第5章　Linux的网络设置

在安装好Linux并保证其正常运作以后，我们开始设置网络。我们将探讨：

- 配置NFC服务器
- 配置PPP服务器
- 配置DNS服务器

5.1　配置NFC服务器

5.1.1　NFS简介

Network File System(NFS)是由SUN公司发展，并于1984年推出的。NFS是一个RPC 服务，它使我们能够共享文件。它的设计是为了在不同的系统间使用，所以它的通信协议设计与主机及操作系统无关。当使用者想用远程文件时，只要用 mount就可把远程的文件系统挂接在自己的文件系统之下，使得远程的文件使用上和本地机器的文件没两样。

图5-1为NFS结构示意图。

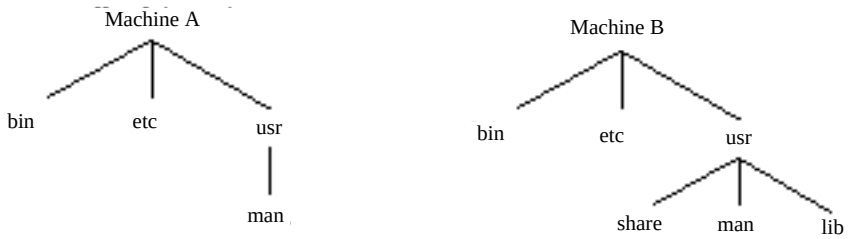

图　5-1

假如我们在机器A上，要把机器B上的/usr/man挂接到机器A的/usr/man，只要：

```
mount   machine_name:/usr/man   /usr/home
```

就可mount过来。而我们不仅可以 mount目录，也可以是一个文件。在挂接之后我们只能对文件做读(或写)的操作，而不能在远程机器上把此文件或目录移动、删除，但须注意的是，我们 mount /usr 后，不能再mount /usr下面的目录，否则会发生错误。

NFS就是使Servers上的文件能被其他的机器 mount，而达到资源共享的目的，共享这些文件的机器就可称为客户机。一个客户机可以从服务器上 mount一个文件或是一个层次的目录(file hierarchies)。然而事实上任何一台机器都可以做 NFS server或NFS client，甚至同时为NFS server和NFS client也可以。

NFS服务器所输出 (export)的文件或目录都记录在 /etc/exports这个文件中，当我们启动NFS服务器时，/etc/rc.local这个批处理脚本文件会自动启动 exportfs这个程序，搜寻/etc/exports这一个文件是否存在，并且赋予正确的权限给所有 export出去的目录树。

但须注意的是，只有服务器所 export出去的路径，NFS 客户才能够mount，同样的，当启

动客户端程序时，系统会自动去 mount所有服务器export的路径，而mount到的所有路径都会记录在/etc/fstab下，类似如下的fstab文件：

```
/dev/sd0a  /        4.2 rw 1 1
/dev/sd0h  /tmp     4.2 rw 1 3
/dev/sd0g  /usr     4.2 rw 1 2
/dev/fd0   /pcfs    pcfs rw,noauto 0 0
sparc20：/swap /swap        nfs rw, intr, bg, soft 0 0
sparc17：/home /home        nfs rw, intr, bg, soft 0 0
sparc17：/home3 /home3      nfs rw, intr, bg, soft 0 0
sparc14：/home4 /home4      nfs rw, intr, bg, soft 0 0
sparc20：/home2 /home2      nfs rw, intr, bg, soft 0 0
sparc20：/var/spool/mail /var/spool/mail  nfs rw, intr, bg, soft 0 0
rs970：/home1 /home1        nfs rw, intr, bg, soft 0 0
```

注意 当客户mount 到一个路径，绝对不是说将服务器上的这一个路径拷贝到本地的机器上，虽然我们可以用cd进入这一个mount到的路径，就如同是使用本地的目录一样。

5.1.2 设置NFS 服务器

此过程分下面几步完成：

1) 定义机器为 NFS文件服务器。

2) 划分服务器端的磁盘，定义哪一些分区是要提供出来作为客户端所共享的文件系统。

3) 在客户表单(Client Form)上定义每一台客户机的参数。

4) 写出 /etc/exports(一般系统都有一个缺省 exports)。

5) 重新启动NFS server 或用指令exportfs -a 输出所有的目录，并且用 nfsd 8 & 启动 nfsd 守护程序，常驻在后台。

6) 关于exportfs。

• 检查 /etc/exports 输出路径的权限，确定只有 root能修改，所有的用户只能读。

• 用exportfs 去增加或删除目录：

```
exportfs -o access=engineering=dancer /usr
exportfs -u /usr
```

• 假如你的机器没有NIS(YP server)的服务，当更改资料时应修改：

```
/etc/passwd、 /etc/group、 /etc/hosts、 /etc/ethers；
```

• 为自己的网络设置安全的exportfs的语法：

```
/usr/etc/exportfs   [ -avu ]   [ -o option ]  [ directory ]
```

-a ： 把/etc/exports中所有路径export出去；

-u ： 把export出去的路径卸下， 如：

```
exportfs  -u  /usr
```

-o option：如 exportfs -o ro /usr ，所有人对 /usr 都为read only。 option 还有root = hostname，access = client， access = netgroup 。

例如：

exportfs -a 把exports中的路径全部export出去；

```
exportfs -o access=engineering=other  /usr
```

/usr 这路径export后只有engineering和other 这两个group 能够 read 和write；

```
exportfs -o access=oak,root=dancer  /usr
```

设置dancer 这台客户机对 /usr 为只读， 且只有oak这一个 group 能read；

• /etc/exports文件的范例

exports文件的语法为：

```
   directory     -option[, option]
```

下面的例子中设置了两个group能rw：

```
/usr                              -access=engineering：accounting
/home                             -access=engineering：accounting
/var/spool/mail                   -access=engineering：accounting
/export/exec/sun3                 -access=engineering：accounting
/export/exec/sun3.sunos.4.1          -access=engineering：accounting
/export/exec/kvm/sun3.sunos.4.1 -access=engineering：accounting
/export/root/birch                -access=birch, root=birch
/export/swap/birch                -access=birch, root=birch
/export/root/oak                       -access=oak, root=oak
/export/swap/oak                  -access=oak, root=oak
/export/root/willow               -access=willow, root=willow
/export/swap/willow               -access=willow, root=willow
/export/root/pine                 -accsee=pine, root=pine
/export/swap/pine                 -accsee=pine, root=pine
```

“ access=client, root=hostname ”，这样只有这一台客户机的超级用户有 rw的权力。

5.1.3 设置NFS客户机

1. 基本设置方法

基本设置方法如下：

1) 编辑好 /etc/fstab 这个文件，确定要mount的路径都在fstab中。

2) 依照fstab所设的内容，在客户机上设置好挂接点 (挂接点就是用mkdir设置的、 exports所输出的路径)。

3) 确定所要mount的路径都在 /etc/exports 中。

4) 可以启动mount去连结server上的目录(如mount -a)。

fstab文件的语法为：

```
filesystem  directory  type  options  freq  password
```

以下是一个/etc/fstab文件的范例：

```
oak：/export/root/boomer  /  nfs  rw  0  0
oak：/export/exec/sun3          /usr        nfs    ro  0  0
oak：/export/exec/kvm/sun3      /usr/kvm    nfs    ro  0  0
oak：/usr/share                 /usr/share  nfs    ro  0  0
oak：/home/oak                  /home/oak   nfs    rw, bg  0  0
```

mount 的语法：

```
mount -t type [-rv] -o [option] servername:pathname /mount_point
```

umount的用法：

```
umount mount_point
umount -a 卸下所有已经mount上的文件系统
```

常见的一些mount 失败的错误信息：

```
/etc/mtab: No such file or directory
```

mtab 这一个路径或是文件必须存在，在 mount之前。

```
mount : ...Block device required
```

远程的机器名称可能敲错了。

```
mount: ...not found  /etc/fstab
```

fstab一定要在客户机上的/etc目录下。

```
not in hosts database
```

文件/etc/hosts中没有这个hosts database，或是NIS 的服务进程ypbind没有在运行。

```
Must be root to use mount
```

一般都只有 root 才能使用mount这个命令，所以mount之前先成为superuser。

```
Stale NFS file handle
```

当我们已经mount 上的文件或目录，在服务器上突然被 remove或unexport时，就会出现此信息。

2. 使用简单的linuxconf

使用系统配置工具 linuxconf，我们可以很简单地设置好 NFS文件系统。首先，运行linuxconf。接着选择NFS菜单上的Add Mount选项。这样将会弹出一个对话框，需将以下信息填入其中：

Device: 输入主机和路径名，二者以冒号分隔。例如： foo.bar.com:/usr/exported 表示foo.bar.com. 主机上的 /usr/exported 目录。

Mount Point: 输入本机上的一个目录用以将 NFS文件系统安装于其下。例如：/mnt/foo。

Options: 输入文件系统的相关选项。默认的设置为： soft,intr,rw. rw表示文件系统是可读可写的。而 soft,intr 选项可以使你的系统在远程服务器 down掉的情况下具备更强的恢复能力。请查看相关手册以获得有关选项的详细解释。

Comment: 该可选域用于存储一条简短的注释信息。

当正确地填完所需信息以后，单击 OK按钮。一条记录就被置于 /etc/fstab中。但是文件系统并没有真正地被安装上。要安装它，在主窗口中选择该文件系统并点击 Mount按钮即可。图5-2为文件系统配置面板。

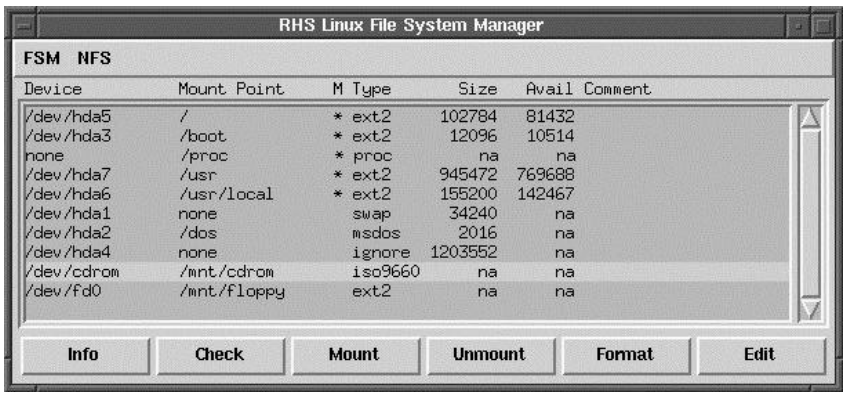

图 5-2

5.1.4 NFS的工作原理

当我们启动 NFS文件服务器时， /etc/rc.local会自动启动 exportfs这个程序，指定可以export的文件或目录，而我们所能 mount的也只能是其所指定的目录。

1. NFS是基于XDR/RPC协议的

XDR(eXternal Data Representation)，外部资料表示法。XDR提供一种方法把资料从一种格式转换成另一种标准资料格式表示法，确保在不同的机器、操作系统及程序语言中，所有资料代表的意义都是相同的。

RPC(Remote Procedure Call)，远程程序调用，请求远程机器给予服务。客户机通过网络传送RPC到远程机器，请求服务。

2. NFS 如何运用 RPC 传送数据

1) 客户送出信息，请求服务。

2) 客户stub把客户送出的参数转换成XDR标准格式，并用系统调用把信息送到网络上。

3) 信息经过网络送达远程主机系统。

4) 远程主机将接受到的信息传给服务器 stub。

5) 把XDR形式的资料，转换成符合主机端的格式，取出客户发出的服务请求参数，送给服务器；

6) (10)是服务器送出服务给客户的逆向传送过程 (见图5-3)。

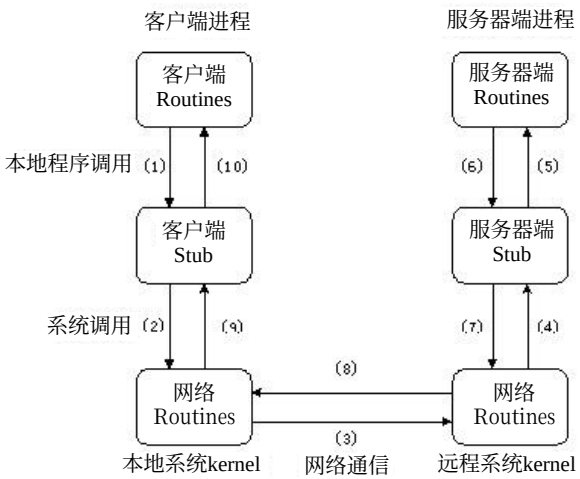

图 5-3

5.1.5 NFS守护程序的启动过程

一个NFS 服务器要inet、portmap、nfs、mount这四个守护程序保持在后台执行的状态下才能运行，如果运行NIS还要加上ypbind 服务进程。

当启动 NFS文件服务器端时， /etc/rc.local文件会做如下的动作：

1) 执行exportfs，读取服务器的/etc/exports，告诉 kernel 所要输出的文件树和存取的权限

```
exportfs  -a
```

2) 启动rpc.mountd服务进程和nfsd服务进程(通常是8个)

```
rpc.mountd  -n
nfsd 8  &
echo  -n  'nfsd'
```

当启动NFS客户端时，rc.local会做如下的动作：

1) 启动boid服务进程，处理读写的程序

```
biod 8
echo -n 'biod'
```

2) 执行

```
mount -vat nfs
```

读取client's /etc/fstab并且mount所有属于NFS类型的文件。

5.1.6 各服务进程的作用

nfsd、biod、rpc.mountd、inetd、portmap都可在/usr/etc下找到：

nfsd 根据客户端对文件系统的需求，启动文件系统请求服务进程，响应客户的请求，而一般文件系统请求服务进程的数目是8，这也就是我们在rc.local 中写“ nfsd 8 & ”的原因。

biod 此指令是在NFS客户端上用的，用来启动异步块I/O服务进程用来建立buffer cache，处理在客户机上的读写。

mountd mountd是个RPC服务器。启动rpc.mountd服务进程后，它会读取/etc/xtab查看哪一台客户机正在mount 哪一个文件系统，并回应客户机所要 mount 的路径(mountd处理的过程可用 showmount 来查看)。

inetd Internet services服务进程，当系统启动时，rc.local会启动inetd读取inetd.conf这个配置文件，读取网络上所有服务器的地址，联接启动 inetd.conf中所有的服务器。 当client请求服务时， inetd 就会为它们启动相关的服务进程，如user使用telnet时 ，inetd启动telnetd 迎合user telnet的需求， 其余像ftp、finger、rlogin之类的应用程序， inetd也都会启动相对应的服务程序ftpd、fingerd、rloingd等。

portmap portmap是个服务程序，主要功能是将 TCP/IP 通信协议的端口数字转换成 RPC 程序数字，因为这样客户端才能进行 RPC 调用。 一般 RPC服务器是被inetd启动的， 所以portmap必须在inetd之前启动，否则无法进行RPC调用。

5.2 建立PPP连接和配置PPP服务器

下面将介绍在 Linux下如何接到一台 PPP服务器，如何使用 PPP进行局域网之间的互联，并且介绍一种把Linux电脑设定为PPP服务器的方法。

5.2.1 有关PPP的基础知识

1. 了解PPP

PPP(Point to Point Protrol，点对点协议)将双方的计算机或单一出口的计算机网络抽象成为对等的点，在两点之间通过串口连接运行网络协议进行通信。这种连接可以是直接的电缆连接或者通过调制解调器以及电话线的连接。一般说来， PPP将创建的连接类似以太网，通过

它，可以连到一台PPP服务器上并使用该服务器所连接的网络资源，就如同你是直接连接在该局域网上；也可以把你的计算机设为一台 PPP服务器，这样一来其他电脑就可以拨入你的电脑并且存取在你局域网里的资源；也可以使用 PPP把网络连接在一起，这时，运行 PPP的两台计算机就是两个网络相互间的出入口。

PPP和以太网的差异，一方面在于速度，标准的以太网最大通信速度是 10Mbps(每秒1兆比特)，但是，现有最快的调制解调器只有 56Kbps(每秒56千比特)。习惯于局域网的用户可能会对这个速度感到无法忍受。另一方面，以太网是广播 (Broadcast)网，可以同时接入多个设备，而PPP则只能是两个“点”之间的连接(当然，这两个“点”可以是局域网)。

2. 客户端及服务端

作为一个严格的端对端协议， PPP连接一旦建立，两个抽象的“点”之间在技术上就是绝对相等的。但是，为了说明双方的拨入关系，我们不妨借用“服务器 (Server)”和“客户机(Client)”的说法。下面，将主动拨入(就是发出呼叫的一方)的一方称为客户端，而将被拨入的一方(也就是接受并处理拨入的一方)称为服务器。在Linux下，任何计算机都可以是 PPP服务器或者客户端(因为两者本来就是对等的)， 甚至可以同时兼做两者，也就是同时建立多个PPP连接。

5.2.2 使用PPP拨号上网

在过去的书籍中有很多对如何使用 PPP拨号上网的介绍，而且常常十分冗长，让读者望而却步，其实，这些繁琐的步骤已经落后于 Linux的发展了，对于高版本的Linux而言，使用PPP拨号上网已经比过去容易了很多。

1. Linux内核(kernel)的支持

现在用的Linux内核都是2.0.x以上的版本了，在默认安装的情况下已经自动包含了 PPP支持，在启动Linux的时候可以看到这样的内容：

```
PPP Dynamic channel allocation code copyright 1995 Caldera, Inc.
PPP line discipline registered.
```

如果没有看到这样的内容，就需要重新配置和编译内核，当提示是否安装系统核心的 PPP支持的时候，选择“是”就可以了。

2. 安装PPP程序(pppd)

系统中不一定包含pppd，如果没有的话，就需要自己下载并安装：

如果你的 Linux 软件包并未包含 PPP 软件， 可从站点sunsite.unc.edu的目录/pub/Linux/system/network/serial/ppp下，或从站点cs.anu.edu.au的目录/pub/software/ppp下下载最新的PPP客户软件包，例如ppp-2.3.5.tar.gz。

用下面的命令解压缩：

```
# tar zxvf ppp-2.3.5.tar.gz
# cd ppp-2.3.5
```

在子目录Linux的两个文件if_ppp.h和ppp_defs.h的开头加入一行：

```
#include<asm/types.h>
```

用以下命令编译和安装：

```
#./cofigure
```

```
Creating links to Makefiles.
Makefile -> linux/Makefile.top
pppd/Makefile -> Makefile.linux
pppstats/Makefile -> Makefile.linux
chat/Makefile -> Makefile.linux
```

这个命令通过检查操作系统，建立四个符号连接。

```
#make
#make install
```

上面两个命令将完成编译和安装。

3. 调制解调器设置与PPP连接测试

下一步需要设置调制解调器的参数，最简单的办法是启动终端连接工具 minicom，按<Alt>+O进入设置菜单，根据你的调制解调器进行设置，主要参数有连接速率、初始化参数等等，一般情况下不需要做什么改动。最后选择“ Save as default ”，把设置保存为缺省配置。

现在可以测试一下PPP连接了，使用minicom建立一个连接，拨号到你的ISP，比如169。注意，拨号前先要配置好用户名和口令： Linux机器在每次拨入的时候要能够辨认的两个主要的提示(服务器发给你的提示信息)是：

要求输入使用者名称的提示(通常是login:)；

要求输入密码的提示(通常是password:)；

一定要设置好你的Linux终端对这两个提示信息的回应。另外，如果必须发出某个指令来启动服务器端的 PPP，那么也需要找出一旦拨入之后服务器给你的输入命令提示和你应该输入的命令。

如果服务器自动启动PPP的话，一旦你拨入完成，将开始见到屏幕上出现“ 垃圾 ”——这是PPP 服务器端传送给你的机器以启动并且配置 PPP 连线的资料。看起来应该是像这样的东西：

```
~y}#.!}!}!} }8}!}$}%U}"}&} } } } }%}& ...}'}"}(}"} .~~y}
```

现在调制解调器已经设置完成，可以继续下面的工作了。

4. 建立PPP连接

在X Windows下建立连接是一件比较简单的事，只需按下列步骤添加并且配置好 PPP设备：

1) 进入X Windows，在Control Panel 里面选择Network Configuration。

2) 在Interface里选择添加PPP设备。

3) 输入ISP电话(如163/169)、用户名、密码。

4) 在DNS里输入ISP提供的DNS，比如163提供的：202.96.0.133。

5) 在Networking中将本地IP和服务器IP都设置为0.0.0.0。

6) 修改/etc/ppp/options，加上defaultroute(在设备表中将此连接设定为默认连接设备)。

7) 如果连接很慢或者经常掉线的话，延长 NO CONNECTION的时间，比如60。

8) 最好在Hardware中去掉Abort connection of well know errors，在 Communication中只保留你的用户名，密码。

现在可以使用Netscape Communicator通过刚刚建立的PPP设备连上Internet了。

5.2.3 配置PPP 服务器

PPP是严格的点对点协议，就像上面所说，“ 客户端 ”和“ 服务器 ”除了在连接建立的过

程中有“主叫”和“被叫”的区别外，其余都是完全对等的，因此，以上面介绍的客户端为基础，建立PPP服务器最重要的就是处理“等待用户拨入”和“建立连接”。下面介绍一种实现PPP服务器的方法。

1. 系统的支持

所谓系统的支持就是Linux核心支持和pppd。

在开始下一步之前先测试一下系统：在提示符下键入“pppd”，不加任何参数，如果能够看到：

```
~y}#.!}!!} }8}}$}%U}"}&} } } } }%}& ...}'}"}(}"} .~~y}y!!)
```

这样的乱码，说明系统支持已经具备，可以进入下一阶段配置了。

2. 编辑配置文件

如果已经按照5.2.2正确建立了PPP连接，下面的文件已经存在，只需加以修改。如果不需要或者还没有建立PPP客户端连接，就自己创建这个文件：

```
etc/ppp/options文件内容
# 自动获取IP地址；
0.0.0.0:
# 不使用任何“溢出”控制序列；
asyncmap 0
# 子网掩码；
netmask 255.255.255.0
# 使用 uucp 格式的锁定文件以避免他人使用串口；
lock
# 使用硬件流量控制；
crtscts
# 使用调制解调器控制线；
modem
# 使用代理地址解释协议来传送数据包；
proxyarp
# 最大传送包大小为 552 bytes;
mtu 552
# 最大接收包大小为 552 bytes;
mru 1500
```

子网掩码和最后两项在不同的机器设置不同，可以在命令行键入：

```
ifconfig
```

来查看具体的数值。

3. 创建用户和编辑用户shell

用adduser命令建立用户帐号，注意，用户的 shell设置为 ppplogin。也可以对不同的用户建立不同的shell，如用户ppp1对应ppp1login。

然后要给每个用户设立一个自动登陆的 shell文件：

```
exec /usr/sbin/pppd passive /dev/ttyS5 552 netmask 255.255.255.0
proxyarp:10.0.0.1
```

参数10.0.0.1是用户用PPP登陆后自动获取的IP，这里是局域网内部的一个例子，使用中可以设置自己的值。

还要记得将ppplogin文件设为可执行的。 然后就可以使用另一台计算机尝试通过调制解调器和电话线连接这台PPP服务器了。

5.3 配置DNS服务器

5.3.1 DNS (Domain Name System)简介

在TCP/IP中用IP地址标识系统，这种标识方法是很不方便的。IP地址是16字节的二进制数，它以16x进制数表示成32个字符。如果用户必须使用4E68:9f24:FFFF:1734:1529:0800:0219:7E14这个冗长的地址，不仅效率极低，也容易出错。

幸好，在TCP/IP中提供了这一问题的解决方案，即采用域名系统 (DNS:Domain Name System)。用户利用此业务，能够以简单的域名标识系统，以便于实际应用。在协议执行时，使用的不是域名而是IP地址。为将域名“翻译”成IP地址或进行相反的操作，可以利用DNS。DNS不仅提供这种变换的服务，还对域名进行层次化的处理。实际上，DNS比其他任何TCP/IP协议都更强调利用层次的概念。

5.3.2 域名服务系统

域名系统是一个分布式的主机信息数据库，采用客户机/服务机制，它将整个网络按照组织结构或管理范围，逻辑地分解为一个层次的结构，形成一个倒立的树形域名系统结构 (类似Unix文件系统的结构)，其顶部是根，根名是空标记""，根以下的每个结点都有惟一的标记名，称为域(Domain) 列，并用“.”进行分隔的标记名串，如:nd.ndb.fib.cn。在域名系统中每个域可由不同的组织管理，每个组织可以将它的域再分成一系列的子域，并将其交由其他组织管理。

整个域名空间按其管理范围，被分为区 (Zone)，每个区从一个域开始，向下延伸到叶子域或别的区的开始点。

5.3.3 域名服务器和解析器

域名系统是采用客户机/服务机制。解析器(Resolver)是客户方，它负责查询域名服务器，解释域名服务器的应答，并将查询到的有关信息返回请求的程序或用户。

存储域名空间信息的程序(主机)就是域名服务器。一个域名服务器可以管理一个区，也可以管理多个区。域名服务器有以下几种类型。

1) 主域名服务器。一个区由主域名服务器管理，它维护这个区的所有域名空间信息。一个区至少要有两个主域名服务器：第一主域名服务器和第二主域名服务器，它们位于不同的主机上。当第一主域名服务器关闭、坏了或过载时，第二主域名服务器能作为备份服务器工作。第一主域名服务器与第二主域名服务器的区别在于：第一主域名服务器从其硬盘上装入其管理区内域名数据，并对本区内其他域名服务器授权；第二主域名服务器从第一主域名服务器获得授权，从第一主域名服务器取得其管理区内域名数据，并周期性地与第一主域名服务器上的数据比较，更新其管理区内域名数据。

2) Cache-Only服务器。所有的域名服务器都是Cache-Only服务器。这就是说，域名服务器把从别的域名服务器所接收到的信息都保存在Cache中，直到信息失效。一个Cache-Only服务器对任何区都不负责任，它将收到的查询请求转发给其他域名服务器，直到获得所需信息。

3) 转发服务器。转发服务器(Forward Server)接到地址映射查询请求时，在其Cache中查

找，如果找不到，就把请求依次转发到指定的域名服务器，直到查询到结果为止，否则返回无法映射。

在Linux上的名字服务是由named 程序来执行的。这是属于bind套件的一部份，这个套件是由Paul Vixie为网际网络软件集团(Internet Software Consortium)所协调开发的。红旗Linux发行套件中就包含named，它缺省安装于/usr/sbin/named。既然我们的红旗Linux系统已经有了named这个使用工具程序，那么我们就可以直接使用它来做构造我们的名字服务器了。

DNS是个以整个网络为范围的(net-wide)资料库，所以我们要小心地将我们的资料放进去，我们还应该学习去使用它、管理它，追查它的错误，那么我们将会成为另一个保持网络免于因管理不善而效率低的好管理者。

5.3.4 配置暂存专用名字服务器

暂存专用名字服务器是DNS配置的一种最简单好用的方式，对拨号网络使用者非常有用。一台暂存专用名字服务器将会为名字查询找出答案并且在下一次你需要那个名字的时候记得答案。

1. 建立/etc/named.boot的文件

首先我们需要建立一个称为/etc/named.boot的文件，当named启动时会读取这个文件。目前它应该单纯地包含以下内容：

```
;  Boot file for nicolais 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
```

注意 在这文件的某些版本中，这个文件的这份列表会在第一个非空白字符前包含一些空格或Tab键。这些不应该出现在文件中。如果发现这份文件包含上面描述的空格或者Tab键，一定要记得删除任何前面的空白。

“directory”这一行告诉named到哪里去找寻文件。所有其后命名的文件都将是相对于此目录的。根据Linux文件系统标准正确的目录应该是在/var/named。因此pz是位于/var/named之下的，也就是，/var/named/pz。

称为/var/named/root.cache的这个文件是在此命名的。这个 /var/named/root.cache 应该包含如下的内容：

```
.      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
```

注意 一定要去掉每行前面的空白字符！

在named.boot里的下一行是primary这一行。本书将会在稍后的章节里解释它的用法，目前只要把它设为在 pz 子目录下一个称为 127.0.0 的文件：

```
@       IN    SOA      linux.bogus. hostmaster.linux.bogus. (
                       1       ; Serial
                       28800   ; Refresh
                       7200    ; Retry
                       604800  ; Expire
                       86400)  ; Minimum TTL
              NS       ns.linux.bogus.
1             PTR      localhost.
```

2. 关于/etc/resolv.conf文件

接下来，我们还需要一份看起来像这样的 /etc/resolv.conf文件：

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

“ search ”这一行指出对于任何你想连往的主机名字应该搜寻的领域。“ nameserver ”这一行指出你的机器可以在哪个位址上找到一台名字服务器，在这个例子中是你自己的这台机器，因为你在它上面执行 named。如果想列出好几个名字服务器把它们都放在一行 “ nameserver ”里，用空格隔开。(注意：named 从不读取这个文件，而是使用 named 的名字解答器会读取。)

下面来说明这个文件有什么作用：如果某个客户端尝试要找寻 foo的话，那么首先尝试的是 foo.subdomain.your-domain.edu 这个名字，然后接下来是foo.your-fomain.edu这个名字，最后则是 foo 这个名字。如果有某个客户端尝试要找寻 sunsite.unc.edu 的话，那首先尝试的是sunsite.unc.edu.subdomain.your-domain.edu这个名字，然后接下来是 sunsite.unc.edu.your-domain.edu 这个名字，最后则会是 sunsite.unc.edu 这个名字。我们最好别放太多领域到 search 这行里去，搜寻它们会多花时间。

这个例子假设我们属于 subdomain.your-domain.edu 这个领域，那么我们的机器，可能会称为your-machine.subdomain.your-domain.edu。在 search 这行里不应该包含你的 TLD(顶层领域 Top Level Domain，在这个例子中是edu 这个领域)。如果我们经常需要连线到在另外一个领域里的主机，那么我们可以把该领域像这样地加进 search 这行里：

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

依此类推。很明显的是你得放入真实的领域名字来取代这些名字。请注意，在领域名字的最后面并没有句号“ . ”。

3. 修正/etc/nsswitch.conf和/etc/host.conf

接下来，应该根据我们libc版本的不同，来修正/etc/nsswitch.conf或者是 /etc/host.conf 文件。

```
/etc/nsswitch.conf
```

这是一个很长的文件，它指出到何处去取得各种不同的资料型态，从什么文件或是资料库取得。它的顶端经常会包含一些有用的注解。找出以“ hosts: ”作为开头的那一行，它应该是这样：

```
hosts:  files dns
```

如果文件里没有以“ hosts: ”作为开头的行，那么把上面一行加上去。它是说程序应该先在/etc/hosts 文件里找寻，然后根据 resolv.conf 询问 DNS。

```
/etc/host.conf
```

它可能包含数行，其中应该有一行以 order开始，而且它看起来会像这样：

```
order hosts,bind
```

如果文件里没有“ order ”这一行的话，那么你应该添加上去。它告诉主机名字解析函数先在/etc/hosts里找寻，然后查问名字服务器(在 resolv.conf 里会在 127.0.0.1 这个地方)。

4. 运行named

这些全部完成后就可以开始运行 named了。如果使用拨号网络连线的话，请先连上网络。键入“ ndc start ”并且按下回车键，没有选项。如果它不行的话，试着使用 “ /usr/sbin/ndc start ”来取代。当在运行named的时候如果观察一下(使用 tail -f /var/adm/messages 指令)系统记录信息文件(通常是称为/var/adm/messages的文件，但也可能在 /var/log 下，或是叫 syslog 的文件)，那么应该会看见像这样的一些东西：

```
Jun 30 21:50:55roke named[2258]starting. named 4.9.4-RELSun Jun 30
21:29:03
METDST 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)
```

如果有任何关于错误的信息，那么就是有个错误发生， named将会指明有错误的文件(可能是named.boot或root.cache 。杀掉named程序并回头检查那些文件。

现在可以用 nslookup 来检查一下：

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

如果这是你所得到的回应，那么它已经能够运作。我们希望是这样。得到任何其他回应都请回头检查每一件事。每一次你改变 named.boot 文件之后都得使用 ndc restart 这个指令重新运行 named 程序。 现在你可以输入查询，尝试找寻某些靠近你的机器。

```
> pat.uio.no
Server:  localhost
Address:  127.0.0.1
Name:     pat.uio.no
Address:  129.240.2.50
```

现在 nslookup 要求你的 named找寻pat.uio.no这台机器。然后它 (named)联系在你 root.cache 文件里所指明的名字服务机器中的一台，并且从那里查问它该如何继续下去。在取

得结果之前可能得花费一点时间，因为它搜寻你在 /etc/resolv.conf 里指明的所有领域。

如果你再试一次的话，将会得到：

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

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

注意这回我们所得到的“ Non-authoritative answer: ”这一行。这代表 named 此次并未到网络外去查问，取而代之的是在它的暂存区里找寻并且在那里找到答案。但是暂存的信息可能会过时，所以它用“ Non-authorative answer: ”来通知你，有这个 (很轻微的)危险性存在。当 nslookup 说这是你第二次查问某台主机时，这是 named 能暂存该项信息并且正常运作的一个信息。你可以使用“ exit ”指令离开nslookup程序。

现在我们已经成功地设立了一个能够暂存的 named系统。

5.3.5 建立一个简单的领域名字服务器

1. 一点基本的理论

在我们真的开始进行这一节以前，我会提供你一些关于 DNS如何运作的理论。如果你不想要仔细地研究它，至少也得很快地略读一下。当你看到应该放进 named.boot文件里去的内容时，再停止这种略读方式。

DNS是一个阶层式的系统。其顶端写作“ .”，其发音为“ 根(root) ”。在根之下有几个顶层领域(TLD)，最知名的是ORG、COM、EDU及NET，还有更多。在寻找一台机器名字时查询会以递回方法从顶端开始。当你想要找出 prep.ai.ustc.edu 的位置时，你的名字服务器必须找到负责edu的一台名字服务器。这个问题，它会去查问 root.cache 文件，而 . 服务器会给它一份 edu 服务器列表。请看下面的例子：

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

开始查问某台根服务器。

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

设置查询型态为 NS (名字服务器记录 name server records)。

```
> set q=ns
```

查问关于 edu. 的资料。

```
> edu.
```

结尾的 . 在这里非常重要，它告诉该服务器，我们所查问的 edu 是在 . 之下的那一个，这稍能缩小搜寻的范围。

```
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
```

这告诉我们 *.root-servers.net 服务 edu. 领域，所以我们可以藉此继续查问 c 服务器。现在我们想要知道是谁负责下一层 ustc.edu. 的领域名字：

```
> ustc.edu.
Server:  c.root-servers.net
Address:  192.33.4.12
Non-authoritative answer:
ustc.edu nameserver = STRAWB.ustc.edu
ustc.edu nameserver = W20NS.ustc.edu
ustc.edu nameserver = BITSY.ustc.edu
Authoritative answers can be found from:
STRAWB.ustc.edu  internet address = 18.71.0.151
W20NS.ustc.edu   internet address = 18.70.0.160
BITSY.ustc.edu   internet address = 18.72.0.3
steawb, w20ns以及 bitsy 负责 mit 领域，选择其中一个并且查询 ai.ustc.edu:
> server W20NS.ustc.edu.
```

主机名字不分大小写，但是我使用鼠标来剪贴，所以这些资料是荧屏的拷贝。

```
Server:  W20NS.ustc.edu
Address:  18.70.0.160
> ai.ustc.edu.
Server:  W20NS.ustc.edu
Address:  18.70.0.160
Non-authoritative answer:
ai.ustc.edu  nameserver = WHEATIES.AI.USTC.EDU
ai.ustc.edu  nameserver = ALPHA-BITS.AI.USTC.EDU
ai.ustc.edu  nameserver = GRAPE-NUTS.AI.USTC.EDU
ai.ustc.edu  nameserver = TRIX.AI.USTC.EDU
ai.ustc.edu  nameserver = MUESLI.AI.USTC.EDU
Authoritative answers can be found from:
AI.USTC.EDU  nameserver = WHEATIES.AI.USTC.EDU
AI.USTC.EDU  nameserver = ALPHA-BITS.AI.USTC.EDU
AI.USTC.EDU  nameserver = GRAPE-NUTS.AI.USTC.EDU
AI.USTC.EDU  nameserver = TRIX.AI.USTC.EDU
AI.USTC.EDU  nameserver = MUESLI.AI.USTC.EDU
```

```
WHEATIES.AI.USTC.EDU internet address = 128.52.32.13
WHEATIES.AI.USTC.EDU internet address = 128.52.35.13
ALPHA-BITS.AI.USTC.EDU   internet address = 128.52.32.5
ALPHA-BITS.AI.USTC.EDU   internet address = 128.52.37.5
GRAPE-NUTS.AI.USTC.EDU   internet address = 128.52.32.4
GRAPE-NUTS.AI.USTC.EDU   internet address = 128.52.36.4
TRIX.AI.USTC.EDU internet address = 128.52.32.6
TRIX.AI.USTC.EDU internet address = 128.52.38.6
MUESLI.AI.USTC.EDU   internet address = 128.52.32.7
MUESLI.AI.USTC.EDU   internet address = 128.52.39.7
```

所以 weaties.ai.ustc.edu 是 ai.ustc.edu 的一台名字服务器：

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

现在我改变查询的型态，我们已经找到该名字服务器，现在我们将要询问 wheaties，它所知道的关于 prep.ai.ustc.edu 任何事情。

```
> set q=any
> prep.ai.ustc.edu.
Server:  WHEATIES.AI.USTC.EDU
Addresses:  128.52.32.13, 128.52.35.13
prep.ai.ustc.edu CPU = dec/decstation-5000.25    OS = unix
prep.ai.ustc.edu
inet address = 18.159.0.42, protocol = tcp
#21 #23 #25 #79
prep.ai.ustc.edu preference = 1, mail exchanger = life.ai.ustc.edu
prep.ai.ustc.edu internet address = 18.159.0.42
ai.ustc.edu  nameserver = alpha-bits.ai.ustc.edu
ai.ustc.edu  nameserver = wheaties.ai.ustc.edu
ai.ustc.edu  nameserver = grape-nuts.ai.ustc.edu
ai.ustc.edu  nameserver = mini-wheats.ai.ustc.edu
ai.ustc.edu  nameserver = trix.ai.ustc.edu
ai.ustc.edu  nameserver = muesli.ai.ustc.edu
ai.ustc.edu  nameserver = count-chocula.ai.ustc.edu
ai.ustc.edu  nameserver = life.ai.ustc.edu
ai.ustc.edu  nameserver = mintaka.lcs.ustc.edu
life.ai.ustc.edu internet address = 128.52.32.80
alpha-bits.ai.ustc.edu   internet address = 128.52.32.5
wheaties.ai.ustc.edu internet address = 128.52.35.13
wheaties.ai.ustc.edu internet address = 128.52.32.13
grape-nuts.ai.ustc.edu   internet address = 128.52.36.4
grape-nuts.ai.ustc.edu   internet address = 128.52.32.4
mini-wheats.ai.ustc.edu  internet address = 128.52.32.11
mini-wheats.ai.ustc.edu  internet address = 128.52.54.11
mintaka.lcs.ustc.edu internet address = 18.26.0.36
```

所以我们从．开始连续找出在领域名字里的下一层名字服务器。如果你使用自己的 DNS 服务器而不是所有这些个其他的服务器，你的 named 当然会暂存所有这些在为你寻找这个答案时所找到的信息，而且在一段时间内它不必再次查问。

一个很少被论及但同样重要的是 in-addr.arpa 领域。它也像“正常的”领域一样是巢状的。

in-addr.arpa让我们可以在拥有主机位址的时候得知该主机的名字。这里有件重要的事要注意：在 in-addr.arpa 这个领域中 ip 数字是以反向顺序书写的。如果你有某台机器的位址 ：192.128.52.43，那么 named 会以类似 prep.ai.ustc.edu 这个例子的方式来处理：找出 arpa. 的服务器，找出 in-addr.arpa. 的服务器，然后再找出 192.in-addr.arpa.的服务器，找出 128.192.in-addr.arpa. 的服务器，接着找出 52.128.192.in-addr.arpa. 的服务器，最后再找出所需的43.52.128.192.in-addr.arpa. 的记录。

2. 建立我们自己的领域

现在来定义我们自己的领域。我们将会创造出 linux.bogus这个领域并且定义其中的机器。在本章中，我们使用一个完全是虚拟出来的领域名字，以便确定我们不会扰乱到网络上的其他地方。我们早就以 named.boot 里的这一行开始了这个部份的设置：

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

请注意，在这个文件里的领域名字结尾并没有加上“ .”符号。第一行把定义 0.0.127.in-addr.arpa 的文件命名为 pz/127.0.0。我们早已建立了这个文件，它是这样的：

```
@                IN    SOA     linux.bogus. hostmaster.linux.bogus. (
                       1       ; Serial
                       28800   ; Refresh
                       7200    ; Retry
                       604800  ; Expire
                       86400)  ; Minimum TTL
                 NS    ns.linux.bogus.
1                PTR   localhost.
```

请注意，在这个文件里所有的完整领域名字结尾的 “ .”符号，这与上面提到的 named.boot 文件形成对比。有些人喜欢以 $ORIGIN 指令起始每个区域文件，但这是不必要的。一个区域文件的基点(就是其所属的 DNS 阶层架构位置)是在 named.boot 文件的“ 领域 ”行里指定的，在这个例子里是 0.0.127.in-addr.arpa。

这个“ 区域文件 ”中包含三种“ 资源记录 ” (resource records, RRs)：一个是 SOA 资源记录，一个是 NS 资源记录和一个 PTR 记录。 SOA 是授权运行(Start Of Authority)的缩写。“ @ ”是个意思为基点的特殊标记，而因为这个文件的“ 领域 ”行说 0.0.127.in-addr.arpa，所以第一行实际上是指：

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

NS 是名字服务器资源记录，它告诉 DNS 什么机器是这个领域 0.0.127.in-addr.arpa 的名字服务器，也就是 ns.linux.bogus. 而最后的 PTR 记录说 1 (等于是 1.0.0.127.IN-ADDR.ARPA, 也就是 127.0.0.1)的名字是 localhost。

SOA 这个记录是所有区域文件的序文，而且在每一个区域文件里都应该有这惟一的一个，最开头的记录。它描述该区域，它从何而来 (一台称为 linux.bogus 的机器)，谁负责其内容(hostmaster@linux.bogus)，这个区域文件是什么版本 (serial: 1)，其他必须做的，以及那些有关暂存与次要名字服务器的任务。剩下的栏位如 refresh, retry, expire 以及 minimum ，你可以使用这份文件里所用的数字，而且不会出问题。

现在重新运行 named(使用 ndc restart 指令)并使用 nslookup 来检验我们做了什么：

```
$ 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
```

所以它管理从 127.0.0.1 得到 localhost 的过程，很好。现在开始我们的主要任务， linux.bogus 这个领域，在 named.boot 里 插入新的一行 primary 指令:

```
primary          linux.bogus              pz/linux.bogus
```

注意 在 named.boot 文件里领域名字的结尾还是没有“ . ”符号。

在这个linux.bogus区域文件里我们将放入一些完全虚拟的资料 :

```
;
; Zone file for linux.bogus
;
; Mandatory minimum for a working domain
;
@         IN    SOA       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
```

关于 SOA 记录有两件事需要注意。首先， ns.linux.bogus 必须 是一台具有 A 记录的真正机器。在 SOA 记录中用 CNAME 记录为名字的机器是不合法的。它的名字不一定要是 ns，它可以是任何合法的主机名字。其次， hostmaster.linux.bogus 应该视为 hostmaster@linux.bogus，这应该是一个邮件位址或别名，是维护这个 DNS 的人经常读信的位址。任何关于此领域的信件会被送到这个位址。它的名字不一定要是 hostmaster，它可以是任何合法的电子邮件位址，然而这个位址“ hostmaster ”应该能工作正常。 在这个文件里有一种新的资源记录型态，即 MX 型态，或是邮件交换者资源记录 (Mail eXchanger RR)。这种资源记录型态告诉邮递系统地址 someone@linux.bogus 的邮件要寄送到哪里，也就是应该寄送到 mail.linux.bogus 或是 mail.friend.bogus。在每个机器名字前面的数字是 MX 资源记录的优先权，数字比较低 (10) 的资源记录是邮件主要应该寄往的机器。如果失败，可以把它寄往数字比较高的机器，一台次要的邮件处理者，也就是在这里具有优先权 20 的 mail.friend.bogus。

用 ndc restart 重新运行 named。以 nslookup 检验结果：

```
$ 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
```

小心地检验，你将会发现一个错误。这一行

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

全都错了。它应该是

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

我故意犯了个错误，所以读者可以借此来学习。仔细看看该区域文件我们会发现这一行

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

遗漏了一个句点。或是说多了个“ linux.bogus ”。在区域文件里如果一个机器名字不是以句点结尾，那么会在其结尾加入基点。所以不论是

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

或是

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

都是正确的。在一个区域文件里领域名字应该要么写出来并以“ .”结尾，要么一点都不要包含进去，而在这种情况下其领域预设为基点。其他人也许喜欢采用另一种方式。我必须强调的是，在 named.boot 文件里领域名字后面不应该有“ .” 的存在。所以在加入我强调的重点后得到一份新的区域文件，其中还包含一些额外的信息：

```
;
; Zone file for linux.bogus
;
; Mandatory minimum for a working domain
;
@    IN    SOA    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      20mail.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"
```

你也许想要移动前面三个 A 记录以便让它们靠近其他的相关记录，而不是像这样放在最前端。这里有几个新的资源记录：主机信息 (HINFO: Host INFOrmation)包括两个部份，使用引号涵括每个部份是个好习惯。第一部份是机器上的硬件或是中央处理单元，而第二部份是机器上的软件或是作业系统。ns 有一颗 Pentium CPU 并且执行 Linux 1.2 系统。 TXT 记录是个随意的文字记录，可以用它来记录任何事情。正式名字 (CNAME: Canonical NAME)可以用来给每台机器数个名字。所以 richard 以及 www 都是 ns 的一个别名。很重要的一点是 A, MX, CNAME, 以及 SOA 记录永远不该参照 CNAME 记录设置的别名，它们只应该参照 A 记录所设置的名字，所以像这样的记录是错的：

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

不过下面这样是对的：

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

还有一点也很重要的是注意正式名字所设置的对电子邮递位址而言不是合法主机名字 ：webmaster@www.linux.bogus 以上面的设置而言是一个不合法的电子邮递位址。即使它在你的系统上可以运作，可以预料的是，很少有电子邮件管理者会去实行这项规则。避免这个问题的方法是使用 A 记录(或者也可能是一些其他的，像是 MX 记录)来取代：

```
www A   127.0.0.2
```

执行 ndc reload 载入新的资料库，这会使 named 再一次读取其文件。运行 nslookup来检查系统。

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

> ls -d linux.bogus
```

这意指应该列出所有的记录

```
[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  PentiumLinux 1.2
 ns                        TXT      "RMS"
 linux.bogus.              SOA      ns.linux.bogus hostmaster.linux.bogus.
(199511301 28800 7200 604800 86400)
```

让我们检查它对于单独的 www 会说什么：

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

www.linux.bogus canonical name = ns.linux.bogus
```

...换句话说，www.linux.bogus 真正的名字是 ns.linux.bogus。

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

而 ns.linux.bogus 拥有 127.0.0.2 这个位址。看起来也很好。

5.3.6 配置实例

这里我们列出一些真正的区域文件，这是一个可以运作的真实例子。

声明 千万不要把它放到你的名字服务器上！把它当作参考资料来阅读好了。如果你想做实验，请用上面虚拟的例子。

1. /etc/named.boot 或/var/named/named.boot

我们发现需要两行 primary 的反向区域：一个是 127.0.0，另一个是 LAND-5 的子网络 206.6.177。还有一行给 land-5.com 的正向区域。

```
;  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
```

2. /var/named/root.cache

记住这个文件是动态的，这里列出来的是旧的。最好自己用 dig 产生一个新的。

```
; <<>> 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
```

3. /var/named/zone/127.0.0

这是基本的、不能省略的 SOA 记录，以及将 127.0.0.1 指向 localhost 的记录。两者都是必需的。不应有其他的东西在此文件中。这个文件可能永远不必更新，除非你的名字服务器或管理人地址变更。

```
@            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.
```

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

在这里我们看到这个必需的 SOA记录，必要的 NS 记录。可以看到它有一个位于ns2.psi.net的次要名字服务器。这样做也是必要的，总是确保有一个次要的服务器做备份。我们还看到，做为 LAND-5 负责的所有不同服务的主机，它通过许多的 CNAME 记录做到这点(另一个做法是用 A 记录)。

就像从 SOA 记录看到的一样，区域文件以 land-5.com 开始，管理人是 root@land-5.com。序号(serial)以 yyyymmdd 的格式加上当天的号码。它可能是 1996 年9月20日那天的第六个版本的区域文件。请记住序号必须单调地递增，这里它只用一位数字表示当天的号码，因此在编辑九次之后他必须等到第二天才能再编辑这个文件。

```
@            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
```

另外值得注意的是，所有的工作站都没有个别的名字，而是用一开头加上最后两位的 IP 数字。用这样的惯例可以大大地简化维护工作，但可能有点不方便，而且，事实上可能是你的客户不满的来源。

5. /var/named/zone/206.6.177

在后面说明这个文件。

```
@               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.
```

反向区域似乎是整个设置中最让人头痛的部份。它是在你有机器的 IP 数字时查询机器名字用的。例如：你有一部 irc 服务器接受 irc 客户端的连线，但你的是挪威的服务器，因此你只希望接受来自挪威及其他斯堪地那维亚国家的连线。当接到来自客户端的连线时， C 程序库能够告诉你连线机器的 IP 位址，因为客户端的 IP 数字是包含在传入的网络封包中的。然后你可以呼叫一函数 gethostbyaddr，以给定的 IP 查询机器名字。 gethostbyaddr 函数会向 DNS 询问，DNS 就会出去查询这台机器名字。假设客户端来自 ws_177200.land-5.com，C 程序库提供给 irc 服务器的 IP 是 206.6.177.200。为找出机器的名字，我们要查询 200.177.6.206.in-addr.arpa。DNS 服务器首先找到负责 arpa. 的服务器。然后找到 in-addr.arpa. 的服务器，接着是反过来的 206，随后是 6，最后找到在 LAND-5 负责 177.6.206.in-addr.arpa 区域的服务器。由此我们得到 200.177.6.206.in-addr.arpa 的答案是“ PTR ws_177200.land-5.com ”记录，意指拥有 IP 206.6.177.200 的名字是 ws_177200.land-5.com。不过就像在解释 prep.ai.ustc.edu 如何查询时所说的，这个过程有点不对。

我们回到 irc 服务器的例子。 irc 服务器只接来自斯堪地那维亚国家的连线，也就是，*.no, *.se, *.dk。ws_177200.land-5.com 这个名字显然并不符合。因此服务器会拒绝连线。如果没有经由 in-addr.arpa 区域到 206.2.177.200 的反向对应，服务器将无法查到机器名字而将会拿 206.2.177.200 来和 *.no, *.se 及 *.dk 比较，当然找不到符合的。

有些人会告诉你，反向对应查询只对服务器重要，或是一点也不重要。当然不是：许多ftp、news、irc 甚至http服务器将不接受无法查到名字的机器连线。因此事实上机器的反向名字对应是必需的。

5.3.7 维护工作

维持它的运作，除了维持它们的继续执行之外，对于 named，还有个维护的任务，那就是维持 root.cache 文件的更新。最简单的方法是使用 dig 程序，首先不加任何参数执行 dig 程序，你将会取得从自己服务器得到的 root.cache。然后以 dig @rootserver . ns 查问所列出的根服务器其中之一。这份输出看起来非常地像一个 root.cache 文件，除了一堆额外的数字以外。这些数字不会有什么妨碍。把它存放到文件里 (dig @e.root-servers.net . ns >root.cache.new)并且用它来取代原本旧的 root.cache 文件。取代了原先的文件之后要记得重新启动 named 程序。

下面我们提供一个指令稿，它可以自动执行来更新 named.cache，这个指令稿假设你的电子邮件可以运作而且“ hostmaster ”这个邮件别名有定义。应该修订它以便符合的设置。

```
#!/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 . ns >root.cache.new

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

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

5.3.8 拨号网络连线的自动设置

我们以使用 PPP 拨号网络为例，如果你使用slip或是cslip连线方式，那么你的设置里几乎每个地方都可能跟我的不同，不过你可以方便地将它移植到你的特定的系统中去。

一般来说，当我们的机器没有连上网络时，我们的 resolv.conf文件单纯地包含这一行：

```
domain uio.no
```

这确保我们不必等待主机名字解析函数库去尝试联系某台不可能帮助我的名字服务器。但是当我连上线的时候我想要运行我的 named并且拥有一个看起来像前面所描述的 resolv.conf 文件。我们可以保持两份resolv.conf 的“样板”文件，resolv.conf.local 以及resolv.conf.connected 来解决问题。后面这一个看起来像前面所描述过的 resolv.conf 文件。要自动化连线到网络的过程我执行一个称为 `ppp-on' 的指令稿：

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

pppd 有个称为 options 的文件，它告诉 pppd 如何取得连线的一些特殊事项。一旦我们的 ppp 连线完成后， pppd 运行一个称为 ip-up 的指令(这在 pppd 的线上使用手册里有描述)。这里是该指令稿里面的一部份：

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

换句话说，我在这里运行我的 named 程序。当 ppp 离线时， pppd 执行一个称为 ip-down 的指令文件：

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

所以，这在连线时配置并设置相关事宜，而且在离线时解除该配置并结束相关程序。某些程序，例如irc 与talk，为了让其工作正常，还必须修改 hosts 文件。我在 ip-up 上加入：

```
cp /etc/hosts.ppp /etc/hosts
echo $myip    roke >>/etc/hosts
hosts.ppp是简单地包含了
127.0.0.1      localhost
```

而 echo 指令插入了我收到的 IP 号码作为我机器的名字(roke)。你应该改用自己机器的名字，可以用 hostname 指令查出。

当你没有连线到网络上时执行 named 可能并不聪明，这是因为 named 将会尝试送出查询到网络上而且其终止时限 (timeout)很长，而每次有某些个程序尝试解析一个名字的时候，你

就得等待到这个终止时限。如果使用拨号网络的话，你应该在连上网时运行 named 并且在离线时杀掉它。如果要在慢速的连线上使用 forwarders 指令，你的网际网络提供者在 1.2.3.4 及 1.2.3.5 设有 DNS 服务器，那么可以插入这么一行：

```
forwarders 1.2.3.4 1.2.3.5
```

到 named.boot 文件里去。这将会减低源自你主机的 IP 流量，可能稍微提升速度。如果你是依线路的资料量付费的话，这点特别重要。这还有个附加价值，让你脱离作为一个暂存的 named 维护者所应负起的责任：你不需要去更新一个空的 root.cache 文件。