DNS HOWTO

Nicolai Langfeldt `janl@linpro.no`

Versione 2.2.1, 18 luglio 1999

Come diventare un piccolo amministratore DNS. Traduzione di Federico Cossu (f.cossu@trident.nettuno.it), flug-firenze.

Indice

1 Preambolo

Parole-chiave: DNS, bind, bind-4, bind-8, named, dialup, ppp, slip, isdn, Internet, domain, name, hosts, resolving, caching.

Questo documento fa parte del Linux Documentation Project.

1.1 Note legali

1.2 Crediti e richieste di aiuto

Voglio ringraziare Arnt Gulbrandsen al quale ho provocato qualche disagio a causa di questo lavoro e che ha provveduto a fornire utili suggerimenti. Voglio anche ringraziare le numerose persone che mi hanno mandato in email suggerimenti e note.

Questo non sarà mai un documento finito. Vi prego di mandarmi delle email se avete incontrato problemi o successi, questo contribuirà a migliorare l'HOWTO. Così vi prego di mandare commenti e/o domande o soldi a janl@math.uio.no. Nel caso in cui mandaste delle email e attendiate delle risposte per favore *assicuratevi* che il return-address sia corretto e operatvo. Inoltre, **per favore** leggete la sezione 8 (Domande e risposte) prima di scrivermi. Un'altra cosa, comprendo solo il norvegese e l'inglese.

Se vi venisse in mente di tradurre questo HOWTO fatemelo sapere in modo che possa tener traccia delle lingue in cui è stato pubblicato, e in modo che possa avvisarvi quando l'HOWTO è stato aggiornato.

1.3 Dediche

Questo HOWTO è dedicato a Anne Line Norheim Langfeldt. Anche se lei non lo leggerà mai poiché non è quel tipo di ragazza.

2 Introduzione

DNS è il Domain Name System. DNS converte i nomi delle macchine negli indirizzi IP che queste macchine hanno nella rete. In pratica fa corrispondere i nomi agli indirizzi e viceversa, e in più fa qualche altra cosa. Questo HOWTO spiega come definire questa corrispondenza (mapping) usando un sistema Linux. Il "mapping" è semplicemente un'associazione tra due cose, in questo caso si tratta del nome di una macchina, come `ftp.linux.org`, e il numero IP (o indirizzo) della stessa macchina `199.249.150.4`.

DNS è, per i non iniziati (voi ;-), una delle aree più opache dell'amministrazione di rete. Questo HOWTO cercherà di rendere più chiare alcune cose. Esso descrive come impostare un *semplice* name server DNS. Partendo da un server "caching only" per poi avviarsi all'impostazione di un server DNS primario per un dominio. Per configurazioni più complesse date uno sguardo alla sezione 8 (Domande e Risposte) di questo documento. Se non fosse descritto qui avrete bisogno di *leggere* la Vera Documentazione. Tornerò più tardi su in che cosa consista questa Vera Documentazione nell'9 (ultimo capitolo)

Prima che cominciate dovrete configurare la vostra macchina in modo che possa fare "telnet" dentro e fuori di essa, e che possa connettersi con successo alla rete, e in particolar modo dovreste poter fare `telnet 127.0.0.1` e ottenere la vostra stessa macchina (provate subito!). Avrete anche bisogno di un buon `/etc/nsswitch.conf` (oppure `/etc/host.conf`), e dei file `/etc/resolv.conf` e `/etc/hosts` come punto di partenza, finché non spiegherò la loro funzione. Se non avete già tutto questo impostato e funzionante il NET-3-HOWTO e il PPP-HOWTO spiegano come farlo. Leggeteli.

Quando dico "la vostra macchina" intendo la macchina sulla quale state cercando di impostare il DNS. Nessun'altra macchina che potreste avere è coinvolta nel vostro lavoro.

Assumerò che non siate dietro un qualche tipo di firewall che blocca le richieste di nomi. Se invece lo siete, avrete bisogno di una speciale configurazione. Guardate la sezione 8 (Domande e Risposte).

Il servizio di risoluzione dei nomi su Unix è fatto da un programma chiamato `named`. Questo fa parte del pacchetto "bind" che è coordinato da Paul Vixie dell'Internet Software Consortium. `Named` è incluso in molte distribuzioni Linux e usualmente è installato come `/usr/sbin/named`. Se avete named probabilmente potrete usarlo; se non l'avete potrete prendere i binari da un qualunque sito ftp su Linux, altrimenti prenderete i più recenti e grandiosi sorgenti da *ftp.isc.org:/isc/bind/src/cur/bind-8/*. Questo HOWTO è riferito alla versione 8 di bind. Le vecchie versioni dell'HOWTO che riferiscono a bind 4 sono ancora disponibili presso *http://www.math.uio.no/~janl/DNS/* nel caso utilizzaste bind 4. Se la man page di named fa riferimento (per precisione alla fine, nella sezione FILES) a `named.conf` avete bind 8, se fa riferimento a `named.boot` avete bind 4. Se avete il 4 e siete coscienti del problema sicurezza dovreste aggiornare alla versione 8.

DNS è un database distribuito sulla rete (net-wide). Fate attenzione a cosa ci metterete dentro. Se ci metterete spazzatura, voi e gli altri ne ricaverete solo quella. Mantenete il vostro DNS snello e consistente e così otterrete un buon servizio da esso. Imparate ad usarlo, ad amministrarlo, a risolverne eventuali problemi, e diventerete un altro buon amministratore che impedisce alla rete di cadere sulle proprie ginocchia a causa della cattiva manutenzione.

In questo documento dichiaro nettamente una manciata di cose che non sono prorpio corrette (sono comunque delle mezze verità). Tutto ciò nell'interesse della semplificazione. Le cose, probabilmente ;-) funzioneranno se crederete a quello che dico.

Suggerimento: Fate una copia di backup di tutti i file che vi chiederò di modificare e che già avete, cosicché possiate tornare operativi dopo aver visto che nulla funzionava.

3 Un name server caching only

Una prima pugnalata alla configurazione del DNS, molto utile per gli utenti dial-up

Un name server caching only troverà le risposte alle richieste di nomi e ricorderà le risposte quando la prossima volta ne avrete bisogno. Questo abbrevierà signficativamente il tempo di attesa per le volte successive, specialmente se avete una connessione lenta.

Prima di tutto vi occorre un file chiamato `/etc/named.conf`. Questo viene letto quando named parte. Per adesso dovrebbe contenere semplicemente:

```
// File di configurazione per un name server caching only

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

```
        // Decommentatare questa cosa potrebbe aiutare se di deve passare
        // attraverso un firewall e le cose non funzionano bene

        // query-source port 53;
};

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

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

La linea '`directory`' dice a named dove guardare per i file. Tutti i file nominati in seguito saranno relativi a questa directory. Perciò `pz` è una directory sotto `/var/named`, ovvero `/var/named/pz`. `/var/named` è la giusta directory in accordo con il *Linux File system Standard*.

In questo è citato il file chiamato `/var/named/root.hints`. `/var/named/root.hints` dovrebbe contenere questo: (*Se avete intenzione di fare copia e incolla di questo file da una versione elettronica di questo documento per favore notate che* ***non*** *ci devono essere spazi vuoti all'inizio del file, ovvero tutte le linee devono cominciare con un carattere non-blank (non deve essere il carattere "spazio"). Alcuni software per il processing dei documenti inseriranno spazi all'inizio delle linee, causando confusione. In questo caso rimuovete i 2 spazi vuoti iniziali*)

```
;
; There might be opening comments here if you already have this file.
; If not don't worry.
;
.                       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
```

Questo file descrive i root name server nel mondo. Esso cambia col tempo e deve essere aggiornato se necessario. Leggete la sezione 6 (Manutenzione) per sapere come fare.

La sezione successiva in `named.conf` è l'ultima `zone`. Spiegherò il suo utilizzo nell'ultimo capitolo, per adesso create solo un file chiamato `127.0.0` nella subdirectory `pz`: (*Ancora, se fate copia e incolla rimuovete gli spazi iniziali*)

```
@               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.
```

Poi, avrete bisogno di un `/etc/resolv.conf` che somiglia vagamente a questo: (*togliete gli spazi!*)

```
search sottodominio.proprio-dominio.edu proprio-dominio.edu
nameserver 127.0.0.1
```

La linea '`search`' specifica su quali domini deve avvenire la ricerca per ogni nome di host al quale volete collegarvi. La linea `nameserver` specifica l'indirizzo del vostro nameserver, in questo caso la vostra stessa macchina poiché è qui che named lavora (127.0.0.1 è corretto, non importa se la macchina ha già un altro indirizzo). Se volete inserire più name server mettete una linea '`nameserver`' per ognuno di essi. (Nota: named non legge mai questo file, il risolutore che usa named invece sì.)

Vediamo cosa fa questo file: se un client cerca la macchina `pippo`, allora il primo tentativo che verrà fatto sarà `pippo.sottodominio.proprio-dominio.edu`, poi `pippo.proprio-dominio.edu` e infine `pippo`. Se un client cerca `sunsite.unc.edu`, il primo tentativo sarà `sunsite.unc.edu.sottodominio.proprio-dominio.edu`, poi `sunsite.unc.edu.proprio-dominio.edu` e infine `sunsite.unc.edu`. Potreste non voler mettere troppi domini nella linea di ricerca, si impiega del tempo a provarli tutti.

L'esempio assume che voi facciate parte del dominio `sottodominio.proprio-dominio.edu`, quindi, probabilmente, la vostra macchina sarà `propria-macchina.sottodominio.proprio-dominio.edu`. La linea di ricerca non dovrebbe contenere il vostro TLD (Top Level Domain, '`edu`' in questo caso). Se avete spesso bisogno di collegarvi a host in un altro dominio, potete aggiungere questo dominio in una linea di ricerca come questa: (*Ricordate di togliere gli spazi iniziali, se presenti*)

```
search sottodominio.proprio-dominio.edu proprio-dominio.edu altro-dominio.com
```

e così via. Ovviamente dovrete metterci domini reali. Per favore notate l'assenza del punto alla fine dei nomi di dominio. Questo è importante.

In seguito, a seconda della versione di libc che avete ci sarà bisogno di sistemare `/etc/nsswitch.conf` o `/etc/host.conf`. Se avete già `nsswitch.conf` sarà questo che sistemerete, altrimenti sarà `host.conf`.

/etc/nsswitch.conf

Questo è un grosso file che specifica dove ottenere diversi tipi di dati, da quale file o database. Solitamente all'inizio contiene utili commenti, che dovreste leggere. Poi cercate la linea che comincia per '`hosts:`', si dovrebbe leggere:

```
hosts:      files dns
```

(*avete ricordato cosa fare degli eventuali spazi iniziali, vero? non lo dirò più.*)

Se non ci fosse una tale linea (che comincia per '`hosts:`') dovrete metterla. Questa linea dice che i programmi devono per prima cosa guardare nel file `/etc/hosts`, dopo devono usare il DNS in accordo con `resolv.conf`.

/etc/host.conf

Probabilmente contiene numerose linee, una di queste dovrebbe cominciare con `order` e somigliare a questa:

```
order hosts,bind
```

Se la linea '`order`' non ci fosse la dovrete aggiungere. Questa linea dice alle routine predisposte alla risoluzione dei nomi che devono per prima cosa guardare nel file `/etc/hosts`, dopo devono chiedere al name server (che voi avete indicato in `resolv.conf` all'indirizzo 127.0.0.1).

3.1 Far partire named

Adesso è ora di far partire named. Se state usando una connessione dial-up, per prima cosa collegatevi. Fate '`ndc start`', e premete return, senza opzioni. Se non funziona provate '`/usr/sbin/ndc start`'. Se non va leggete la sezione 8 (Domande e Risposte). Se andate a leggere il file che contiene il log di sistema (usualmente chiamato `/var/adm/messages`, controllate anche la directory `/var/log` e un altro file da controllare in questa è `syslog`) quando named parte (fate `tail -f /var/log/messages`) dovreste leggere qualcosa di simile a:

(le linee che terminano per \ continuano sulla linea successiva)

```
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.
```

Se ci sono messaggi d'errore significa che c'è un problema. Named dirà il nome del file scorretto (uno tra named.conf e root.hints spero :-)). Uccidete named e tornate indietro per controllare quel file.

Adesso potete testare la vostra configurazione. Usate nslookup per esaminare il vostro lavoro.

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

>
```

Se corrisponde a quello che vedete significa che sta funzionando. Altrimenti tornate indietro e ricontrollate tutto. Ogni volta che cambiate il file `named.conf` dovete far ripartire named con il comando `ndc restart`.

Adesso potete immettere una interrogazione (query). Cercate di fare il look-up di macchine vicine a voi. `pat.uio.no` è vicina a me, all'università di Oslo:

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

Name:    pat.uio.no
Address:  129.240.130.16
```

nslookup adesso ha chiesto al vostro named di cercare la macchina `pat.uio.no`. Poi contatta una delle macchine name server indicate nel file `root.hints`, e chiede loro la strada per arrivarci. Potrebbe essere necessario un po' di tempo prima che sia disponibile il risultato, come potrebbe essere necessario cercare in tutti i domini elencati in `/etc/resolv.conf`.

Se lo chiederete nuovamente otterrete questo:

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

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

Notate la linea "`Non-authoritative answer:`" che abbiamo ottenuto questa volta. Significa che named non è uscito sulla rete per fare la richiesta: l'informazione era già nella cache. Ma questa informazione (memorizzata nella cache) *potrebbe* essere non aggiornata (scaduta). Verrete informati di questa possibilità (molto piccola) con il messaggio '`Non-authorative answer:`'. Quando `nslookup` risponde in questo modo alla seconda richiesta per uno stesso host è sicuro che named ha messo nella cache l'informazione e che sta funzionando. Si esce da `nslookup` dando il comando '`exit`'.

3.2 Migliorarlo ancora

Nelle reti grosse, ben organizzate, accademiche o relative a ISP (Internet Service Provider) scoprirete che a volte le persone che lavorano sulla rete mettono a punto una gerarchia di impiego dei server DNS, che aiuta ad alleggerire il carico sulla rete interna e sui server esterni. Non è facile capire se vi trovate dentro o fuori una rete. Comunque non è importante e usando il server DNS del vostro provider come "forwarder" farete in modo che le risposte alle vostre richieste siano più veloci e meno pesanti per la vostra rete. Se usate un modem questa può essere una piccola vittoria. Tanto per fare un esempio assumeremo che il vostro provider di rete (network provider) abbia due name server e che voglia farveli usare, con numeri IP `10.0.0.1` e `10.1.0.1`. Allora nel vostro file `named.conf`, all'interno della sezione d'apertura chiamata "options" inserite queste linee:

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

C'è anche un trucco carino per le macchine dial-up che usano i forwarder, è descritto nella sezione 8 (Domande e Risposte).

Fate ripartire il vostro name server e testatelo con nslookup. Dovrebbe funzionare bene.

3.3 Congratulazioni

Adesso sapete come impostare una versione con cache di named. Prendetevi una birra, del latte o qualunque cosa vi piaccia per celebrare l'evento.

4 Un *semplice* dominio

4.1 Ma prima un po' di teoria

Prima di iniziare *veramente* con questa sezione vi proporrò un po' di teoria e un esempio su come il DNS lavora. E voi dovrete leggerlo perché vi sarà utile. Se non ne avete voglia dovrete almeno dargli uno sguardo veloce. Fate invece attenzione alle parti che dovrebbero andare nel vostro file `named.conf`.

DNS è un sistema gerarchico, strutturato ad albero. L'apice è indicato come '`.`' e pronunciato 'root'. Al di sotto di `.` c'è un gran numero di Top Level Domains (TLD), i più noti sono `ORG`, `COM`, `EDU` and `NET`, ma ce ne sono molti altri. Proprio come in un albero esso ha la radice e si dirama verso l'esterno. Se avete qualche conoscenza d'informatica riconoscerete nel DNS un albero di ricerca, e scoprirete nodi, nodi-foglia e spigoli.

Quando comincia la ricerca di una macchina la richiesta procede ricorsivamente nella gerarchia, iniziando dall'apice. Se volete trovare `prep.ai.mit.edu` il vostro name server dovrà prima scoprire quale name server gestisce `edu`. Esso dunque chiede a uno dei `.` server (sa già quali sono i server `.` , è a questo che serve il file `root.hints`), e il `.` fornisce una lista dei server `edu`:

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

Iniziate a interrogare un name server:

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

Impostate il tipo di interrogazione (query type) su NS (name server records):

```
> set q=ns
```

interrogate su `edu`:

```
> edu.
```

Il . finale qui è significativo, dice a `nslookup` che la nostra richiesta è relativa a `edu` è direttamente sotto a `.` (e non sotto un altro dominio presente nel `search`, questo velocizza la ricerca)

```
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
```

Questo significa che tutti i server `ROOT-SERVERS.NET` risolvono `EDU.`, così potremo interrogare uno qualunque di essi. Continueremo a usare il `C`. Adesso vogliamo sapere chi risolve il prossimo livello del nostro nome di dominio: `mit.edu.`:

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

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

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

`steawb`, `w20ns` e `bitsy` risolvono `mit.edu`, ne sceglieremo uno e lo interrogheremo sul prossimo livello ancora: `ai.mit.edu`:

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

I nomi di host non sono case sensitive, ma io uso il mouse per tagliare e incollare così come vengono dallo schermo.

```
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
```

Quindi `muesli.ai.mit.edu` è un nameserver per `ai.mit.edu`:

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

Adesso cambio il tipo di interrogazione (query): abbiamo trovato il name server e quindi vogliamo chiedere tutto ciò che `muesli` sa a proposito di `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
```

E così partendo da `.` abbiamo scoperto i name server successivi per ogni livello nel nome di dominio. Se usavate il vostro server DNS invece di tutti questi altri, il vostro named avrebbe naturalmente messo nella propria cache tutte le informazioni trovate durante la ricerca, e per un bel pezzo non le avrebbe più richieste.

Nell'analogo albero ogni "`.`" del nome rappresenta un punto di diramazione. E le parti che stanno tra i "`.`" sono i nomi dei singoli rami dell'albero.

Abbiamo scalato l'albero scegliendo un nome a piacere (`prep.ai.mit.edu`), prima abbiamo scoperto la radice (`.`) e poi siamo andati alla ricerca del prossimo ramo da scalare, nel nostro caso `edu`. Una volta trovato questo, l'abbiamo scalato passando per il server che conosceva questa parte di nome. Dopo abbiamo ricercato il ramo `mit` sopra il ramo `edu` (per avere `mit.edu`) e abbiamo scalato anch'esso sfruttando il server che conosceva `mit.edu`. Ancora, abbiamo cercato `ai.mit.edu` come prossimo ramo, e per scalarlo abbiamo sfruttato il server che lo conosceva. Adesso siamo arrivati al giusto server, al giusto punto di diramazione. L'ultimo passo da fare è scoprire `prep.ai.mit.edu`, ma è molto semplice. In informatica solitamente si dice che `prep` è una *foglia* dell'albero.

Un dominio poco discusso in precedenza è `in-addr.arpa`. Anch'esso è gestito come i domini normali. `in-addr.arpa` ci permette di ricavare il nome dell'host quando abbiamo il suo indirizzo. Una cosa importante da notare qua è che gli indirizzi IP sono scritti in ordine inverso nel dominio `in-addr.arpa`. Se avete un indirizzo di una macchina: `192.128.52.43` named procede come nell'esempio `prep.ai.mit.edu`: scopre il server `arpa.`. Scopre il server `in-addr.arpa.`, scopre il server `192.in-addr.arpa.`, scopre il server `128.192.in-addr.arpa.`, scopre il server `52.128.192.in-addr.arpa.`. Scopre i record richiesti per ricavare il nome di `43.52.128.192.in-addr.arpa.`. Geniale ehh?? (dite 'Sì') Tuttavia, a livello teorico, la conversione dei numeri potrà risultare difficoltosa per anni.

Ho detto una bugia. DNS non funziona precisamente come ho descritto. Ma comunque in maniera simile a questa.

4.2 Il nostro dominio

Adesso definiremo il nostro dominio. Stiamo per creare il dominio `linux.bogus` e per definire le macchine in esso. Utilizzo nomi di dominio completamente fasulli per essere sicuro di non disturbare nessuno. (Nessuno là fuori.)

Un'ultima cosa prima di iniziare: non tutti i caratteri sono permessi nei nomi degli host. Siamo limitati ai caratteri dell'alfabeto inglese: a-z, numeri: 0-9 e al carattere '-' (dash, trattino). Ricordatevelo. Maiuscole e minuscole hanno lo stesso valore per il DNS, così `pat.uio.no` è identico a `Pat.UiO.No`.

Abbiamo già iniziato questa parte con questa linea in `named.conf`:

```
zone "0.0.127.in-addr.arpa" {
        type master;
```

```
        file "pz/127.0.0";
};
```

Per favore notate in questo file l'assenza del '`.`' alla fine dei nomi del dominio. Questo significa che ora definiremo la zona `0.0.127.in-addr.arpa`, che siamo il master server per essa e che essa è descritta nel file chiamato `pz/127.0.0`. Abbiamo già impostato questo file:

```
@               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.
```

Per favore notate in questo file il '`.`' alla fine di tutti i nomi di dominio completi, in contrasto col file `named.conf` di prima. Alcune persone hanno l'abitudine di iniziare ogni file relativo a una zona con una direttiva `$ORIGIN`, ma questo è superfluo. L'origine (che appartiene alla gerarchia del DNS) di un file zona è specificata nella sezione delle zone nel file `named.conf`, in questo caso è `0.0.127.in-addr.arpa`.

Questo 'file di zona' contiene 3 'record di risorse' (RR): un RR SOA, un RR NS e un RR PTR. SOA è l'acronimo di Start Of Authority (Inizio dell'Autorità). La '@' è una notazione speciale che indica l'origine, e poiché la colonna 'dominio' di questo file dice 0.0.127.in-addr.arpa la prima linea in realtà significa:

```
0.0.127.in-addr.arpa.   IN      SOA ...
```

NS è il RR Name Server. Non c'è la '@' all'inizio di questa linea: è implicita perché l'ultima linea comincia con la '@'. Questo fa risparmiare un po' di battute sulla tastiera. In questo modo la line NS può essere scritta:

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

Essa dice al DNS quale macchina è il name server per il dominio `0.0.127.in-addr.arpa`: è appunto `ns.linux.bogus`. È consuetudine associare a 'ns' la parola name server, analogamente ai web server che di solito sono associati a `www.`*qualcosa*; il nome può però essere qualunque.

E alla fine il record PTR dice che l'host all'indirizzo 1 nella sottorete tt/0.0.127.in-addr.arpa/, ovvero 127.0.0.1 è chiamato `localhost`.

Il record SOA è il preambolo di *tutti* i file di zona, e deve esisterne esattamente uno in ogni file di zona. Questo record descrive la zona, da dove esso viene (una macchina chiamata `ns.linux.bogus`), chi è responsabile per i suoi contenuti (`hostmaster@linux.bogus`, dovrete inserire il vostro indirizzo e-mail qui), quale è la versione del file di zona (serial: 1), e altre cose che riguardano il server DNS secondario o quello che fa da cache. Per il resto dei campi (refresh, retry, expire e minimum) usate pure i valori indicati in questo HOWTO e sarete a posto.

Adesso fate ripartire named (il comando è `ndc restart`) e usate nslookup per esaminare cosa avete fatto:

```
$ 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
```

si ottiene `localhost` da 127.0.0.1, bene. Ora per il nostro scopo principale, il dominio `linux.bogus`, inserite una nuova 'zona' in `named.conf`:

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

Notate ancora l'assenza del '`.`' finale nel nome del dominio nel file `named.conf`.

Nel file di zona `linux.bogus` metteremo dei dati completamente fasulli:

```
;
; File di zona per linux.bogus
;
; File di zona completo
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; numero di serie, data di oggi + # di serie di oggi
                        8H              ; refresh, secondi
                        2H              ; retry, secondi
                        1W              ; expire, secondi
                        1D )            ; minimum, secondi
;
                NS      ns              ; Indirizzo Inet del name server
                MX      10 mail.linux.bogus     ; Mail Exchanger Primario
                MX      20 mail.friend.bogus.   ; Mail Exchanger Secondario
;
localhost       A       127.0.0.1
ns              A       192.168.196.2
mail            A       192.168.196.4
```

Bisogna notare due cose a proposito del record SOA. `ns.linux.bogus` *deve* essere una macchina effettiva con un record A. Non è legale avere un record CNAME per la macchina indicata nel record SOA. Il suo nome non deve essere per forza 'ns', può essere un qualsiasi nome legale di host. La seconda cosa, hostmaster.linux.bogus deve essere interpretato come hostmaster@linux.bogus, questo dovrà essere un alias per un indirizzo email vero, o una mailbox, purché la/le persona/e che mantengono il DNS leggano la posta frequentemente. Ogni mail che riguarda il dominio sarà spedita all'indirizzo indicato in questo record. Il nome non deve essere per forza 'hostmaster', potrà essere un normale indirizzo email, di solito ci si aspetta che 'hostmaster' funzioni a dovere (cioè che la posta indirizzata ad esso arrivi da qualche parte).

C'è un nuovo tipo di RR in questo file, MX o RR Mail eXchanger. Esso indica ai sistemi adibiti allo smistamento della posta dove mandarla, quando è ad esempio indirizzata a `qualcuno@linux.bogus`; nel nostro caso si tratta di `mail.linux.bogus` o `mail.friend.bogus`. Il numero che precede ogni nome di macchina indica la priorità del RR MX. Il RR con il numero più basso (10) è quello che indica il mail server al quale, se possibile, deve essere mandata la posta per primo. Se non funzionasse la posta potrà essere spedita a un server con un numero più alto, un mail server secondario, ovvero `mail.friend.bogus` che appunto ha priorità 20 nel nostro caso.

Fate ripartire named con `ndc restart`. Esaminate i risultati con 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
```

Con un accurato esame scoprirete un bug (un errore). La linea

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

è sbagliata. Dovrebbe essere:

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

Ho deliberatamente fatto quest'errore in modo che impariate da esso :-). Cercando nel file di zona scopriremo che alla linea

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

manca un punto. Oppure che ha un 'linux.bogus' di troppo. Se un nome di macchina non finisce con il punto nel file di zona, l'origine viene aggiunta alla sua fine causando il doppione `linux.bogus.linux.bogus`. Dunque sia:

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

---

oppure

```
                MX      10 mail                 ; Mail Exchanger Primario
```

è corretto. Io preferisco il secondo modo perché è più breve da scrivere. Ci sono alcuni esperti di bind che non approvano, altri invece sì. Quindi in un file di zona il dominio dovrebbe essere scritto per intero e fatto terminare da un '`.`' o dovrebbe essere escluso del tutto, in questo caso verrebbe sostituito dall'origine.

Devo stressarvi con la storia del file named.conf, *non* ci devono essere '`.`' alla fine dei nomi di dominio. Non avete idea di quante volte un '`.`' di troppo (o la sua assenza) abbia ingarbugliato le cose e confuso delle indiavolate persone.

Dunque ecco qua il nuovo file di zona, con qualche informazione extra messa nel modo giusto:

```
;
; File di zona per linux.bogus
;
; Il file di zona completo
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; # di serie, data di oggi + # di serie di oggi
                        8H              ; refresh, secondi
                        2H              ; retry, secondi
                        1W              ; expire, secondi
                        1D )            ; minimum, secondi
;
                TXT     "Linux.Bogus, il proprio consulente DNS"
                NS      ns              ; Indirizzo Inet del name server
                NS      ns.friend.bogus.
                MX      10 mail         ; Mail Exchanger Primario
                MX      20 mail.friend.bogus. ; Mail Exchanger Secondario

localhost       A       127.0.0.1

gw              A       192.168.196.1
                HINFO   "Cisco" "IOS"
                TXT     "Il 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"
```

Ci sono un sacco di nuovi RR qua: HINFO (Host Information) è composto da due parti, è buona cosa dividerle entrambe con le virgolette. La prima parte indica l'hardware o la CPU della macchina, e la seconda parte il software o il S(istema)O(perativo) della macchina. La macchina chiamata 'ns' ha una CPU Pentium e Linux 2.0 come SO. CNAME (Canonical Name) è un modo per dare a ogni macchina diversi nomi. Così www è un alias per ns.

L'utilizzo del record CNAME è un po' controverso. Ma è bene seguire la regola per cui un record MX, CNAME o SOA non dovrebbero *mai* riferirsi a un record CNAME, dovrebbero far rifimento soltanto a qualcosa che abbia un record A, cioè non è auspicabile avere

```
foobar          CNAME   www                     ; NO!
```

ma è corretto avere

```
foobar          CNAME   ns                      ; SÌ!
```

È anche consigliabile assumere che un CNAME non è un nome di host legale per un indirizzo email: `webmaster@www.linux.bogus` è un indirizzo email illegale generato dall'impostazione errata di cui sopra. Anche se per voi funziona, qualche amministratore di server di posta vi farà rispettare questa regola. Il modo per impedire tutto questo è usare un record A (o anche un record tipo MX):

```
www             A       192.168.196.2
```

Alcuni maghi-dell'architettura-di-bind raccomandano di *non* usare CNAME per nulla. Ma la discussione sul perché o sul perché no va oltre questo HOWTO.

Ma coma potrete notare, questo HOWTO e molti siti non seguono questa regola.

Caricate il nuovo database facendo `ndc reload`, questo imporrà a named di rileggere i suoi file.

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

> ls -d linux.bogus
```

Questo fa sì che vengano elencati tutti i record, risulta così:

```
[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       "Il 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"
```

È buono. Come potete vedere somiglia un sacco allo stesso file di zona. Vediamo cosa dice per `www` da solo:

```
> 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
```

In altre parole, il vero nome di`www.linux.bogus` è `ns.linux.bogus`, e vi da qualche informazione a proposito di ns, abbastanza per connettervi ad esso se voi foste un programma.

Ora siamo a metà strada.

4.3 La zona inversa (reverse zone)

Adesso i programmi possono convertire i nomi di linux.bogus negli indirizzi a cui vorrebbero connettersi. Ma c'è bisogno anche della zona inversa, un DNS tale da poter covertire un indirizzo in un nome. Questo nome è utile a un sacco di server di diversi tipi (FTP, IRC, WWW e altri) per decidere se colloquiare con voi o meno, e se si, anche quanta priorità dovrà essere assegnata loro. Per un pieno accesso a tutti i servizi su Internet è richiesta una zona inversa.

Mettete questo in `named.conf`:

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

È esattamente come per `0.0.127.in-addr.arpa`, e i contenuti sono simili:

```
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151 ; # di serie, data di oggi + # di serie di oggi
                        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.
```

Adesso fate ripartire named (`ndc restart`) e esaminate ancora il lavoro con nslookup :

```
> 192.168.196.4
Server:  localhost
Address:  127.0.0.1

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

pare OK, elencate tutto per fare un esame accurato:

```
> 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
```

Sembra buono! Se il vostro output non viene simile a questo guardate i messaggi d'errore nel vostro syslog. Ho spiegato come farlo all'inizio del capitolo.

4.4 Qualche parola di avvertimento

Ci sono delle cose che a questo punto vorrei aggiungere. I numeri IP usati negli esempi sono stati presi dai blocchi delle "reti private", cioè non è permesso usarli esplicitamente su Internet. Per questo sono comodi da usare in un HOWTO. La seconda cosa riguarda la linea `notify no;`. Dice a named che non deve notificare al suo server secondario (slave, schiavo) quando riceve un aggiornamento di uno dei suoi file di zona. In bind-8 named può notificare agli altri server quando riceve un aggiornamento dei file di zona. Questo è utile nell'uso comune, ma per gli esperimenti privati con le zone questa possibilità deve essere esclusa, non vogliamo che i nostri esperimenti inquinino Internet vero??

E naturalmente, questo dominio è veramente fasullo, e così tutti gli indirizzi in esso. Per un vero esempio tratto dalla vita reale leggete il prossimo capitolo.

4.5 Perché non funziona il lookup inverso (reverse lookup)

Ci sono un paio di “grattacapi” che possono essere normalmente evitati quando si fa il lookup sempre degli stessi nomi (o dei nomi per cui lo si fa più spesso) quando si imposta la zona inversa. Prima che andiate avanti c'è bisogno che il lookup inverso funzioni bene sul vostro nameserver. Se così non fosse, tornate indietro e mettetelo a posto prima di continuare.

Discuterò due problematiche del lookup inverso viste dall'esterno della vostra rete:

4.5.1 La zona inversa non è delegata

Quando chiedete a un provider di servizi un dominio e un intervallo di indirizzi di rete, il dominio è normalmente delegato di conseguenza. Una delega è quel record NS che tiene assieme il tutto, che vi permette di passare da un nameserver all'altro come è stato spiegato nella sezione teorica di sopra. L'avete letta vero? Se la zona inversa non vi funziona tornate indietro e leggetela. Ora.

Anche la zona inversa deve essere delegata. Se avete ottenuto la rete `192.168.196` con il dominio `linux.bogus` dal vostro provider di servizi, loro dovranno mettere un record `NS` nella vostra zona inversa così come per la vostra zona di forward. Se seguirete la catena partendo da `in-addr.arpa` fino alla vostra rete, probabilmente troverete un'interruzione nella catena. Molto probabilmente a livello del vostro service provider. Una volta scoperta l'interruzione contattate il provider e chiedetegli di correggere l'errore.

4.5.2 Vi hanno dato una sottorete classless

Questo sarebbe un argomento un po' avanzato, ma le sottoreti classless (senza classe) ormai sono molto comuni e probabilmente ne avete una a meno che non siate un'azienda di media grandezza.

Una sottorete classless è ciò che oggi manda avanti Internet. Qualche anno fa si è fatto molto rumore a causa della scarsità di numeri IP. Le brillanti persone che lavorano al IETF (l'Internet Engineering Task Force, sono loro che mantengono la funzionalità di Internet) unirono le loro menti e risolsero il problema. Ma bisogna pagare un prezzo. Il prezzo è che avrete meno che una sottorete di classe “C” e qualche cosa potrebbe non funzionare. Leggete per favore *Ask Mr. DNS at http://www.acmebw.com/askmrdns/00007.htm* per una buona spiegazione e per come trattare il problema.

L'avete letto? Non sto per spiegarlo quindi leggetelo.

La prima parte del problema è costituita dal fatto che il vostro ISP deve capire la tecnica descritta da Mr. DNS. Non tutti i piccoli ISP hanno una chiara visione di questa. Se sarà il caso dovrete spiegar loro come fare ed essere insistenti. Ma anche voi assicuratevi di aver capito bene prima ;-). A questo punto loro imposteranno una buona zona inversa sui loro server, e voi potrete esaminarla correttamente con nslookup.

La seconda e ultima parte del problema è costituita dal fatto che voi dovete comprendere la tecnica. Se non siete sicuri rileggetela ancora. Dopo potrete impostare le zone inverse della vostra sottorete classless come descritto da Mr. DNS.

Nei dontorni c'è un'altra trappola in agguato. I vecchi risolutori *non* sono abilitati a sfruttare il trucco del `CNAME` nella catena della risoluzione e falliranno nel tentativo di risolvere inversamente (reverse-resolving) la vostra macchina. Questo problema può portare ad assegnare al servizio una scorretta classe di accesso, all'accesso negato o a qualcosa del genere. Se inciampaste in un servizio di questo tipo l'unica soluzione (che io conosca) è che il vostro ISP inserisca direttamente nel suo file di zona truccato (il file relativo alla vostra zona classless) il vostro record PTR anziché il record CNAME truccato (è un modo per ingannare il DNS e per fargli accettare qualcosa per cui non è preparato).

Altri ISP offriranno diversi modi per risolvere questo problema, come dei form via web (Web-based) che permettono di inserire i vostri dati sulla zona inversa, oppure con altri sistemi "automagici".

5 Esempio di un vero dominio

Dove si elencano alcuni *veri* file di zona

Gli utenti mi hanno suggerito di includere un esempio reale di un dominio funzionante come l'esempio di tutorial.

Utilizzo questo esempio col permesso concessomi da David Bullock di LAND-5. Questi file risalgono al 24 Settembre 1996, successivamente vennero da me modificati perché si adattassero alle restrizioni di bind 8 e perché comprendessero delle estensioni. Questo implica che quello che leggerete qua differisce un po' da quello che otterreste facendo una query sul nameserver di LAND-5.

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

Qui troveremo le sezioni relative alle due zone inverse richieste: la rete 127.0.0, e la sottorete LAND-5 `206.6.177`. Poi c'è la linea relativa alla zona di forward per land-5, `land-5.com`. Si noti come i file siano stati sistemati nella directory chiamata `zone` anziché in `pz` come ho fatto in questo HOWTO.

```
// 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";
};
```

Se aveste intenzione di usare queste righe nel vostro named.conf (ma solo per gioco) **PER FAVORE** mettete "`notify no;`" nelle sezioni relative alle due zone `land-5` così da evitare incidenti.

5.2 /var/named/root.hints

Tenete a mente che questo è un file dinamico, e quello che c'è qua sarà vecchio. È meglio che ne procuriate uno recente, con dig, come verrà presto spiegato.

```
; <<>> 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

Solo lo stretto necessario, il record obbligatorio SOA, e un record che mette in corrispondenza 127.0.0.1 con `localhost`. Entrambi sono richiesti. Non deve esserci nient'altro in questo file. Probabilmente non ci sarà mai bisogno di aggiornare questo file, a meno che non cambino gli indirizzi del nameserver o hostmaster.

```
@               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

Qui possiamo vedere il record obbligatorio SOA e i record NS richiesti. Si può vedere che è presente un nameserver secondario in `ns2.psi.net`. Questo è come dovrebbe essere, è bene avere *sempre* un server secondario fuori dalla vostra rete che faccia da backup. Si può notare anche la presenza di un host principale (master host) chiamato `land-5` che si prende cura della maggior parte dei servizi Internet, e questo è fatto tramite i record CNAME (alternativamente si possono usare i record A).

Come si può vedere dal record SOA, il file di zona comincia con `land-5.com`, la persona da contattare è `root@land-5.com`. Anche `hostmaster` è spesso utilizzato per il responsabile di zona. Il numero seriale è nel formato standard yyyymmdd (aaaammgg) con il numero che indica il giorno (todays serial number) a seguire. Questa è forse la sesta versione del file di zona del 20 settembre 1996. Ricordate che il serial number *deve* essere incrementato in maniera monotonica, qui c'è solo *una* cifra per i todays serial #, così dopo 9 volte che si è editato il file bisogna aspettare il giorno successivo prima che si possa editarlo di nuovo. Comunque considerate che si possono usare 2 cifre.

```
@       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
; {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
```

Esaminando i nameserver di land-5 scoprirete che gli host hanno un nome del tipo `ws_`*numero*. Con le ultime versioni di bind-4 named ha iniziato a porre delle restrizioni sui caratteri che potevano essere usati nei nomi di host. In questo HOWTO io ho sostituito '-' (dash, trattino) con '_' (underline) in modo da uniformarmi alle regole di bind-8 per i nomi di host.

Un'altra cosa da notare è che le workstation non hanno nomi individuali, ma un prefisso seguito dalle ultime due parti del numero IP. Usare delle convenzioni simili può semplificare significativamente la manutenzione, ma può risultare impersonale e in effetti potrebbe anche irritare i vostri clienti.

Vediamo anche che `funn.land-5.com` è un alias per `land-5.com`, ma ciò è fatto tramite un record A, non con un record CNAME. Questo è un buon modo di procedere.

5.5 /var/named/zone/206.6.177

Commenterò questo file più sotto.

```
@               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.
; {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 zona inversa costituisce quella fase della configurazione che causa più grane. Essa serve a ricavare il nome di un host dall'indirizzo IP di una macchina. Esempio: voi siete un server IRC e accettate connessioni dai client IRC. Inoltre siete un server IRC Norvegese a volete che queste connessioni provengano da client

Norvegesi o da altre nazioni Scandinave. Quando ricevete una richiesta di connessione da un client la libreria C è in grado di dirvi il numero IP della macchina che sta tentando la connessione, poiché il numero IP del client è contenuto in ogni pacchetto che attraversa la rete. A questo punto potrete chiamare una funzione chiamata gethostbyaddr che ricava il nome di un host a partire dal suo indirizzo IP. Gethostbyaddr interrogherà un server DNS, il quale attraverserà il DNS in cerca della macchina. Supponiamo che il client si connetta da ws-177200.land-5.com. La libreria C presente nel server ricava il numero IP del client che tenta la connessione, in questo caso è 206.6.177.200. Per scoprire il nome di questa macchina bisogna prima scoprire `200.177.6.206.in-addr.arpa`. Il server DNS troverà prima i server `arpa.`, poi troverà i server `in-addr.arpa.`, seguendo il percorso inverso passando per 206, poi per 6 e alla fine troverà il server responsabile per la zona `177.6.206.in-addr.arpa` presso LAND-5. Da questo finalmente si potrà ricavare che in `200.177.6.206.in-addr.arpa` c'è un record "`PTR ws-177200.land-5.com`", e questo significa che il nome associato a `206.6.177.200` è `ws-177200.land-5.com`. Come è stato detto per la spiegazione della ricerca (looking up) di `prep.ai.mit.edu`, anche questa è leggermente fittizia.

Riprendiamo l'esempio del server IRC. Il server IRC accetta connessioni solo da paesi Scandinavi, ovvero `*.no`, `*.se`, `*.dk` il nome `ws-177200.land-5.com` non corrisponde a nessuno di questi ovviamente, e il server negherà la connessione. Se *non* ci fosse la corrispondenza inversa (reverse mapping) di `206.2.177.200` tramite la zona `in-addr.arpa` il server sarebbe incapace di scoprire il nome e avrebbe dovuto comparare `206.2.177.200` con `*.no`, `*.se` e `*.dk`, senza trovare nessuna corrispondenza.

Alcune persone vi diranno che il mapping del lookup inverso è importante solo per i server, o per nulla importante. Non è così: molti server ftp, news, IRC e anche qualche server http (WEB) *non* accetteranno connessioni da macchine per le quali non riescono a trovare il nome. E così il mapping inverso diventa di fatto *obbligatorio*.

6 Manutenzione

Mantenerlo operativo

C'è un altro compito di manutenzione che dovete fare sui named, oltre a quello di tenerli operativi. Consiste nel mantenere aggiornato il file `root.hints`. Il modo più semplice è usare dig, fate partire dig senza argomenti, otterrete `root.hints` in accordo col quello che c'è sul vostro server. Poi fate una richiesta a uno dei root server elencati con `dig @rootserver`. Vedrete che l'output somiglierà terribilmente al file `root.hints`. Salvatelo in un file (`dig @e.root-servers.net . ns >root.hints.new`) e rimpiazzate il vecchio `root.hints` con questo.

Ricordate di fare reload di named dopo aver rimpiazzato il cache file.

Questo script mi è stato mandato da Al Longyear, può essere fatto partire automaticamente per aggiornare `root.hints`, impostate una riga nel crontab che lo faccia partire una volta al mese e dimenticatelo. Lo script assume che il vostro sistema di posta funzioni e che l'alias di posta 'hostmaster' sia definito. Dovrete editarlo perché si conformi alle vostre esigenze.

```
#!/bin/sh
#
# Update the nameserver cache information file once per month.
# This is run automatically by a cron entry.
#
# Original by Al Longyear
# Updated for bind 8 by Nicolai Langfeldt
# Miscelanious error-conditions reported by David A. Ranch
# Ping test suggested by Martin Foster
```

```
#
(
 echo "To: hostmaster <hostmaster>"
 echo "From: system <root>"
 echo "Subject: Automatic update of the root.hints file"
 echo

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

 # Are we online?  Ping a server at your ISP
 case `ping -qnc 1 some.machine.net` in
   *'100% packet loss'*)
        echo "The network is DOWN. root.hints NOT updated"
        echo
        exit 0
        ;;
 esac

 dig @e.root-servers.net . ns >root.hints.new 2>&1

 case `cat root.hints.new` in
   *NOERROR*)
        # It worked
        :;;
   *)
        echo "The root.hints file update has FAILED."
        echo "This is the dig output reported:"
        echo
        cat root.hints.new
        exit 0
        ;;
 esac

 echo "The root.hints file has been updated to contain the following
information:"
 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 "The nameserver has been restarted to ensure that the update is complete."
 echo "The previous root.hints file is now called
/var/named/root.hints.old."
```

```
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

Qualcuno di voi potrebbe aver notato che il file `root.hints` è disponibile anche in ftp da Internic. Per favore, non usate ftp per aggiornare `root.hints`, il metodo sopra descritto è molto più amichevole per la rete, e per Internic.

7 Convertire dalla versione 4 alla versione 8

Questa era originariamente una sezione sull'uso di bind 8, scritta da Davie E. Smith (dave@bureau42.ml.org). L'ho rivista in modo da conformarla al nuovo nome della sezione.

Non c'è molto. Eccetto che per l'uso di named.conf al posto di named.boot, tutto è identico. E bind8 è fornito con uno script perl che converte i file vecchio stile nel nuovo. Esempio di named.boot (vecchio stile) per un name server che fa solo da cache (caching-only):

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

Dalla linea di comando, nella directory bind8/src/bin/named (*si assume che abbiate la distribuzione dei sorgenti. Se avete il pacchetto dei binari lo script sarà da qualche parte, comunque non so esattamente dove dovrebbe stare. -ed-*), digitate:

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

Il quale crea 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";
};
```

Questo script funziona per tutto ciò che potrebbe stare dentro un file `named.boot`, sebbene non aggiunga tutti i miglioramenti e le nuove opzioni di configurazione che bind8 supporta. Qui c'è un `named.conf` più completo, che fa le stesse cose ma che è appena più efficiente.

```
// This is a configuration file for named (from BIND 8.1 or later).
// It would normally be installed as /etc/named.conf.
// The only change made from the 'stock' named.conf (aside from this
// comment :) is that the directory line was uncommented, since I
// already had the zone files in /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";
};
```

Trovate questo file nella distribuzione di bind8, nella directory bind8/src/bin/named/test, insieme a copie dei file di zona, che possono essere prese e usate immediatamente.

I formati dei file di zona e `root.hints` sono identici, così come i comandi per aggiornarli.

8 Domande e risposte

Per favore leggete questa sezione prima di scrivermi in email.

1. Il mio named vuole il file named.boot

   State leggendo l'HOWTO sbagliato. Leggete per favore la vecchia versione di questo HOWTO, che tratta bind 5, presso *http://www.math.uio.no/˜janl/DNS/*

2. Come si usa il DNS dall'interno di un Firewall?

   Un indizio: `forward only;` probabilmemte avrete bisogno anche della riga

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

all'interno della parte “options” del file `named.conf` come suggerito nella sezione-esempio 3 (Un name server caching only.)

3. Come fare in modo che il DNS distribuisca un servizio attraverso gli indirizzi disponibili, tipo `www.sito.occupato` per ottenere un effetto di bilanciamento, o simile??

   Create numerosi record **A** per `www.sito.occupato` e usate bind 4.9.3 o successivo. Poi sarà bind a far ruotare le risposte. *Non* funziona con le precedenti versioni di bind.

4. Voglio impostare il DNS su una intranet (chiusa). Cosa devo fare?

   Cancellerete il file `root.hints` e farete solo i file di zona. Questo significa anche che non dovrete aggiornare i file di hint tutte le volte.

5. Come si imposta un name server secondario (slave)?

   Se il server primario/master ha indirizzo 127.0.0.1 mettete una linea come questa nel file named.conf del vostro secondario:

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

   Potrete elencare numerosi master server alternativi, la zona può essere copiata da dentro la lista `masters`, separata da ';'.

6. Vorrei bind in esecuzione quando mi disconnetto dalla rete.

   Ci sono quattro articoli che riguardano questo:

   - Specifico per bind 8, Adam L. Rice mi ha mandato questa email, su come far funzionare tranquillamente il DNS su una macchina dial-up:

```
Ho scoperto che con le ultime versioni di BIND questo [<em/mischiare i
file, -ed/] non è più necessario. C'è una direttiva "forward" oltre a
quella "forwarders" che controlla come queste vengono
usate. L'impostazione di default è "forward first", la quale prima
chiede a ognuno dei forwarder, e poi se il tentativo fallisce, prova
da sola a fare il lavoro seguendo un normale approccio. Questo
implica un normale comportamento di gethostbyname() e impiega una
quantità spropositata di tempo quando il link non è attivo. Ma se
"forward only" è l'impostazione scelta, allora BIND si blocca quando
non riesce a ottenere una risposta dai forwarders, e gethostbyname()
si ferma immediatamente. Quindi in questo caso non c'è bisogno di
smanettare con i file di /etc e di far ripartire il server.

Per quanto mi riguarda, ho solo aggiunto le linee

forward only;
forwarders { 193.133.58.5; };

nella sezione opzioni { } del mio file named.conf. Tutto ciò funziona
veramente bene. L'unico svantaggio è che questo riduce un software
```

```
incredibilmente complicato per il DNS allo stato di una stupida cache.
In un certo senso, io vorrei solo far fuzionare una stupida cache al
posto del DNS, ma ciò non sembra essere altro che un pezzo di software
disponibile per Linux.
```

- Ho ricevuto questa mail da Ian Clark <ic@deakin.edu.au> dove egli spiega come affronta il problema:

```
Faccio partire named sulla mia macchina che fa servizio di 'Masquerading'
(mascheramento). Ho due file root.hints, uno chiamato root.hints.real che
contiene i veri nomi dei root server e l'altro chiamato root.hints.fake
che contiene...

----
; root.hints.fake
; this file contains no information
----

Quando vado off line copio il file root.hints.fake su root.hints e
faccio ripartire named.

Quando vado on line copio root.hints.real su root.hints e faccio
ripartire named.

Tutto ciò viene eseguito da ip-down e ip-up rispettivamente.

La prima volta che faccio una richiesta (query) off line, named non
avendo dettagli su di essa lascia una riga simile a questo messaggio...

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

con la quale posso convivere.

Questo per me funziona certamente. Posso usare il nameserver per le
macchina locali quando sono fuori dalla rete senza il ritardo del
timeout per i nomi di dominio esterni e mentre sono collegato alla rete
le richieste (query) per i domini esterni funzionano normalmente.
```

- Ho ricevuto anche informazioni su come bind intergisce con NFS e con il portmapper su una macchina generalmente offline da Karl-Max Wanger:

```
Sono abituato a eseguire il mio named su tutte le mie macchine che
solo occasionalmente sono connesse a Internet via modem. Solo il
nameserver fa da cache, esso non ha un'area di autorità e chiede
qualunque cosa ai nameserver del file root.cache. Come è prassi comune
con Slackware, viene fatto partire prima di nfsd e mountd.

Con una delle mie macchine (un notebook Libretto 30), avevo il problema
che qualche volta avrei voluto fare il mount di essa da un altro sistema
connesso alla mia rete locale, ma la maggior parte delle volte non
funzionava. Avevo lo stesso effetto usando indistintamente PLIP, una
scheda ethernet PCMCIA o il PPP su una interfaccia seriale.

Dopo qualche supposizione e esperimento scoprii che apparentemente
named incasinava il processo di registrazione con portmapper che nfsd
e mountd devono fare all'avvio (faccio partire questi demoni all'avvio
come al solito). Eseguire named dopo nfsd e mountd elimina completamente
questo problema.
```

```
Anche se non ci sono svantaggi da attendersi da una sequenza di boot così
modificata, vorrei consigliare a tutti di farla in questo modo per prevenire
potenziali problemi.
```

- Infine, ci sono delle informazioni stile HOWTO su questo argomento presso *Ask Mr. DNS at http://www.acmebw.com/askmrdns/#linux-dialup*. Siccome sono a proposito di bind 4, dovrete adattare quello che dice a bind 8.

7. Il caching name server memorizza la sua cache? C'è un modo per controllare la dimensione della cache?

   La cache è completamente immagazzinata in memoria, *non* verrà mai scritta sul disco in nessuna occasione. Ogni volta che si blocca named (con il comando 'kill') la cache è persa. La cache *non* è controllabile in alcun modo. Named la gestisce in accordo a qualche semplice regola e basta. Non potete controllare la cache o la sua dimensione in nessun modo per nessuna ragione. Se questo non vi andasse bene potrete sempre cercare di modificare named andando ad agire sul codice sorgente di esso. Non è comunque un'operazione raccomandabile.

8. Named salva la cache fra un riavvio e l'altro? Posso fare in modo che la salvi?

   No, named *non* salva la cache quando si ferma. Questo significa che la cache deve essere ricostituita ogni volta che fermate e fate ripartire named. *Non* c'è modo di salvare la cache in un file. Se questo non vi andasse bene potrete sempre cercare di modificare named andando ad agire sul codice sorgente di esso. Non è comunque un'operazione raccomandabile.

9. Come si ottiene un dominio? Io vorrei impostare il mio dominio chiamato (ad esempio) `linux-rules.net`. Come posso avere il dominio che vorrei mi fosse assegnato?

   Contattate per favore il vostro provider di servizi di rete. Loro sapranno aiutarvi in questo. Sappiate che in molte parti del mondo ci sarà bisogno di pagare per avere un dominio.

9 Come diventare un grande amministratore DNS

Documentazione e strumenti

Esiste della vera documentazione. Sia online che su carta stampata. La lettura di buona parte di essa è necessaria per fare il passo dall'amministratore DNS a tempo perso a quello a tempo pieno. Su carta stampata il libro è *DNS and BIND* di C. Liu and P. Albitz edito dalla O'Reilly & Associates, Sebastopol, CA, ISBN 0-937175-82-X. Io l'ho letto, è eccellente, anche se basato su bind 4, ma questo non è un grosso problema. C'è anche una sezione sul DNS sul libro *TCP/IP Network Administration*, di Craig Hunt edito dalla O'Reilly..., ISBN 0-937175-82-X. Un altro "must" per la Buona amministrazione DNS (e buono per qualsiasi altra cosa) è *Zen and the Art of Motorcycle Maintenance (Lo Zen e l'arte della manutenzione della motocicletta)* di Robert M. Pirsig :-) Disponibile come ISBN 0688052304 e altri.

Online troverte della roba presso `<http://www.dns.net/dnsrd/>` (DNS Resources Directory), `<http://www.isc.org/bind.html>`; una FAQ, un manuale di riferimento (BOG; Bind Operations Guide) oltre che documenti, definizioni di protocolli e trucchi (hacks) sul DNS (di questi, la maggior parte, se non tutti, e alcune delle RFC elencate sotto, sono anche contenuti nella distribuzione di bind). Io non ho letto tutto, e infatti non sono un amministratore DNS a tempo pieno. Arnt Gulbrandsen invece ha letto la BOG ed è rimasto meravigliato da essa :-). Il newsgroup *comp.protocols.tcp-ip.domains* è relativo al DNS. Inoltre come aggiunta a tutto questo ci sono numerose RFC sul DNS, le più importanti sono probabilmente queste:

RFC 2052

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

RFC 1918

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

RFC 1912

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

RFC 1912 Errors

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

RFC 1713

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

RFC 1712

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

RFC 1183

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

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.