noms de machine possédant un champ A. Ainsi, il serait faux d'avoir :

```
foobar CNAME richard ; NO!
```

En revanche, ceci serait correct :

```
foobar CNAME ns ; Yes!
```

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 elle 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 127.0.0.2
```

Paul Vixie, le gourou dont nous avons déjà parlé, recommande de ne *jamais* utiliser de CNAME. Envisagez donc *très* sérieusement de ne pas en utiliser.

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]
 linux.bogus.                   SOA   ns.linux.bogus
hostmaster.linux.bogus. (199511301 28800 7200 604800 86400)
 linux.bogus.                   NS    ns.linux.bogus
 linux.bogus.                   NS    ns.friend.bogus
 linux.bogus.                   MX    10   mail.linux.bogus
 linux.bogus.                   MX    20   mail.friend.bogus
 linux.bogus.                   TXT   "Linux.Bogus, your DNS consultants"
 localhost                      A     127.0.0.1
 mail                           A     127.0.0.4
 mail                           MX    10   mail.linux.bogus
 mail                           MX    20   mail.friend.bogus
 mail                           HINFO 386sx       Linux 1.0.9
 donald                         A     127.0.0.3
 donald                         MX    10   mail.linux.bogus
 donald                         MX    20   mail.friend.bogus
 donald                         HINFO i486        Linux 1.2
 donald                         TXT   "DEK"
 www                            CNAME ns.linux.bogus
 richard                        CNAME ns.linux.bogus
 ftp                            A     127.0.0.5
 ftp                            MX    10   mail.linux.bogus
 ftp                            MX    20   mail.friend.bogus
 ftp                            HINFO P6          Linux 1.3.59
 ns                             A     127.0.0.2
 ns                             MX    10   mail.linux.bogus
 ns                             MX    20   mail.friend.bogus
 ns                             HINFO Pentium     Linux 1.2
 ns                             TXT   "RMS"
 linux.bogus.                   SOA   ns.linux.bogus hostmaster.linux.bogus.
(199511301 28800 7200 604800 86400)
```

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
```

... en d'autres termes, le vrai nom de `www.linux.bogus` est `ns.linux.bogus`

```
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     nameserver = ns.friend.bogus
ns.linux.bogus  internet address = 127.0.0.2
```

et ns.linux.bogus possède l'adresse 127.0.0.2. Ca a l'air bon aussi.

On se relaxe

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

5 Un exemple tiré d'un domaine réel

Ou 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 de la différence entre un vrai domaine et l'exemple bidon ci-dessus n'était pas très claire.

Une chose importante à propos de cet exemple : ne l'entrez *pas* dans vos serveurs de nom ! Ne l'utilisez que comme référence. Si vous voulez expérimentez, faites-le avec l'exemple du domain `bogus`. J'utilise cet exemple avec la permission de David Bullock de LAND-5. Ces fichiers étaient à jour le 24 Septembre 96, et peuvent donc différer de ce que vous pouvez trouver en questionnant les serveurs de nom de LAND-5 aujourd'hui. Et souvenez-vous : enlevez les caractères blancs en tête de ligne ;-)

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

Voici les deux lignes primary pour les deux zones inverses dont nous avons besoin : le réseau 127.0.0, ainsi que le sous-réseau 206.6.177 de LAND-5. Et une ligne primary pour la forward zone land-5.com. Notez enfin qu'au lieu de mettre les fichiers dans un répertoire appelé `pz`, comme je fais dans ce HOWTO, il les place dans un répertoire appelé `zone`.

```
;  Boot file for LAND-5 name server
;
directory /var/named
;
; type          domain                          source file or host
cache           .                               root.cache
primary         0.0.127.in-addr.arpa            zone/127.0.0
primary         177.6.206.in-addr.arpa          zone/206.6.177
primary         land-5.com                      zone/land-5.com
```

5.2 /var/named/root.cache

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 dig.

```
; <<>> DiG 2.1 <<>>
;; res options: init recurs defnam dnsrch
```

```
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr rd ra; Ques: 1, Ans: 9, Auth: 0, Addit: 9
;; QUESTIONS:
;;      ., type = NS, class = IN

;; ANSWERS:
 .       518357  NS      H.ROOT-SERVERS.NET.
 .       518357  NS      B.ROOT-SERVERS.NET.
 .       518357  NS      C.ROOT-SERVERS.NET.
 .       518357  NS      D.ROOT-SERVERS.NET.
 .       518357  NS      E.ROOT-SERVERS.NET.
 .       518357  NS      I.ROOT-SERVERS.NET.
 .       518357  NS      F.ROOT-SERVERS.NET.
 .       518357  NS      G.ROOT-SERVERS.NET.
 .       518357  NS      A.ROOT-SERVERS.NET.

;; ADDITIONAL RECORDS:
H.ROOT-SERVERS.NET.     165593  A       128.63.2.53
B.ROOT-SERVERS.NET.     165593  A       128.9.0.107
C.ROOT-SERVERS.NET.     222766  A       192.33.4.12
D.ROOT-SERVERS.NET.     165593  A       128.8.10.90
E.ROOT-SERVERS.NET.     165593  A       192.203.230.10
I.ROOT-SERVERS.NET.     165593  A       192.36.148.17
F.ROOT-SERVERS.NET.     299616  A       192.5.5.241
G.ROOT-SERVERS.NET.     165593  A       192.112.36.4
A.ROOT-SERVERS.NET.     165593  A       198.41.0.4

;; Total query time: 250 msec
;; FROM: land-5 to SERVER: default -- 127.0.0.1
;; WHEN: Fri Sep 20 10:11:22 1996
;; MSG SIZE  sent: 17  rcvd: 312
```

```
;  Boot file for LAND-5 name server
;
directory /var/named
;
; type          domain                          source file or host
cache           .                               root.cache
primary         0.0.127.in-addr.arpa            zone/127.0.0
primary         177.6.206.in-addr.arpa          zone/206.6.177
primary         land-5.com                      zone/land-5.com
```

5.3 /var/named/zone/127.0.0

Seulement 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 besoin 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 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 à 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 de manière monotone, 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. Envisagez d'utiliser deux chiffres au lieu d'un seul.

```
@       IN      SOA     land-5.com. root.land-5.com. (
                        199609206       ; serial, todays date + todays serial #
                        10800           ; refresh, seconds
                        7200            ; retry, seconds
                        10800           ; expire, seconds
                        86400 )         ; minimum, seconds
                NS      land-5.com.
                NS      ns2.psi.net.
                MX      10 land-5.com.  ; Primary Mail Exchanger

localhost       A       127.0.0.1
```

```
router          A       206.6.177.1

land-5.com.     A       206.6.177.2
ns              CNAME   land-5.com.
ftp             CNAME   land-5.com.
www             CNAME   land-5.com.
mail            CNAME   land-5.com.
news            CNAME   land-5.com.

funn            A       206.6.177.3
illusions       CNAME   funn.land-5.com.
@               TXT     "LAND-5 Corporation"

;
;       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
; {Many repetitive definitions deleted - SNIP}
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
```

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.

5.5 /var/named/zone/206

Les commentaires se trouvent après ce 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.
3       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.
; {Many repetitive definitions deleted - SNIP}
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 pour l'explication de la résolution de prep.ai.mit.edu, tout ceci est légerement fictif.

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.

Certains gens 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 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.cache` 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 énormement à un fichier root.cache, avec quelques chiffres en plus. Ces chiffres supplémentaires sont inoffensifs. Sauvez-le dans un fichier (`dig . @e.root-servers.net >root.cache.new`) et remplaçez l'ancien fichier `root.cache` avec.

N'oubliez pas de relancer named après avoir remplacé le fichier root.cache.

Al Longyear m'a envoyé ce script qui peut être lancé automatiquement pour mettre à jour `named.cache`. 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 vous faudra modifier quelque peu ce fichier pour qu'il fonctionne chez vous.

```
#!/bin/sh
#
# Update the nameserver cache information file once per month.
# This is run automatically by a cron entry.
#
(
 echo "To: hostmaster <hostmaster>"
 echo "From: system <root>"
 echo "Subject: Automatic update of the named.boot file"
```

```
 echo

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

 dig . @rs.internic.net >named.cache.new

 echo "The named.boot file has been updated to contain the following
information:"
 echo
 cat named.boot.new

 chown root.root named.cache.new
 chmod 444 named.cache.new
 rm -f named.cache.old
 mv named.cache named.cache.old
 mv named.cache.new named.cache
 ndc restart
 echo
 echo "The nameserver has been restarted to ensure that the update is
complete.
"
 echo "The previous named.cache file is now called
/var/named/named.cache.old."
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

7 Configuration automatique pour les connexions en dialup

Ce chapitre explique comment j'ai configuré les choses pour que tout se fasse de manière automatique. Le façon dont j'ai procédé peut ne pas vous convenir du tout, mais ça peut quand même vous donner des idées. En outre, j'utilise ppp pour faire du dialup alors que beaucoup utilisent slip ou cslip, c'est pourquoi votre configuration peut être totalement différente de la mienne. Mais le programme dip de slip doit pouvoir faire beaucoup de ces choses pour vous.

Normalement, lorsque je ne suis pas connecté au Net, mon fichier `resolv.conf` contient simplement la ligne suivante :

```
domain uio.no
```

Ca permet d'éviter que la librairie de résolution de nom n'essaye de se connecter à un serveur de noms qui ne peut rien pour moi. Mais lorsque je me connecte, je veux démarrer mon named et avoir un `resolv.conf` qui ressemble à ceux que nous avons vus plus haut. J'ai résolu ce problème en gardant deux fichiers 'à

la' `resolv.conf` que j'ai appelés `resolv.conf.local` et `resolv.conf.connected`. Le dernier ressemble au fichier `resolv.conf` décrit plus haut dans ce document.

Pour me connecter automatiquement au Net, je lance un script appelé 'ppp-on':

```
#!/bin/sh
echo calling...
pppd
```

pppd possède un fichier appelé `options` qui lui donne tous les détails nécessaires pour établir la connexion. Une fois que ma connexion ppp est établie, le pppd lance un script appelé `ip-up` (Tout ceci est décrit dans la page de man de pppd). Voici un extrait de ce script :

```
#!/bin/sh
interface="$1"
device="$2"
speed="$3"
myip="$4"
upip="$5"
 ...
cp -v /etc/resolv.conf.connected /etc/resolv.conf
 ...
/usr/sbin/named
```

C'est ici que je démarre mon named. A la fermeture de la connexion ppp, pppd lance un script appelé `ip-down` :

```
#!/bin/sh
cp /etc/resolv.conf.local /etc/resolv.conf
read namedpid {$<$}/var/run/named.pid
kill $namedpid
```

De cette manière, les choses sont configurées lors de la connexion et déconfigurées à la déconnexion.

Quelques programmes, je pense en particulier à irc et talk, font un peu trop de suppositions, et pour que l'option dcc d'irc ou talk puissent fonctionner, il faut modifier le fichier hosts. J'insère cette ligne dans mon script `ip-up` :

```
cp /etc/hosts.ppp /etc/hosts
echo $myip roke >>/etc/hosts
```

`hosts.ppp` contient tout simplement

```
 127.0.0.1 localhost
```

C'est la ligne echo qui insère l'adresse IP que j'obtiens pour mon nom d'hôte (roke). Il faut que vous remplaciez roke par le nom sous lequel votre machine se connaît. On peut l'obtenir par la commande hostname.

Il n'est pas très malin de faire tourner named lorsque l'on n'est pas connecté au Net. En effet, named essaye d'envoyer des requêtes sur le Net et, comme il a un grand timeout, il vous faut attendre ce timeout à chaque fois qu'un de vos programmes essaye de résoudre un nom. Si vous utilisez une connexion en dialup, il vous faut démarrer named à la connexion et l'arrêter lors de la déconnexion. J'ai reçu du courrier me disant que le comportement normal n'était pas celui-là, mais je n'ai pas réussi à le faire marcher sans avoir à attendre de longs timeouts. Merci de me mailer *tous* les détails si vous avez des informations plus précises.

Certaines personnes utilisent une directive forwarders parce qu'elles se trouvent au bout d'une ligne très lente. Si votre fournisseur d'accès a des serveurs de nom aux adresses 1.2.3.4 et 1.2.3.5, vous pouvez insérer la ligne suivante:

```
forwarders 1.2.3.4 1.2.3.5
```

dans le fichier `named.boot`. Ceci vous permettra de diminuer la quantité de trafic IP généré par votre machine, et peut-être accélérer un peu les choses. C'est particulièrement important si vous êtes facturé au volume de données transférées. En plus, ça vous évite d'avoir à maintenir un serveur de nom pour faire cache: vous n'avez pas à maintenir à jour un fichier `root.cache` qui est désormais vide.

8 FAQ

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 me mailer.

1. Question: Comment utiliser un DNS si l'on se trouve derrière un firewall? Voici quelques indices: 'forwarders', 'slaves', et jetez un coup d'oeil à la bibliographie qui se trouve à la fin de ce HOWTO.

2. Question: Comment dire à un DNS qu'il doit faire une rotation entre un certain nombre d'adresses pour un service donné, comme par exemple si l'on veut obtenir un effet d'équilibrage de charge sur www.busy.site, ou quelque chose du même genre?

   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.

3. Mon système ne possède pas le programme ndc. Que puis-je faire? Votre système doit avoir une vieille version de bind, certainement obsolète. Si la sécurité est une chose importante pour vous, passez à une version récente de bind tout de suite. Si ce n'est pas le cas, vous pouvez vivre sans ndc. Et à la place de lancer `ndc start`, tapez `named`. `ndc reload` devient ainsi `named.reload` et `ndc restart` devient ainsi `named.restart`. Tous ces programmes se trouvent le plus souvent dans `/usr/sbin`

4. 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.boot du serveur secondaire:

```
secondary      linux.bogus             127.0.0.1       sz/linux.bogus
```

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

   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 fais 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.cache.fake
; this file contains no information
--------------

Quand je me deconnecte, je copie le fichier root.cache.fake vers
root.cache et je relance named.

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

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

Lorsque je suis deconnecte, named rajoute ceci au fichier messages apres
la premiere requete concernant un nom de domaine qu'il ne connait pas :

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

Ce qui n'est pas tres genant.

Ca marche tres bien dans mon cas. Je peux utiliser le serveur de noms
pour les machines locales lorsque je suis deconnecte du Net en evitant
les delais introduits par les timeout lies a la recherche des noms de
domaine exterieurs. Et lorsque je suis connecte au Net, les requetes
concernant les noms de domaines exterieurs marchent normalement.
```

6. Où est-ce que le serveur de nom qui fait cache stocke 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.

7. Est-ce que named sauvegarde le contenu du cache entre deux restarts? 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 ce n'est, encore une fois, pas recommandé.

9 Comment devenir un administrateur DNS de la plus haute volée

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. 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. Prisig:-) Disponible sous la référence ISBN 0688052304.

Online, vous trouverez des trucs sur `(http://www.dns.net/dnsrd/)`, `(http://www.vix.com/isc/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. 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 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 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.