第 二十四 章

進階篇——建立網際網路伺服器

在本書的最後一章，我們將使用UNIX來做出現今最聞名的事，即提供網際網路服務。UNIX系統在提供這些服務時有長時間執行的優勢角色。直到最近，幾乎所有的網際網路伺服器都是UNIX機器。UNIX控制這個領域，雖然它所使用的技術是非專屬於特定廠商和建立在開放標準的基礎上。

我們在本章會介紹形成網際網路骨幹的三個重要的服務—DNS （名稱伺服器），郵件和網頁伺服器。本書對它們其中之一來說都算是入門書籍，因此我們不可能期望包含它們大部分的功能。然而，我們會考量它們重要的子集合(subsets)，亦即剛好足夠，不只能了解它們如何運作，也能設定這些服務「由您自己動手做」。很幸運地，在這個領域的分支是極小的，因此，我們在這裡的討論可適用於任何的UNIX系統。因為所有的網際網路服務在使用Linux時變成免費和自由，所以我們將在我們的網際網路伺服器上使用Linux。

要建立我們的網際網路伺服器且在一種逐漸擴充並連貫的方式下進行，我們將工作在一個真實世界的應用程式包括虛構公司及使用虛構網域名稱。雖然其中的兩個服務（郵件和網頁）不用修改就能直接執行，我們仍需要知道更多關於它們的事。在本章中，有許多大量的詳細資料要討論，但作者試著簡化問題可能的範圍。

目　標

- 了解一個組織對郵件和網頁服務的系統需求。(24.1)
- 學習DNS如何使用名稱伺服器來處理區域的資料。(24.2)
- 設定一個主要名稱伺服器。(24.3)
- 設定解析系統和處理named服務程式。（24.5 和 24.6）
- 學習sendmail如何寄送和接收郵件。(24.7)
- 使用/etc/sendmail.cf來處理一些一般的需求。(24.8)
- 使用aliases來轉送信件。(24.9)
- 分別在直接連接網際網路及使用撥號線路的主機上設定一個郵件伺服器。(24.10)

- 使用fetchmail從一個POP伺服器下載郵件。(24.11.1)
- 了解HTTP如何運作及設定**httpd.conf**來處理一些一般的網頁伺服器功能。（24.13和24.14）
- 設定虛擬主機和了解使用控制目錄存取的技術。 （24.14.6和 24.15）

24.1 在Rotional Planets的網路環境

Rotional Planets公司有一個廣大的網路，它經由一條專線直接地連接到網際網路。公司已經被分配網域名稱*planets.com*，現在打算要在它們自己本身的網路上設定網際網路服務。在這網路中，主機*saturn* 和*uranus*將執行名稱服務，*jupiter*和*saturn* 將提供郵件服務。在它們之間，*saturn*和 *jupiter*在各自的部門中擔任主要的伺服器，*neptune*將提供網頁服務。

其餘的網路被放在一個防火牆**(firewall)**後面，即一部電腦用來保護內部網路避免遭受外部的攻擊。*jupiter*扮演防火牆和提供轉送的能力，它讓一些在內部網路上的機器經由這個通訊閘道器存取網路。有些工作站使用Windows，它們在夜晚會被關機和必須從*jupiter*下載他們的郵件。有一些工作站使用Linux，它們一直在運作不關機而且有郵件從*jupiter*自動轉送給它們。一個使用者從她家中的機器*mercury*使用撥接線路來連接網路。

因為Rational Planets有相當大的建置，它決定要出租一些空間和伺服器。一家小型機構Rational Velvet，剛剛獲得它自己的網域名稱*velvet.com*。因為Velvet沒有直接連線到網際網路，他們的管理部門已決定利用Planets可用的設備。Planets將提供Velvet 20 MB的磁碟空間給他們作為網頁服務之用，和另外的20 MB來接收所有代表他們的信件。*jupiter* 也接受Velvet的郵件，它將有許多FQDN，包括*jupiter.planets.com, mail.planets.com*, *mail.velvet.com*及*velvet.com*。

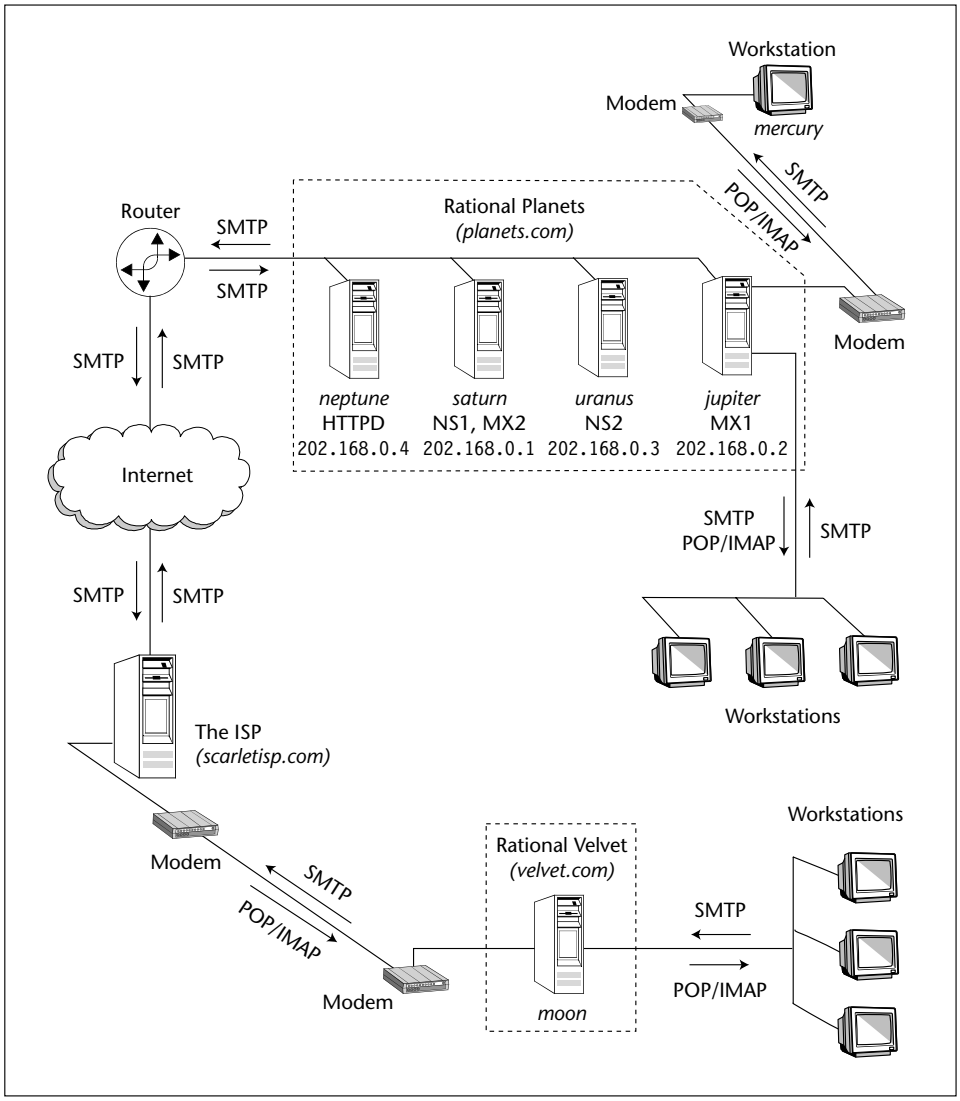
Planets在他們的伺服器上提供無限制數量的郵件信箱給Velvet但不能超過20 MB的限制。一個電子郵件訊息可以被寄給Velvet上的任何使用者並能使用像*user@velvet.com*的地址。同樣的，一個使用者在Planets上也有郵件地址像*user@planets.com*。要能從Planet的站台接收郵件，Velvet已向一個本地ISP註冊而他們有*scarletisp.com*的FQDN。這ISP也提供Velvet一個免費的郵件帳號*velvet@scarletisp.com*。Velvet使用PPP來間歇性的連接到ISP，然後從兩個來源下載郵件，亦即從擁有它們自己網域的Planet伺服器上和從一個ISP給的郵件信箱上。

經過一段時間後，Velvet決定使用Planets為他們標記的20MB空間，來提供他們自己的網頁服務 。主機*neptune*於是有三個FQDN—*neptune.planets.com*, *www.planets.com*和*www.velvet.com*。我們將會實現我們大部分的假設，但是暫時我們已有足夠的材料可以開工了。在您開始動手之前，請注意看一下圖24.1。

24.2　網域名稱伺服器(Domain Name Service-DNS)

網際網路不能使用/etc/hosts來進行主機—位址的解析，它是使用DNS (11.1.3)，並利用一個巨大分散式的資料庫來做這些對應工作。這個系統的起源歸功於Paul Mockapetris，他寫出它的詳細規格和首先實作完成。然而，為柏克萊版本的UNIX，Kevin Dunlap稍後寫出廣為流傳的實作稱為BIND （柏克萊網際網路網域名稱：Berkeley Internet Name Domain)。BIND 8現在已被包裝在幾乎是所有的UNIX機器上。

圖24.1　**在**Rational Planets**的網路環境**



除了維護主機名稱—地址(hostname-address)的資料庫外，BIND也指定伺服器來處理一個網域的郵件。它看起來不可思議，但一個郵件訊息可被寄到*henry@neptune.planets.com* 而不需要阻塞在主機*neptune*中。BIND指定一個或更多的郵件交換器**(mail exchangers)**（接收郵件的主機）並可依喜好設定優先權，因此這些主機會依正確順序被一一試驗。進一步來說，如果BIND 不能夠解析一個FQDN，它必須能交付這問題到一些其他同樣執行名稱服務的主機。

24.2.1 DNS如何運作

DNS將網域名稱領域(namespace)分成許多**zones**，與該區域對應的授權認證。每一個區域的系統管理者有責任維護一個或是更多的名稱伺服器**(name servers)**，即含有該區域名稱—地址訊息的資料庫。主要名稱伺服器含有最近的訊息。次要名稱伺服器可獲得經由區域傳送**(zonal transfer)**而來的主要伺服器的訊息，它也提供當主要伺服器失敗時的一個備援。還有第三種形式的名稱伺服器，但只能快取**(caching-only)**，它僅僅轉移詢問到一個特別群組裡的十三個主要名稱伺服器。在下面幾節中，我們將考慮完成全部三種形式的設定。

一個網域和一個區域間的不同是相當微小的，但還是必須去了解。讓我們進一步考慮網域*birds.edu*分割成兩個次要網域(sub-domain)， *parrots.birds.edu*和*cuckoo.birds.edu*。在*birds.edu*的系統管理者決定要把*cuckoo.birds.edu*的名稱管理授權出去並保持原本*parrots.birds.edu*的名稱管理。這裡有兩個不同的區域，即*cuckoo.birds.edu* 和*birds.edu*。此處，*birds.edu* 區域包含*parrots.birds.edu* 的次要網域，但不包含*cuckoo.birds.edu*。區域一起出現在一個全部網域的架構圖中，跟多個檔案系統結合成單一個檔案系統所用的方法是一樣的。

每一個區域有自己的名稱伺服器群組（主要和次要），並且它們所提供的都是該區域官方的**(authoritative)**（正確的）回答。一個名稱伺服器接受解析系統**(resolver)**的詢問，它代表應用程式動作來獲得一個主機的IP位址，它自己不是單獨的程式，而是一組的程式館常式（library routine）被連結到應用程式像`telnet`和`ftp`等。通常，多個名稱伺服器必須被詢問以獲得IP位址。這個工作最好由單獨的名稱伺服器來做，因為解析系統會依靠本地名稱伺服器而讓它做所有的工作。

當有需求提出時，一個名稱伺服器應該可以執行解析，但如果它無法做到，它不會想要放棄。它必須能提供另一個名稱伺服器的IP位址（一個參考），讓您更進一步接近要被解析的主機。如果這伺服器也不能提供答案，它也必須把要查詢的問題送到別的主機。在DNS中客戶端—伺服器端(client-server)的結構確保在區域之間的參考和連接被正確的利用，直到最後把地址解析出來。

在上面的*birds.edu*網域，如果*birds.edu* 區域的名稱伺服器被詢問有關在*cuck-*

*oo.birds.edu*區域中一個主機的IP位址，它明顯地不知道答案為何。但是它必須能提供知道這個答案的*cuckoo.birds.edu*區域名稱伺服器的IP位址。

解析是依據一個階層式樣式。如果本地伺服器不能提供*www.cuckoo.birds.edu*（此處*www*是主機）的地址，它會從記錄中檢查看是否它知道關於*cuckoo.birds.edu*的名稱伺服器, 然後是 *birds.edu*，以此類推。在某些參考點時這樣的動作就必須停止，而且所有動作會停在根名稱伺服器**(root name servers)**上。這些伺服器知道所有最上層(top-level)網域的名稱伺服器，像是*com*, *edu*, *ca*, *gb*等等。在網際網路上所有的名稱伺服器都有根名稱伺服器的IP位址，當所有的解析結果失敗時，將會直接的接觸它們。

現在考量一個主機向位在*cuckoo.birds.edu*的本地名稱伺服器提出查詢，它想知道*www.planets.com*的地址，這個本地名稱伺服器顯然不會知道，所以它就直接連線根名稱伺服器。它們其中之一將傳回*com*名稱伺服器的IP位址，本地名稱伺服器現在跟*com* 名稱伺服器做查詢，然後它會參考到*planets.com*的名稱伺服器。這個伺服器知道*www.planets.com* 的地址，然後傳回答案。全部過程發生的很快，儘管根名稱伺服器事實上在每一秒中處理幾千個需求 （僅僅只有十三個，他們全部都執行BSD UNIX）。


Note

此外，一些根名稱伺服器不只提供最上層網域的名稱伺服器之IP地址，事實上它們就是這些網域的名稱伺服器。

24.3　設定DNS——設定主要的伺服器

BIND是UNIX中的名稱伺服器，而且被named服務程式控制著。如果執行ps -e（在Linux中是ps ax）會顯示named， 那麼BIND就執行在您的系統上。最近出版的是BIND 8，使用的版本數字比之前的版本（BIND 4）多4。雖然named從其中一個的*rc*手稿中被執行，另一個有相當多功能的指令手稿ndc也能啟動它，測試您的BIND設定時，您將發現ndc相當方便。

系統使用的設定檔案為/etc/named.conf（在BIND 4中是/etc/named.boot），另外還有兩個到四個之間的額外檔案，其中包含主機名稱IP位址的對照表。我們將以BIND 8來設定不同的伺服器類型，包括主要、次要、只能快取。注意到一部機器只能提供單一類型的BIND服務。

/etc/named.conf，即由named使用的主要檔案，像是檔案系統的超級區塊(superblock)。它含有總結的訊息像是主機所提供的伺服器類型，作為它持有記錄、名稱、資料庫檔案位置的區域。依據名稱伺服器的類型，下面這些檔案能被named所使用：

- hints檔案包含根名稱伺服器的IP地址和FQDN。named 通常保留這些伺服器的列表在它的快取中，當它自己不能解析主機名稱時，而它就能使用它們。
- localhost檔案轉換回授地址127.0.0.1到虛擬的主機名稱*localhost*。
- zone檔案含有主要的資料庫，它包括這個伺服器所服務的區域中所有的主機。這檔案也含有名稱伺服器的IP位址和區域的郵件伺服器。每一個zone都要有一個 zone 檔案。
- 反向查詢(reverse lookup)檔案對應IP位址到它們符合的主機名稱。有些應用程式（像rsh）需要執行這個轉換。每一個區域也要有它們自己的反查檔案。

主要伺服器需要以上的全部四個檔案，次要伺服器只需要前兩個。次要伺服器建立的zone和reverse lookup檔案是從主要伺服器中下載而來。只能快取伺服器只需要hints檔案，但通常也有localhost檔案。我們現在將以網域*planets.com*設定每種名稱伺服器，*planets.com*使用網路位址202.168.0 （而不是202.168.0.0）。

對於網路直接連線到網際網路上的情況，您需要其中一個主機來執行主要名稱伺服器。我們首先將建立四個檔案，將它們儲存在一個目錄中，然後建立設定檔案(named.conf)來指定它們和它們的位置。這些檔案可以取任何名字除了主要設定檔/etc/named.conf之外。*saturn* 將扮演主要名稱伺服器。

24.3.1 Hints檔案

這個hints檔案包含能被named了解格式的根名稱伺服器的列表。在網際網路上共有十三個，它們提供所有最上層網域的名稱伺服器地址。如果您要知道*netscape.com*之名稱伺服器的IP位址，他們只會告訴您這個地址，我們把這個檔案命名為named.cache：

```
# cat named.cache
.                       3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.     3600000      A     198.41.0.4
.                       3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.     3600000      A     128.9.0.107
.                       3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.     3600000      A     192.33.4.12
.....
.                       3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.     3600000      A     202.12.27.33
```

這內容已被修改成只顯示四個根名稱伺服器的項目，當named啟動時，它讀取這個檔案，並與表列的其中一個根名稱伺服器連絡，然後得到最新的列表。它接著將這個列表保留在它的快取上。您應該定期的從*ftp://ftp.internic.net/domain /named.root*取得最新的檔案。您不需要編輯這個檔案；它總是保持一個能被named了解的格

式。

對於每一個根伺服器，這裡有兩個記錄，即NS和A，我們不久將討論它們。A記錄儲存了IP位址和伺服器的FQDN。這顯示了DNS如何維護地址的對照表，我們將看到這種格式重複的出現在zone檔案中。

Note

在您的Linux機器上，您將發現hints檔案在您的版本包裝中，可能是/*var*/*named*/*named.ca*或是/*var*/*named*/*root.hint*。如果您想要，您也能改變它的名稱。

24.3.2 localhost檔案

localhost檔案提供需要用來執行機器的回授位址解析的訊息。named 應該能轉換地址127.0.0.1到名稱*localhost*，我們將這檔案命名為planets.local：

```
# cat planets.local
@          IN          SOA       localhost.      root.localhost. (
                                   2000061601      ; Serial number
                                   28800           ; Refresh
                                   1400            ; Retry
                                   604800          ; Expire
                                   86400)          ; TTL
           IN          NS        localhost.
1          IN          PTR       localhost.
```

這檔案格式由主要伺服器所使用的兩個另外的檔案所分享，因此您應該了解它們建構的方式。檔案是由**Start of Authority**(SOA)記錄起頭，它在這一行中最前面有@字元。這個@字元參考到目前的起源 **(origin)**，即這個檔案所表示的區域。在我們的例子中，這個起源是0.0.127.in-addr.arpa 網域（24.3.5）。接著是關鍵字IN（網際網路：Internet）和SOA。接下來是主機名稱(localhost.)，接著是郵件應該被送達的使用者之郵件地址(root.localhost.)。

在這裡有兩件事要注意。最後兩個名稱由.來結束。通常，您必須確定當使用這些設定檔案時，主機的FQDN總是在結尾上加上點。如果不是，named 會認為它是一個主機名稱，於是試著在它後面加一個網域名稱。也注意到郵件地址以root.localhost.顯示而不是root@localhost，因為在named 中@代表了起源（origin）。

SOA記錄有五個欄位在圓括號範圍內（；之後是註解）。其中的序號是當這個檔案有任何改變時就必須更新。次要伺服器會定期地檢查主要伺服器檔案的這個項目來決定是否重新載入資料庫。您能有一簡單的序號或一個用日期作基礎(date-base)的*YYYYMMDDNN* 格式，*NN*用來指定序號。

更新週期意味著次要伺服器每28,800秒將檢查序號來決定是否應繼續區域傳送。理想的狀況，它應該是一個較低的值，而且不需要設的比我們還要長。您可以

在/etc/named.conf中放一個notify敘述，通知次要伺服器更改已發生。

如果主要伺服器無法回應，次要伺服器會每隔1400秒重新嘗試來取得資料。如果主要伺服器無法工作，然後次要伺服器又不能更新這檔案時，它也應該在另一個604,800秒（七天）的時間內繼續去回應名稱伺服器的查詢。TTL（存活時間：Time-to-Live）欄位顯示當某一個欄位沒有指定數值時所有記錄會用到的預設值，這裡只設定一天。

SOA記錄之後是兩個更多的記錄，第一個是NS（名稱伺服器：name server）記錄，它顯示localhost本身就是區域的名稱伺服器。我們稍後將解釋PTR（指標：pointer）記錄，但是請暫時接受它是替反解網域*0.0.127.in-addr.arpa* 做反向解析的記錄。此處，主機使用的地址1（即127.0.0.1）會被轉換到名稱localhost中。

24.3.3 區域(Zone)檔案

zone檔案是儲存所有主機名稱，即地址的對照表，它用來取代/etc/hosts大部分的功能。我們必須要有一個單獨的區域檔案來設定*planets.com*區域。這個檔案是有點複雜 （圖24.2），在我們開始討論它的內容之前，請先看看這個圖吧。

這個檔案在這裡被區分成三個部分。它包含所有您需要去了解的內容，也就是在一個中型(medium-sized)網路下如何設定一部主要名稱伺服器。SOA記錄標示了主要名稱伺服器的FQDN為*saturn.planets.com*，所有關於BIND的郵件必須被寄到*root@saturn.planets.com*中。至於序號和時間相關(time-related)的參數已經被討論過了。

一個區域檔案是由顯示的資源記錄**(resource records)** (RR)來設定其屬性；您已經看過它們部分的內容。每一個資源記錄有四個到五個欄位，記錄類型被顯示在第三個欄位，IN一定是顯示在第二個欄位。頭尾的欄位包含了主機名稱和IP地址，我們可以在這裡發現RR的四種類型：

- NS——名稱伺服器記錄。這記錄被用來指出主要和次要伺服器的位置。
- A——地址記錄。這包含了主機名稱—地址的對照表，與 /etc/hosts比較起來它是一個稍微擴大的格式。
- MX——郵件交換記錄。DNS也指出被用來接收一個網域郵件的這些伺服器。
- CNAME——您能用來取代一個主機正式名稱 **(canonical name,** offical)的別名。舉例來說，即使我們以*www.planets.com*來存取*neptune*，但*www.planets.com*的正式名稱 仍舊是*neptune.planets.com*。

有兩個名稱伺服器(NS)被記錄在這裡。第一個提供了在*saturn.planets.com*的主要伺服器和在*uranus.planets.com*上的次要伺服器。因為它們自己本身是完整的FQDN，必須用一個點接在它們的最後面。兩個檔案在這裡的第一個欄位都是空

圖 24.2　planets.master：**區域檔案**

```
@        IN     SOA     saturn.planets.com.   root.saturn.planets.com. (
                                3            ; Serial number
                                28800        ; Refresh
                                1200         ; Retry
                                604800       ; Expire
                                86400)       ; TTL
; Name servers and mail servers
                 IN         NS        saturn.planets.com.
                 IN         NS        uranus.planets.com.
                 IN         MX  10    jupiter.planets.com.
                 IN         MX  20    saturn.planets.com.
; Hosts of this zone
localhost        IN         A        127.0.0.1
saturn           IN         A        202.168.0.1
jupiter          IN         A        202.168.0.2
uranus           IN         A        202.168.0.3
planets.com.     IN         A        202.168.0.3
neptune          IN         A        202.168.0.4
                 IN         MX  10   jupiter.planets.com.
; Aliases
mail             IN         CNAME    jupiter.planets.com.
www              IN         CNAME    neptune.planets.com.
ftp              IN         CNAME    uranus.planets.com.
```

的，這個欄位是可選擇的，當您不提供它時，named知道它表示最後指定的zone（在這例子中，就是目前的zone，*planets.com*）。

接下來是郵件交換記錄(**MX**)，*jupiter*是最優先的郵件伺服器，它有一個10的優先權。主要名稱伺服器*saturn* 也是一個備援郵件伺服器，但有一個較低的優先權20。named 將指示所有進入*planets.com*區域的郵件，首先到*jupiter*中，如果失敗的話，郵件將被*saturn*接收。如果工作站不能存取網際網路，那麼網路管理員必須要設定一個在*saturn* 和內部子網路之間的連線，以預防*jupiter* 無法工作。

下一個段落，包含一個資料庫，其中是主機名稱在左邊和IP地址在右邊的格式。在這裡，大部分主機名稱之後沒有一個點，表示named在檢視它的IP位址之前將附加上網域名稱。它們其中之一 (*neptune*)也含有一個單獨的**MX**記錄。BIND允許這樣，這意味著所有郵件位址為*user@neptune.planets.com* 也將被*jupiter*所接受。注意到*jupiter*也接受從*user@planets.com*來的郵件。

您會期望主機名稱也有別名，BIND透過**CNAME**記錄來提供它們。這個記錄有別

名在左邊和主機的正式名稱在右邊。它讓您使用*www.planets.com* 來存取網頁網站，它的主機在*neptune*上。此處，*www* 簡單的扮演主機*neptune*的別名。您也能用A記錄來建立別名，但CNAME的優點是可讓您改變*jupiter*的IP位址而不需要去修改CNAME的項目。

這裡有一個特殊的項目顯示：一個網域名稱而沒有主機名稱，如*planets.com*。因為它自己本身已是完整的FQDN，它需要用點結束。即使*planets* 不是一個主機名稱，您被允許使用*uranus.planets.com* 和*ftp.planets.com*，還有 *planets.com* 的地址來存取主機*uranus*。我們經常在網際網路上以這種方式來表示 ；我們使用*internic.net* 之外還用*www.internic.net*。


Caution

絕對不要用一個CNAME記錄指向另一個CNAME記錄。它必須只能指向一個A記錄，並且MX記錄必須指定主機的正式名稱而不是它的CNAME(alias)。

24.3.4 反向區域檔案

我們應該要有的第四個檔案是可以滿足一個反向解析的需求，換言之，轉換一個IP地址到一個網域名稱。許多的UNIX工具需要做這樣的工作，而且有針對此目的之特別zone檔案。named 在*in-addr.arpa*網域中執行所有的反向解析。它聽起來不可思議，但接著該如何做呢。*planets.com*的反解網域要以*0.168.202.in-addr.arpa* 來表示，即使用反向的IP八位元組。

反向這些八位元組看起來還算合理，因為網域名稱和IP地址是以相反方向被建構成的。IP地址最左邊的八位元組是它最高的位組，而高的位組同樣也是在網域名稱的最右邊。將IP地址的結構反轉讓它與網域名稱一致。檔案planets.reverse向您顯示此反向的項目：

```
# cat planets.reverse
@       IN              SOA     saturn.planets.com.   root.planets.com. (
                                        1               ; Serial number
                                        28800           ; Refresh
                                        7200            ; Retry
                                        604800          ; Expire
                                        86400)          ; TTL
                IN          NS      saturn.planets.com.
                IN          NS      uranus.planets.com.
1               IN          PTR     saturn.planets.com.
2               IN          PTR     jupiter.planets.com.
3               IN          PTR     uranus.planets.com.
4               IN          PTR     neptune.planets.com.
```

這個檔案向您顯示*0.168.202.in-addr.arpa*網域的項目。除了NS記錄外，這檔案展示

了**pointer**記錄(**PTR**)來執行反向解析。因為在這裡我們處理一個Class C的網路，**PTR**記錄的第一個欄位顯示IP地址的最後一個八位元組數字。


Note

在反向解析檔中PTR記錄的第一個欄位，事實上含有反向格式之主機部分的IP地址。如果您有一個Class B網路152.167.0.0，您的反向網域應該是167.152.in-addr.arpa。一個使用IP位址152.167.34.56 的主機，在檔案中的第一個欄位就會是56.34。

24.3.5　named.conf：主要的設定檔案

現在我們已經建立我們需要的所有檔案，我們必須建立我們的總結檔**/etc/named.conf**，它用來指定資料庫檔案的位置和它們服務的用途。這個檔案的結構在BIND 4和BIND 8之間已經有很大的改變。圖24.3顯示了BIND 8版本。

這個檔案含有大括號組成的敘述群組而且用一個分號表示結束。注意到大括號中，它本身也要用;分號來結束。這個檔案中對於伺服器所處理的每一個區域都包含一個**zone**敘述，但要有一個**options**敘述做開頭，這個敘述只是宣告所有zone檔案被放置在**/var/named**目錄中。如果您要的話，您能選擇一個不同的目錄名稱，但是您很少會想這樣做。註解是依C++的風格而在他們前面加上//。

圖 24.3　/etc/named.conf：**一個**BIND 8**版本**

```
options {
       directory "/var/named" ;
};
zone "." {
       type hint ;             // 前被指定成w/“ cache ”
       file "named.cache" ;
};
zone "planets.com" {
       type master ;           // 以前習慣被稱作“ primary ”
       file "planets.master" ;
};
zone "0.168.202.in-addr.arpa" {
         type master ;
         file "planets.reverse" ;
};
zone "0.0.127.in-addr.arpa" {
         type master ;
         file "planets.local" ;
};
```

每一個我們已討論過的檔案都屬於named 了解的一個 zone。.（根）和 *planets.com* 區域屬於正向解析，而這些使用*in-addr.arpa*網域屬於反向解析。所有檔案都是 master類型除了快取檔案外，快取檔案載入根名稱伺服器的IP位址並放在機器的快取中。這些檔案的位置顯示出是放在被**options**敘述指定相對的**/var/named**目錄中。在這裡，我們不用再更進一步深入細節或討論其他可用的選項，因為這裡的設定應該已滿足大部分的用途。

24.4 次要和快取伺服器

在*saturn*上的主要伺服器之zone檔案中定義*uranus* 為次要伺服器。主要和次要伺服器的設定大部分都非常類似；次要伺服器也有快取和localhost檔案並且用同一個內容。然而，次要伺服器建立的zone和反向 zone檔案是從主要伺服器來載入它們。在*uranus*的 **/etc/named.conf**檔案中兩個正向和反向解析的項目看起來像：

```
zone "planets.com" {
      type slave;                                  // 此處現在是次要
      file "planets.slave";                        // 這檔案將被建立
      master { 202.168.0.1 };                      // 當從這裡載入它之後
};
zone "0.168.202.in-addr.arpa" {
      type slave ;                                 // 同樣的註解應用在這裡
      file "planets.slave.reverse";
      master { 202.168.0.1 };
};
```

在這裡有兩個不同的地方。**type**變成是次要伺服器，**master**敘述被用來從主要伺服器載入zone檔案。次要伺服器只需要知道主要伺服器的IP地址。檔案**planets.slave**和**planets.slave.reverse**會藉著區域傳送被建立在次要伺服器中。您可以經由檢查這些檔案的內容來了解您的次要伺服器是否有正常的運作。

即使上面的檔案被定義成次要伺服器的類型，反向網域*0.0.127.in-addr.arpa* 的檔案依舊是主要伺服器的類型。hints檔案中的**options**敘述和**zone**敘述保留原樣未改變。

只有快取伺服器的設定仍舊是簡單的，它只使用hints檔案（還有可選擇的localhost檔案）。因此，只有在**/etc/named.conf**中用一個單獨的**zone**敘述來指向這個檔案。而**options**敘述則保持不變。

24.5 設定解析系統

在UNIX系統的解析系統使用設定檔案**/etc/resolv.conf**，這個檔案在前一章要已介紹過(23.9.1)，用來設定一台主機可以連線到網際網路。在*planets.com*網路上

的每一部主機都有這些項目放在它們的**resolv.conf**：

```
search planets.com
nameserver 202.168.0.1
nameserver 202.168.0.3
```

nameserver的項目指向主機*saturn*和*uranus*。解析系統將嘗試列表中的第一個名稱伺服器，之後它才會試第二個。這裡的**search**敘述可以使用別名和短名稱。如果您使用ftp或 telnet 時用一個沒有包括點的名稱，在**search** 行指定（這裡只有一個）的網域會被附加到名稱之後而且DNS的名稱領域會再搜尋一次。這意味著您能使用ftp uranus之外還有ftp uranus.planets.com。

Linux系統同時使用另一個檔案**/etc/host.conf**（是“host” 而不是 ”hosts ”）。這個檔案包含簡單的兩行：

```
order hosts bind
multi on
```

order的敘述決定了解析的順序。此處，**/etc/hosts** 將首先被搜尋，之後才會存取名稱伺服器。我們將不討論它的下一行。

24.6　ndc和nslookup：測試設定

在Linux系統上，named將它的PID儲存在**/var/run/named.pid**中。這意味著您能使用掛斷（hangup）訊號來殺掉和重啟服務程式，即kill -HUP `cat /var/run/named .pid`。然而，ndc提供一個更好的方式來處理這個服務程式。這是一個指令手稿，它接受**start**, **stop**和**restart**的參數：

```
ndc start
ndc stop
ndc restart                                    像ndc stop 緊接著用 ndc start一樣
```

您也能使用nslookup 命令來測試named是否知道名稱伺服器的名稱和郵件交換主機。這個命令也能以非互動方式加上FQDN來獲得主機的IP位址。此處，我們將使用互動方式和設定**type** 為 **ns**來找尋名稱伺服器：

```
# nslookup
Default Server:  saturn.planets.com
Address:  202.168.0.1
> set type=ns                                  只有名稱伺服器的記錄
> planets.com
Server:  saturn.planets.com
Address:  202.168.0.1
planets.com     nameserver = uranus.planets.com
planets.com     nameserver = saturn.planets.com
uranus.planets.com     internet address = 202.168.0.3
saturn.planets.com     internet address = 202.168.0.1
```

當我們設定**type**為**mx**時，我們將知道named是否認識郵件交換主機：

```
> set type=mx                                          只有郵件交換機的記錄
> planets.com
planets.com     preference = 20, mail exchanger = saturn.planets.com
planets.com     preference = 10, mail exchanger = jupiter.planets.com
        .........        Name server lines deleted
saturn.planets.com     internet address = 202.168.0.1
jupiter.planets.com    internet address = 202.168.0.2
```

您最好使用這些命令來測試您的設定。在這裡，nslookup能正確地驗證名稱伺服器和郵件交換主機。現在存取一些機器，使用單獨的主機名稱和完整的FQDN。舉例來說，您能在您的瀏覽器上輸入**www.planets.com**來驗證是否一個網頁伺服器被執行在*neptune*上。您也可以用ftp uranus 驗證是否解析系統會展開主機名稱到它的FQDN。


Tip

要測試名稱伺服器的執行是否正確，使用nslookup **和一個**FQDN**像是這樣：**nslookup ftp.planets.com**，看是否**nslookup**會輸出主機的**IP**位址。**

24.7 郵件伺服器

電子郵件可能是第一個服務（除了BIND之外，郵件服務必須首先被執行），您會被要求設定在您的網際網路伺服器上。即使網際網路郵件是有點複雜，但是設定一個郵件伺服器給一個小型機構卻不需要如此。在我們之前設計的場景中，Rational Planets也得處理Velvet的郵件，這樣的確增加了設定的複雜程度。然而，如果您能讀到這裡，您已幾乎確定可以做到。

郵件傳送和投遞的機制已經在13.6節中被討論過了，我們重新整理一下基本的理論， 網際網路郵件是由三個代理程式共同工作的：

- 郵件用戶代理(The Mail User Agent MUA)程式，編寫和讀取郵件。
- 郵件傳送代理(Mail Transport Agent MTA)程式，寄送和接收郵件。
- 郵件投遞代理(Mail Delivery Agent MDA)程式，從MTA接收郵件和投遞它到使用者的信箱。

在您繼續建置這個有點複雜的設定之前，正好可以看一下圖24.1來了解郵件代理間的關係。MTA的工作今日普遍是由SMTP（簡單郵件傳輸協定）處理。sendmail 是SMTP最常見的實現。在下面的章節中，我們將討論和設定sendmail。

當一個使用者間歇性的連線到一個網路（通常使用撥接線路），郵件不可能直接寄送給使用者。第四種代理程式此時開始起作用，它從遠端的伺服器擷取郵件。兩種傳輸協定用來儲存和擷取郵件的方法是POP（郵局傳輸協定：Post Office

Protocol）和IMAP（網際網路訊息存取傳輸協定：Internet Message Access Protocol）。fetchmail 是Linux系統中一個標準的POP/IMAP客戶端程式。我們將學習設定fetchmail來從一個POP3（最新也是最普遍被使用的實現）伺服器收回郵件。

24.7.1　sendmail：最通用的(Universal)MTA

在一個網路中，使用者在工作站上通常將他們的郵件送到一個**hub**（一個郵件伺服器），而它能直接與外部的世界連接。同樣地，工作站也不能直接接收郵件；所有進來的郵件進到hub中。這方法可保護個人使用者，避免捲入郵件處理的麻煩。如果一封郵件訊息不能被送出，在告知寄件者前hub的工作就是持續嘗試五天。

當hub繞送這訊息時，它改寫寄件者的地址以便顯示出起源於hub。換言之，它應該重寫地址*henry@neptune.planets.com* 到*henry@jupiter.planets.com*，*jupiter* 是Planets 中用來處理郵件的hub。如此外部的世界就不知道henry實際用來傳送郵件的機器，而此為一個非常有用的安全功能。這意味著henry能移到另外一台不同的主機，卻仍然能使用同樣的電子郵件地址。如果hub處理全部*planets.com*網域的郵件，那麼大部分的內部機構也能隱藏hub的名稱，如*henry@planets.com*.

因為進來的郵件經由hub繞送，使用者有三種選項：

- 用telnet或rlogin登入後在hub檢視郵件訊息。如果他們在hub上有帳號，全部Planets的使用者都能如此做。
- 要求信件被轉送到他們的機器上，因此，他們自己的機器必須隨時開機。
- 從hub上的一個POP/IMAP伺服器上取回郵件。只有這個方法Velvet能從Planet的郵件伺服器上收回郵件。如果他們不要郵件轉送到他們的機器上，那麼Planets上的使用者也能使用這個功能。

上面簡單的介紹了sendmail，它是UNIX系統上最通用的MTA。在UNIX和Linux系統上，有另一個MTA可用，像是MMDF和smail。但是sendmail 可處理大量的網際網路郵件。它的的作者是柏克萊分校的Eric Allman，目前也仍然是他在維護。sendmail 是一個非常複雜和怪異的程式，甚至還是UNIX的標準。人們想到編輯它的設定檔就害怕（但我們不會）。

不像一些其他的傳輸協定如UUCP，sendmail不使用中間主機。它跟另一個在接收端的MTA互相溝通，並且在確認之後，直接傳送郵件到接收端的MTA中。此程式可能是或可能不是另一個sendmail程式，但它必須要了解SMTP。依照可能被設定的轉送功能，在MDA最後接管之前，郵件可以被多個MTA處理。如果一個郵件訊息不能被傳送，sendmail將此訊息放入佇列等候並在告知使用者無法傳送之前

會每隔一段時間區間重試數次。

sendmail 使用了aliasing （別名）的觀念來轉送郵件給任何地方的一個或更多的使用者。這些別名被保留在檔案**/etc/aliases**中；當sendmail接收一個郵件訊息時，它會讀取一個轉化過的別名檔案。

24.7.2 啟動和停止sendmail

sendmail位於**/usr/sbin**（在Solaris中，**/usr/lib**）。像pppd，它也有伺服器端和客戶端的組件，但它能將兩種模式(client, server)同時運作在單一個指令的執行中。它從其中一個*rc*手稿中被執行成為伺服器服務程式，並在埠25傾聽進來的全部郵件，。如果sendmail只接收進入的郵件，而且不送郵件到外面，它必須以下面的方法被執行：

```
/usr/sbin/sendmail -bd                                    服務模式
```

-bd選項讓sendmail以一個服務程序方式被執行。sendmail，像任何的MTA，不會執行郵件的傳送。除了它轉送郵件至另一個MTA中，它將郵件送交給傳送程式。UNIX系統把使用者的郵件維護在**/var/spool/mail**（在SVR4中，**/var/mail**）目錄的單獨文字檔中。一個給henry的郵件訊息會附加到檔案**/var/spool/mail/henry**中。

要寄出的郵件會被移到目錄**/var/spool/mqueue**中排隊。如果sendmail只寄出外送(outgoing)的郵件，那麼它必須以客戶端的方式被執行並用下面兩種方法之一：

```
/usr/sbin/sendmail -q                                     執行一次
/usr/sbin/sendmail -q 15m                                 每15分鐘檢查佇列一次
```

在第一個例子中，sendmail以執行一次方式處理佇列。您能使用cron 服務在固定的時間來執行這個命令。或者您也能使用第二個形式，它每十五分鐘查看佇列一次並進行清理。儘管將sendmail 執行在一個服務模式中需要系統管理者的權限，**-q**選項能被所有使用者使用。

即使sendmail 配合**-bd**選項被執行成服務程式，您也能配合**-q**選項再次執行它。在很多的例子中，同一台機器處理進來和出去的郵件。這是在固定連線到網際網路的一個主機上執行sendmail 時最常被用到的方法：

```
/usr/sbin/sendmail -bd -q 15m
```

sendmail 的特性被**/etc/sendmail.cf**（在Solaris中，**/etc/mail/sendmail.cf**）檔所控制。當您改變這個檔案內容，確認您殺掉和再重新啟動 sendmail。 您也可藉由適當的手稿（例如，/etc/rc.d/init.d/sendmail）再加上 **restart**的參數。 如果*rc*手稿在您的機器上只支援**start**和**stop**的參數，那您可以用ps命令找到sendmail 的PID之後，試著用**SIGHUP**訊號來殺掉它。

在Linux的系統中，sendmail 儲存（在**/var/run/sendmail.pid**中有兩個獨立的行）PID和用來執行它的命令列。先kill儲存在第一行的這個PID，緊接著執行第二行的內容：

```
kill `head -1 /var/run/sendmail.pid`                    停止sendmail
`tail -1 /var/run/sendmail.pid`                         啟動sendmail
```

我們還未使用過顯示在第二行中的命令替代(command substitution)，但這是一個聰明的方法來執行置於(embedded)一個檔案中的一個命令列。更聰明的人可能會為這個命令建立別名。

24.7.3 mailq：看郵件佇列

用mailq命令能看到郵件佇列。它只是一個符鏈連結指向sendmail命令，實際上相等於執行sendmail -bp：

```
# mailq
            Mail Queue (1 request)
--Q-ID-- --Size-- ----Q-Time---- ------------Sender/Recipient------------
QAA00943      478 Fri Dec 10 16:22 <sumit@saturn.planets.com>
                                   <XLDEL/TMH/VIBHA@tmh.satyam.net.in>
```

這會顯示訊息識別碼(message-id) (**QAA00943**)，寄件者的電子郵件位址在第一行並顯示詳細的工作狀況。接收者顯示在第二行。message-id被置放在sendmail 為每個郵件訊息所建立的兩個檔案名中。用ls 指令顯示 **/var/spool/mqueue**中的這兩個檔案：

```
# ls /var/spool/mqueue
dfQAA00943
qfQAA00943
```

每一個郵件訊息以這兩個檔案的形式被保留在這個目錄中。“ df ”檔案含有郵件內容，而“ qf ” 檔案包含郵件標頭。每一位使用者都能執行這個命令來看佇列的訊息。可惜，沒有命令可以從佇列中移除郵件訊息。如果一個訊息必須被刪除，系統管理者必須使用rm來刪除對應的“ df ”和“ qf ”檔案。

24.8 sendmail.cf：設定檔案

sendmail 內容可以在它的設定檔**/etc/sendmail.cf**被發現。這個檔案讓很多人感到恐懼，它幾乎包含了一千行，其中的一部分內容可能是您看過最神秘的編碼。幸運的是，sendmail 能被設定在最小型和中等系統中而不需要讓您去了解它的大部分內容。唯一您必須要知道的部分，在下一個段落中會談到。

您很少會去從頭開始編寫一個**sendmail.cf**；所有的UNIX系統附有一個事先裝置好的這個檔案。您只需要改變此檔中少數的幾行就能使sendmail 運作。然而，很多使用者也害怕做這個而離開。在了解這個簡化sendmail 設定的需求後，sendmail 的程式開發者們建立了m4 的巨集程序(macro-processor)。在這裡，您在一個單獨的檔案中設定一些變數，讓m4產生**sendmail.cf**。我們將不討論m4 而寧願直接地面對和修改**sendmail.cf**。

sendmail.cf 被區分成許多的段落，每一個部分都在處理它建置上的不同工作。在這節中，我們將學到解譯一些它的巨集和類別。當我們設定我們的郵件伺服器時，我們也將遇到一些其他的參數。sendmail 不同於其他的設定手稿；全部的敘述必須從欄位**1**開始。

24.8.1 巨集(Macros)

巨集可以用簡單的變數來看待，它們可在檔案的開頭部分被找到。所有的巨集定義開始於一個字母**D**（被 **sendmail.cf** 使用的字母**M**代表郵寄者，mailer），使用 grep “^D” /etc/sendmail.cf，您能簡單的找到它們的位置。一些重要的巨集顯示如下：

```
Dj$w.planets.COM
DSscarletisp.com
DMplanets.com
DnMAILER-DAEMON
DZ8.9.3
```

*sendmail**的版本編號***

字母**D** 緊接一個字元的巨集名稱，之後是它的值，且全部不需要間隔。在這裡**S**被設定為**scarletisp.com**，**Z**有一個值**8.9.3**。像是shell的變數，所有的巨集變數在被計算時是用一個**$**符號，您將會發現它們在**sendmail.cf**中不同的區段被參考。

許多sendmail 巨集是用小寫字母開始。您可能已經看過他們，即**$w**。所有的小寫字母不被定義在檔案中，但被定義在sendmail的內部，它們在最開頭的區段就有許多。在這裡有一些行顯示sendmail 如何使用巨集來決定訊息標頭的格式：

```
H?D?Date: $a
H?F?From: $?x$x <$g>$|$g$.
H?x?Full-Name: $x
H?M?Message-Id: <$t.$i@$j>
```

每一標頭由**H**來識別，接著是一個旗標包含在一些**?**和標頭的範本。**$a**儲存系統日期並使用RFC 822的格式，**$g**是寄件者的地址，**$x**是全名。Message-Id是一長字元的字串，通常是有一些數值像**<199908230446.KAA00913@mail.hill.com>**。它是從日期、時間用數字表達(**$t**)的方式而取得，佇列識別(queue-id) (**$i**)和主機的FQDN

(**$j**)。您能充分的體會，沒有兩個郵件訊息能在任何時刻有同樣的Message-Id。


Note

在訊息標題的From:**行讀起來很有趣。在這裡，變數是被有條件的設定。它可能是$x <$g>（**RFC 822**格式）或是$g 要依據$x是否被設定。$?, $|和$. 扮演了以**if**, **else和 endif 關鍵字組成的任何if結構。它可以解釋成：“**If x exists ($?x), the value is $x <$g> else ($|) it is $g endif ($.).**”如果全名沒有出現在**From:**行，那麼這就是您需要去檢查的行。**

24.8.2　類別

sendmail 也使用字母**C**來表示一個類別。像巨集一樣，在它後面也跟著一個單一字元和一行的一個或多個數值。這裡有一些類別，它將幫助您更了解sendmail的運作：

```
CO @ % !
Cwlocalhost mail.planets.com planets.com velvet.com
CE root
```

第一個類別**O**（字母“Oh”）含有三個值由空白來間隔。這個類別禁止您使用字元**@**, **%**和**!**在電子郵件地址中使用者名稱的部分。

類別**w**包含全部主機的FQDN，而**w**之後是預期要接收郵件的主機。您能指定全部的這些FQDN放在一行之中，但是當那兒有太多FQDN時，您能使用檔案類別(**F**)來代替。檔案的類別使用**F**字元並以同樣的單一字元（這裡是**w**）來指出列表可以從一個檔案中讀入。在這裡，如果您有一百個FQDN要處理，您或許可以使用**w**：

```
Fw/etc/sendmail.cw                                    在這定義所有的別名
```

在Rational Planets，全部的郵件被*jupiter*和它的別名*mail*接收。當有一個特定MX記錄(24.3.3) 指向正式名稱*jupiter*，卻沒有指向別名*mail*（MX記錄不能指向別名主機)。一旦您加入*mail.planets.com*到別名類別（不是用**Cw**就是**Fw**），系統也將接收到地址是*user@mail.planets.com*的郵件。請注意這裡它也接收寄到 *user@velvet.com*的郵件。這就需要一些額外的設定，可藉由把Velvet的項目放到 **/etc/mailertable**中，但我們將在本書中忽略這些問題。

24.8.3 選項(Options)

sendmail 提供超過一百個選項來定義它的檔案位置，處理錯誤訊息和逾時等。選項用字母**O**來定義，定義的方式已經從**8.7**版之後改變，一個選項要用以下這些方法來表示：

```
OA/etc/aliases                                    8.7版之前
O AliasFile=/etc/aliases                          8.7版之後
```

第一個形式使用單一字元和它的值而中間沒有任何空白。在這裡，選項OA的值是/etc/aliases。第二個形式使用一個完整的關鍵字AliasFile當作選項的名字。此外，在它之前有個空白之後接著一個 = 。大小寫在這裡是不重要的；Aliasfile和aliasfile有同樣的意思。以下一些重要的選項：

```
O HelpFile=/usr/lib/sendmail.hf
O ForwardPath=$z/.forward.$w:$z/.forward
O QueueDirectory=/var/spool/mqueue
O Timeout.queuewarn=4h
O Timeout.queuereturn=5d
```

第二行指定sendmail要找尋.forward檔時須依循的路徑。它使用兩個巨集$z和$w。您幾乎能猜到這些巨集代表的意義：$w表示主機名稱和$z 表示家目錄。

您一定看過從MAILER-DAEMON 傳來的訊息表示它不能傳送一個郵件。最後兩個選項決定sendmail在它放棄之前要隔多久嘗試重新傳送郵件。如果sendmail在四小時之內還不能傳送郵件，它會送出一個警告訊息給寄件者，但仍繼續嘗試。如果在五天之內它依舊不能送出，它就會放棄並從mqueue目錄中刪除郵件。

24.9 別名(Aliases)

sendmail 有一個非常有彈性和多樣的特色，它可以繞送郵件到任何的主機上的任何使用者。由於**aliasing** 的功能而讓此特色變為可能，它被表示在檔案/etc/aliases中。檔案中的每一行區分成兩個部分，左邊的部分包含需要被做成別名的使用者名稱，右邊則包含目的地（信件要送達的位置）。在此檔案中您會發現一些令人意外的項目：

```
postmaster: root
mailer-daemon: postmaster
enquiry: henry
charlie: charlie@neptune.planets.com
friends: romeo,juliet@yahoo.com,jack,jill,andrew
navlist: :include: /home/nav/subscriber.lst
hegel: /dev/null
```

這裡的一些項目展示了您能使用這個檔案的不同方法。每一個使用者名稱之後是一個：然後加上空白。右邊包含別名的展開。這個展開的內容可能有多個使用者名稱或是檔案。所有寄到enquire的郵件會被轉送到henry。

planets.com 網域的主要郵件伺服器是*jupiter* (24.3.3)。因此郵件地址到charlie 通常會被佇列(spool)到*jupiter*中，但是在這裡的別名會重新轉向到主機*neptune*中。這

個別名有一些特殊意義。即使全部的使用者以*user@planets.com* 形式在hub 接收郵件，別名讓您重新轉向一些或全部郵件到它們實際上「屬於(belong to)」的機器上。

當sendmail檢查收到的郵件訊息標頭，它首先檢查使用者名字在**To:**欄位並與**/etc/aliases** 比較。它將別名展開，然後繼續檢查展開後是否又有另一個別名。此處，郵件地址是mailer-daemon最終會進入系統管理者(root)的信箱中，因為postmaster也是別名轉向到 root。

讓我們現在看看另外的別名。郵件地址寄給friends會被五個使用者收到，他們其中之一甚至沒有在本地網域而在網際網路上；sendmail 可以允許這樣。friends在這裡表示一個郵寄名單，它允許一群使用者的地址用單一名稱來代表。五個使用者已足夠容納在**/etc/aliases**中，但是當那兒有五百位朋友呢，所以他們最好放在一個檔案中。使用**:include:** 關鍵字，您能指示sendmail 來查驗檔案**subscriber.lst**來展開別名navlist。

收件者也能是一個檔案。如果hegel已離開公司又沒有留下他的新地址，那麼全部寄給他的郵件應該被擋住。沒有比直接送訊息到**/dev/null**來做這件事的其他更好的方法。

當使用者加入或離開時系統管理者能修改這個檔案來增加或刪除別名。但之後她必須使用newaliases命令來編譯(compile)它。執行命令時不需要任何參數。像mailq一樣， newaliases也是另一個連結指到sendmail 並相等於執行sendmail -bi。它建立二進位格式的檔案**aliases.db**。當sendmail檢查一封信的標頭時，它讀取**aliases.db** 檔而不是**aliases**檔。


Caution

絕對不要更改您可以找到任何一個aliases 檔案中postmaster **和**mailer daemon**的別名。**


Note

除了使用/etc/aliases之外，每一個使用者能做她自己的轉向而藉由將目的地的電子郵件地址放在 $HOME/.forward。臨時轉送最好藉由.forward擋，這樣就不用麻煩系統管理者。然而，當一個使用者離開公司或被調走， /etc/aliases是一個比較好的選擇，因為使用者的帳號可能被刪除，她的家目錄可能被移除， .forward也將會不見。

24.10　為planets.com設定郵件伺服器

在我們的*planets.com* 的網路中，網域資料庫(24.3.3) 的**MX**記錄指定*jupiter*和*saturn* 負責處理全部網域的郵件。 *jupiter*(**10**)有比*saturn*更高的權限 (**20**)。因為全部的主機轉送他們的郵件到 *jupiter*中，他們不需要將sendmail 執行在服務程式的模式

下 (-bd)，但只須執行在佇列模式下（用 -q)。只有 *jupiter* 和*saturn* 需要執行sendmail 的服務模式。

在這個設定中，全部主機除了*jupiter*和*saturn* 外，可以用一個簡單的sendmail.cf的設定，舉例來說，在*neptune* 中的檔案應該包含一行以宣告郵件要被轉送到一台「聰明(smart)」中繼主機，就是*jupiter*主機。這個動作可以使用*neptune*的sendmail.cf檔的S巨集來做到：

```
DSjupiter.planets.com                                    設定聰明的中繼主機
```

為什麼我們需要一部中繼伺服器**(relay server)**？凡是不能連線到網際網路或沒有能力傳送郵件等限制因素的主機就需要這樣的伺服器。 寄信的工作通常是件艱鉅的任務。收件端的機器可能無法工作，或是因為線路中斷沒有路徑連到主機。一部負責的郵件伺服器（也是一部中繼伺服器）在它放棄前，必須在合理的時間區間內持續與在另外一邊的MTA溝通。

通常依照組織的策略會將在電子郵件中的主機名稱隱藏起來。在這樣的例子中， *jupiter* 必須被告知它要偽裝**(masquerade)** 成*planets.com* 而不是 *jupiter.planets*.com。您必須用下面的方法在 *jupiter*中設定M巨集：

```
DMplanets.com                                            偽裝成planets.com
```

這意味著在 *jupiter*的一個使用者henry 在寄送郵件時，寄件者的名稱被改寫成 *henry@planets.com* 而不是*henry@jupiter.planets.com*。這對在*jupiter*上的使用者不會造成問題，但對Planets來說可能要有一個統一的結構來處理整個*planets.com*網域的寄件者，包括這些在*neptune*, *uranus*, *saturn* 等等。在這樣的例子中，偽裝的動作必須設定在所有轉送他們的郵件到*jupiter*的主機。這次我們使用類別M：

```
CM neptune.planets.com saturn.planets.com uranus.planets.com
```


Note

用M類別(CM)來偽裝一整個網域，您必須確定這些使用者名稱在整個組織中是唯一的。這也是一個健全方法來組織使用者帳號。如果使用者名稱在整個網域中不是唯一，您就不能使用CM。

Planets的郵件服務現在已準備好給它自己使用。但Planets也能處理Velvet的郵件服務，sendmail 必須知道這個。這就是w類別的工作：

```
Cwlocalhost planets.com velvet.com
```

此處是建置在它上面全部的郵件設定。然而，我們還沒有準備好因為仍舊必須決定有關處理接收到的郵件。

在我們的範例設定中， *jupiter*固定連結到網際網路上並將全部使用者的郵件儲存在/var/spool/mail中。使用者如何存取郵件？如果使用者用telnet的方式登入到*jupiter*，他們能使用任何MUA像elm或pine看到他們的郵件。如果他們使用Netscape（或是他們機器原本的MUA），那麼他們必須下載他們的郵件到他們的機器中。它有許多解決的方式，而Planets能為不同的工作站選擇不同的解決方式：

- 如果一部主機總是在啟動中而且連結到*jupiter*，那麼您能使用/etc/aliases為那個主機上的使用者轉送郵件。 *jupiter* 不需要在/etc/passwd內有那些使用者的帳號。
- 如果全部的主機固定的連線到*jupiter*，那麼全部的郵件能轉送給他們，或是使用者能被允許使用NFS以遠端方式掛載 /var/spool/mail檔案系統。
- 如果一部主機間歇性的連線到*jupiter*(可能經由一個撥號線路)那麼*jupiter* 也必須扮演一個POP或IMAP伺服器。在這主機上的使用者能使用POP或IMAP傳輸協定來取回他們的郵件，而他們必須在主機 *jupiter*上要有帳號。

我們知道如何使用/etc/aliases，而且我們也知道如何掛載一個NFS檔案系統。現在，請參考圖24.1，您將注意到主機*mercury*是經由一條撥接線連結到*jupiter*。這表示可選第三個選擇，而且使用這種選項的主機需要個別的郵件設定。

24.11　POP和IMAP：為離線使用的傳輸協定

當接收端郵件伺服器必須固定的連線到網際網路上（否則，接收的郵件將被拒絕），但工作站就不需要經常連結到這個伺服器。使用者通常在夜晚的時候關閉他們的機器，如此就排除經由/etc/aliases轉送郵件的所有機會。個別的使用者也會在規則的時間間隔下經由一條撥接線路連線到他們ISP的郵件伺服器。這些形式的請求是「取回(pulling)」郵件而不是「 送出(pushing)」它。POP和IMAP就是在這種狀況下使用的傳輸協定。

在Planet的網路中，如果使用者不被允許掛載遠端的/var/spool/mail，那麼 jupiter 必須保留郵件在一個POP或IMAP伺服器上。我們將考慮POP3（最新可被利用的），因為它是大部分網際網路伺服器上的標準；IMAP是一個尚未被採用的一個較佳的標準。當POP3被啟動時，郵件信箱的位置仍舊放在相同目錄，除了他們能從一個遠端機器配合這個協定來取用。

POP3是被inetd從/etc/inetd.conf檔來啟動。為了讓它執行， *jupiter* 必須在這個檔案中包含類似的這些行：

```
pop3     stream  tcp     nowait  root    /usr/sbin/tcpd  /usr/sbin/popper -s
pop-3    stream  tcp     nowait  root    /usr/sbin/tcpd  ipop3d
```

在Red Hat和SuSE Linux上的這些設定上，由包裝程式tcpd 執行popper或ipop3d 程式(23.7)。POP3使用埠編號**110**，這應該能反應出在**/etc/services** 中同樣的一個項目：

```
pop3            110/tcp         # POP version 3
pop-3           110/tcp         # POP version 3
```

確定這行沒有在 *jupiter*上的**/etc/inetd.conf**被註解掉。我們現在將考慮為那些需要從*jupiter*的POP伺服器下載郵件的主機設定郵件。這包含了經由撥號線路（像*mercury*）連線的主機，還有那些被直接連線卻寧願使用POP。

24.11.1 fetchmail：一個POP/IMAP客戶端

Linux使用一個聰明的程式稱為fetchmail 來從POP或IMAP伺服器取得郵件。這個客戶端程式需要使用者名稱，傳輸協定和郵件伺服器的FQDN當參數。以下是 *mercury* 上的使用者romeo連結到 *jupiter* 來接收他的郵件：

```
$ fetchmail -u romeo -p POP3 jupiter
Enter password for romeo@jupiter: *******
2 messages for romeo at jupiter (12140 octets).
reading message 1 of 2 (2354 octets) .. flushed
reading message 2 of 2 (9786 octets) ......... flushed
```

fetchmail 會提示要求密碼，而密碼不會顯示在終端機上。它接著從POP伺服器上取得郵件。這些郵件訊息會從伺服器上刪除，這也是fetchmail 的預設動作。在這過程中只有新的郵件訊息能被取走。

fetchmail被當作是執行SMTP的主機，像*mercury*（或是*moon*）和SMTP執行在郵件伺服器jupiter上或是ISP伺服器（像*scarletisp*）之間的連結通道。然而，整體的介面是一致的，沒有任何環節被繞過(bypassed)。讓我們把這個弄清楚； *mercury* 也必須在服務模式下執行sendmail，而且 fetchmail 必須將接收到的郵件交給sendmail。 sendmail接著將此郵件送給一個遞送程式像是procmail。這就好像*mercury* 直接地連結到 *jupiter*的情況。同樣考量到適用於主機*moon*，它會處理Velvet的郵件。

當您用您的網路做試驗時，您可能要這些郵件訊息被保留在伺服器上(**-k**) 直到您確定您要如何為止。預設上，fetchmail 只接收新的訊息，但您也能覆寫此設定而要求它取走全部的訊息(**-a**)。您能稍後再刪除它們(**-F**)：

```
fetchmail -u romeo -p POP3 -k jupiter.planets.com          保留全部的訊息
fetchmail -u romeo -p POP3 -a jupiter.planets.com          得到全部的訊息
fetchmail -u velvet -p POP3 -F mail.scarletisp.com         刪除全部的訊息
```

現在取回的信件被放在文字檔案/var/spool/mail/username中。romeo現在能使用任何以字元為基礎(character-base)的客戶端程式像elm和pine來看郵件。使用Netscape，它又是另一個不同的故事，因為它認為全部的郵件應置放於$HOME/nsmail中，然而，使用符號連結，您能建立任何的郵件檔案位於任何Netscape看得到的地方。此外，Netscape能直接地從*jupiter*取得它的郵件而不需要*fetchmail-sendmail* 來繞送。

您或許要執行fetchmail 的服務模式和經由一個手稿來傳送密碼（用明碼）。fetchmail 使用 .fetchmailrc ，全部的 fetchmail 選項都能被儲存在這個檔案中。在這裡有一個簡單範例來使用POP3，它將郵件訊息留在伺服器上：

```
poll jupiter.planets.com proto POP3
user romeo with password d276y45t
keep                                          像-k選項
```

poll敘述使用郵件伺服器當參數，proto 指定所使用的傳輸協定形式。因為密碼在這裡是被存成明碼，檔案必須不能被群組和其他使用者所讀取。在任何的狀況下，如果檔案的權限不是被設定成600，則 fetchmail 就不會執行。


Note

如果*mercury* 也是執行POP伺服器，那麼郵件從主機*jupiter*上的POP伺服器來取得，它會保留在*mercury*的伺服器上。其他的主機就能從*mercury*上取得郵件（也能將它儲存在另一個POP伺服器上）。

24.12　在Velvet上設置郵件系統

我們現在必須在Velvet的主機*moon*上設定郵件伺服器來傳送和接收郵件。注意Velvet能存取Planet的主機 *jupiter* 和用撥接PPP帳號的方式連線ISP的主機。這情形是不同於之前的，這次的郵件必須從兩個來源下載：

- 從*jupiter*上的POP伺服器，Velvet有一個多投落點(multidrop)的信箱（一個信箱分配給多個使用者）。Velvet也必須服務許多使用者，他們使用從*user@velvet.com* 來的電子郵件。
- 從ISP的主機*mail.scarletisp.com*上執行的POP伺服器，Velvet有一個PPP撥接帳號以及電子郵件地址*velvet@scarletisp.com*。

讓Velvet傳送郵件，ISP必須扮演精明的中繼主機。在主機*moon*上的sendmail.cf 因此要被設定以便所有從*moon*送出的郵件會使用PPP來轉送到這個中繼主機。偽裝也必須在*moon*上被實現，以便所有的地址都是以*user@velvet.com*的格式出現：

```
Dsmail.scarletisp.com                         中繼主機
DMvelvet.com                                  開啟偽裝
```

不像在網域*planets.com* 中的主機 *jupiter*，*moon* 只是間歇性連線到網際網路（經由ISP），而這就是為什麼需要一個中繼主機的原因。這次，sendmail 在*moon* 上執行時，必須被告知郵寄是「昂貴(expensive)」的，它不須在接收時馬上就嘗試傳送郵件，這需要您來改變這個選項：

```
O HoldExpensive=True
```

您也必須在寄件器的定義上做一些改變。寄件器被**M**來定義，共有四種SMTP的寄件者形式被定義在**sendmail.cf**中：

```
Msmtp,          P=[IPC], F=mDFMuXe, S=11/31, R=21, E=\r\n, L=990,
                T=DNS/RFC822/SMTP,
                A=IPC $h
Mesmtp,         P=[IPC], F=mDFMuXae, S=11/31, R=21, E=\r\n, L=990,
                T=DNS/RFC822/SMTP,
                A=IPC $h
Msmtp8,         P=[IPC], F=mDFMuX8e, S=11/31, R=21, E=\r\n, L=990,
                T=DNS/RFC822/SMTP,
                A=IPC $h
Mrelay,         P=[IPC], F=mDFMuXa8e, S=11/31, R=61, E=\r\n, L=2040,
                T=DNS/RFC822/SMTP,
                A=IPC $h
```

smtp寄件器處理正常的STMP郵件。**esmtp** 使用擴充的SMTP。未編碼的八位元(unencoded 8-bit)郵件現在由**stmp8**來處理。中繼的(relay)寄件器繞送全部的郵件經由中繼主機。每一個寄件器的定義也有一些參數。**P**顯示[**IPC**]當作一個寄件器程式的路徑；這表示是sendmail 程式。**S**和**R**顯示sendmail 規則，用來重新改寫寄件者和接收者的標頭。

其他的參數顯示一個郵件訊息的單獨行被**\r\n**(**E**)所中止，以及那一行不能超出990個字元 (**L**)。主機名稱會被DNS解析，電子郵件地址是RFC-822形式，sendmail 命令本身(**SMTP**) 被用來執行寄件器。

F表示旗標可應用在寄件器上。大部分這些旗標常用於巨集，像**$a**, **$b**等等。我們不用困擾任何的預設，除了我們要在最後加一個**e**以外。這使得寄件器很昂貴，它不能試著連接後就馬上傳送郵件，而是要等待sendmail -q來執行。


Note

從版本8.9開始，sendmail 預設不會做中繼的工作，並會印出訊息Relay access denied（拒絕中繼處理）。如果您要啟動這功能，以便其他的網路能經由您的郵件伺服器繞送他們的郵件，那麼把他們的網域名稱或他們的網路地址加在/etc/mail/relay-domains中，然後重新啟動sendmail。

郵件的遞送現在可經由撰寫一個手稿來執行，它撥號出去到ISP，在兩端啟動

PPP服務，然後執行sendmail -q命令。您能使用dip, chat（或是一個新的工具wvdial）來撥號，從您的crontab檔案來執行這手稿。因為pppd在連線時執行/etc/ppp/ip-up，您也能把sendmail 敘述放置在那個檔案中。注意sendmail 總是執行在服務模式下(sendmail -bd)。

在這裡順便提一下關於Velvet在 *jupiter*上的多投落點信箱**(multidrop mailbox)**。 sendmail 允許在單一網域中郵件寄給多個使用者時可轉送到一個單獨的使用者帳號。我們必須要做的就是要求fetchmail 從這個帳號來得到郵件。因此我們有兩個電子郵件信箱要處理，一個是*mail.velvet.com*（為 *jupiter*的一個別名），另一個在*mail.scarletisp.com*，在*moon* 上的**.fetchmailrc**將有兩個設定敘述：

```
poll mail.scarletisp.com proto POP3
user velvet with password bnet6797
poll jupiter.planets.com proto POP3                    或是 mail.velvet.com
no dns aka velvet.com                                  這個敘述是必要的
user pop1244238 password hf789bfd is * here
```

第一組的設定從ISP的郵件伺服器下載Velvet上的郵件。第二組的設定使用Planets指定給Velvet的信箱其使用者名稱為pop1244238 。因為這個信箱含有多個使用者的郵件，fetchmail 必須檢查每一個郵件標頭和確認郵件被分送到不同使用者的信箱中。這裡的*隱含著分派的動作。如果這裡您有一個使用者名稱，全部的郵件就會被塞到該使用者的信箱中。

如果一個POP伺服器在*moon*上執行，那麼個別的使用者就可以從他們的Windows機器上使用內建在Netscape Messenger和Outlook Express中的POP客戶端功能來下載他們的郵件。


Note

Netscape Messenger **也是一個POP客戶端程式，也能從一個POP伺服器來取得郵件。然而，**fetchmail **更優越因為它只要下單獨的命令就能從多個伺服器中取得郵件。**Netscape Communicator 6 **也將有這項功能。但**fetchmail **也能從一個多投落點的信箱來接收郵件，而**Netscape**卻不能。**

24.13 網頁服務

全球資訊網是網際網路上最受歡迎的服務。網頁伺服器通常以**httpd servers**的名稱被大家所熟知，亦即用位在埠80執行的服務程式來命名。有幾種httpd伺服器的類形已被使用在全球資訊網上，他們大部分是以NCSA（NCSA的伺服器）最初的設計為基礎。最初的伺服器有許多的缺點，需要在它們上面加上修補程式。一個工作團隊的人員運用有系統的方法執行這個任務並建立「一個修補過的伺服器」（ A

patchy server)。他們稱它為Apache，今天有超過一半的全球資訊網httpd servers是使用Apache。Apache是所有Linux系統上的標準伺服器。它的版本已被開發到所有UNIX的平台上(*http://www.apache.org*)。

網頁伺服器使用HTTP傳輸協定來處理客戶端（瀏覽器）來的需求。通常，這個需求是取得HTML文件和它相關的圖片。使用額外的CGI程式來幫忙(20.20)，網頁伺服器也能提供動態的HTML輸出。網頁伺服器在單獨機器上執行，也能提供多重的網站。安全因此是一個非常重要的客題，Apache有一個綜合的機制來加強目錄、主機和使用層的安全。我們現在先了解大部分的這些功能並由我們自己來設定一個網頁伺服器。

24.13.1 Apache：被修補過的伺服器

Apache在Linux 系統是以httpd程式放置於**/usr/sbin**中。像ftp, telnet和pop-3，它能由inetd來啟動，但這個服務通常執行在獨立模式下並於開機時由其中一個 *rc* 手稿來執行。httpd 是由系統管理者執行，但它會產生個別的httpd程序來服務客戶端的需求，用ps axf 命令來將它們顯示出來：

```
270  ?  S    0:03 /usr/sbin/httpd -f /etc/httpd/httpd.conf -D SSL
280  ?  SW   0:00  \_ (httpd)
281  ?  SW   0:00  \_ (httpd)
282  ?  SW   0:00  \_ (httpd)
283  ?  SW   0:00  \_ (httpd)
284  ?  SW   0:00  \_ (httpd)
```

在這裡有五個額外的程序，他們的PPID全部是270，即從*rc* 手稿中初始執行的 httpd 之 PID。 像named和sendmail一樣，Apache儲存主要httpd程序的PID在一個檔案中，即/var/run/httpd.pid。這意味著您能用kill `cat /var/run/httpd/pid` 來殺掉這程序。在您改變它的設定檔後，您也能用hangup訊號來重新啟動程序 (kill -HUP `cat /var/run/httpd/pid`)。

Apache的運作被三個設定檔所控制，包括**httpd.conf**, **srm.conf**和**access.conf**，它是繼承NCSA伺服器的傳統。然而，使用在這三個檔案的directives（語法）是共通的。從1.3版之後，Apache的包裝會有一個個別的設定檔**httpd.conf**。這檔案在Red Hat中位於**/etc/httpd/conf**中，在SuSE則位於**/etc/httpd**。

預設的情況下，Apache已經完全的配置在Linux系統中，甚至已經在執行了，除非您在系統安裝時自己選擇不同的方式。您不用改變太多設定檔的內容，然而，在大型的系統中，一台機器可能會扮演多個網域的網頁伺服器，而Apache允許這樣的特色。在這樣的情況下，您需要為每一個虛擬主機的這些檔案建立個別的項目並控制它目錄的存取權限。您也許需要調整幾個參數才能更有效率的運作。

24.14　httpd.conf：設定檔案(Configuration File)

在這幾節中，我們將討論這些參數（Apache稱它們為指令directives）。不是由它們出現在檔案中的順序，而是由它們的功能來分組。

24.14.1　一些基本的設定

Apache使用**ServerRoot**指令來指定設定檔被放置的目錄。請確定檔案被置放在那裡：

```
ServerRoot /etc/httpd
```

Apache首先檢查**httpd.conf**，然後是**srm.conf** 和**access.conf**。即使您將全部的內容移到**httpd.conf**，Apache仍會檢查另外兩個檔案，除非您直接忽略它們：

```
ResourceConfig /dev/null
AccessConfig /dev/null
```

這裡也有分別的指令用來指定httpd 的PID檔案位置和它的存取和錯誤檔案：

```
PidFile /var/run/httpd/pid
ErrorLog /var/log/httpd/error_log
CustomLog /var/log/httpd/access_log common
```

Apache使用埠80。所有在1024以下的埠都被視為有優先權。然而，只有主要的httpd程序需要系統管理者的權限。真正在執行工作的子httpd程序，事實上是以一般使用者的權限來執行。在安全的考量下，確定httpd 所使用的UID和GUID不會被任何使用者用到。Red Hat對兩者都使用nobody：

```
User nobody                    在/etc/passwd中的nobody
Group nobody                   在/etc/group中的nobody
```

網頁伺服器管理員的（負起維護網頁伺服器的人）地址預設為*root@localhost*。當使用者在伺服器上碰到問題時，這地址是使用者經常看到的。因此，網頁伺服器管理員可能不是使用系統管理員帳號，而且主機也可能有它自己的FQDN，您應該設定**ServerAdmin**和**ServerName** 指令給予有意義的值：

```
ServerName www.planets.com
ServerAdmin webmaster@planets.com
```

ServerName 是伺服器傳回給客戶端的名字。舉例來說，即使網頁伺服器在*neptune* 主機上執行 ，您可能要使用者在他們的URL視窗看到名稱是*www.planets.com*，在您設定**ServerName**會傳回一個名稱不同於主機的正式名稱之前，確定別名（這裡是

www）顯示在DNS中以用來存取網頁伺服器（當作一個CNAME記錄）。zone檔案的確以www當作別名(24.3.3)，因此我們能像上面一樣合法的設定ServerName。

24.14.2 載入模組(Loading Modules)

Apache由一些模組組成，很多模組在軟體建立程序時會被引用進來，這些模組提供了基本的網頁伺服器的功能。很多指令是模組特有(module-specific)的，您不能預期一個特別的指令會正常工作，除非對應的模組是有效的。1.3的版本以後，Apache也能在執行時動態的載入模組，因此您能由一個較小的執行程式開始，並且只加入這些您需要的模組。

有兩個指令可以讓這些模組生效。LoadModule 指令載入可執行的模組，其目的檔的副檔名為 .so，您可以找到像這樣的一些行：

```
LoadModule mime_module          /usr/lib/apache/mod_mime.so
LoadModule cgi_module           /usr/lib/apache/mod_cgi.so
LoadModule proxy_module         /usr/lib/apache/libproxy.so
LoadModule auth_module          /usr/lib/apache/mod_auth.so
```

對每一個模組而言，有一個對應的AddModule指令應該也被置放。這個指令載入來源程式來建立可執行的模組。來源檔案名稱的結尾一定是.c：

```
AddModule mod_mime.c
AddModule mod_cgi.c
AddModule mod_proxy.c
AddModule mod_auth.c
```

不需要載入所有的模組；不必要的模組將降低伺服器的效能，請載入剛好是您需要的模組。舉例來說，如果您的網站不用CGI，那麼就不要載入cgi_module模組。

24.14.3 資源定位(Resource Locations)

除了這些設定檔外，Apache需要分開的目錄來儲存伺服器的網頁，即HTML文件。它被稱為文件根(**document root**)目錄。HTML檔案的預設位置是用 DocumentRoot 來設定：

```
DocumentRoot /home/httpd/html
```

這意味著當您用URL *http://www.planets.com/survey.html*存取一個網頁時，您實際上是存取在檔案系統中的/home/httpd/html/survey.html檔。Apache的安全功能不允許您在此階層架構往上移和存取一個檔案，例如在/usr/sbin內的檔案（匿名ftp也

不允許這樣)。

CGI程式也造成大部分網頁上的入侵(break-ins)，他們需要被放置在與**DocumentRoot**不同的目錄，這一定是**cgi-bin**目錄，而且可以用**ScriptAlias**來設定：

```
ScriptAlias /cgi-bin/ /home/httpd/cgi-bin/
```

ScriptAlias只能設定一個有包含CGI程式目錄的別名。它不能被用來定義其他目錄的別名。Icon（圖像）也有它們自己的目錄，但是這次別名的設定就必須使用**Alias**：

```
Alias /icons/ /home/httpd/icons/
```

此**Alias**敘述用來設定任何目錄的路徑是很有用的（除了含有CGI程式目錄的以外)。

一部主機可能服務多個使用者，這些使用者需要分開的目錄來放置他們的網頁，這是由**UserDir**來做決定，預設是**public_html**：

```
UserDir public_html
```

這個目錄會被建立在使用者的家目錄中。因此，當Apache遇到了一個 *www.planets.com/~juliet* 的URL需求，它會提供來自目錄**/home/juliet/public_html**中的預設網頁（假設家目錄是放在**/home**中)。Apache也允許選擇性的存取“UserDir”目錄：

```
UserDir disable                          關閉所有的使用者
UserDir enable henry romeo juliet        除了這裡的三個人
```

Apache允許使用者來覆寫它所設定的安全限制。它指定檔案的名稱而其中包含使用者定義(user-defined)的限制：

```
AccessFileName .htaccess
```

這意味著每個使用者能在他想個別去設定存取權限的目錄下放置一個**.htaccess** 檔案。這個檔案能包含很多使用在設定檔中的指令。但它們只能應用在那個目錄和其所有子目錄。

24.14.4　檔案類型

什麼是預設網頁？當URL用一個目錄名稱作為結束，Apache使用指令**DirectoryIndex** 在那個目錄中尋找一個或更多的檔案。預設是設定為**index.html**,

但如果在網站(site)中包含數個使用者，通常會發現合理的一組預設設定：

```
DirectoryIndex default.htm index.htm index.html index.cgi welcome.html
```

為了能滿足Windows 3.1的使用者，此設定也接受三個字元的副檔名。如果在目錄中沒有檔案被找到，Apache會顯示該目錄的一個列表。還有一些極好的選項來控制顯示內容和讓您加入屬於自己的裝飾，但是我們將不會討論它們，除了注意最好讓這個設定保持開啟：

```
FancyIndexing on
```

Apache定義新的編碼檔案形式，而且它自己處理它們。舉例來說，檔案壓縮用compress和gzip能由瀏覽器本身即時被解壓縮而不需要執行uncompress和gunzip程式：

```
AddEncoding x-compress Z
AddEncoding x-gzip gz
```

Mosaic和Netscape也能自動地做解壓縮。因此，如果從您瀏覽器上點選一個目錄列表中的.gz檔案，您將立刻看到被解壓縮過的檔案。

先前討論的內容可以充分的被應用在單一網站主機所提供的網頁服務上。大型系統的環境下通常提供數個網站使用虛擬主機的概念。在這樣的例子中，每一個網站對於這些目錄將會存放在不同的位置中，而通常是它們自己的家目錄下面。稍後，我們將檢視虛擬主機的設定。

24.14.5 調整(Tuning)伺服器

光芒畢露的內容將吸引很多人造訪此網站。對許多網站來說，在一天之中有成千上百的造訪記錄是很平常的。Apache預設的設定並不允許這樣，因為要服務每一個客戶端需求，它必須得產生單獨的httpd子服務程式，而不是等待客戶端要求時才產生這樣的程序，一些程序必須保持準備好的狀況來減少啟動時間。並且由下面的這些設定來控制：

```
MinSpareServers 5
MaxSpareServers 20
StartServers 8
```

任何時間中，五個程序將被閒置來等待服務一個需求。在一個忙錄的網站中，您必須持續的調整這個伺服器程序數量的最大值。系統在啟動期間已經開啟了八個程序。

除了上面的敘述之外，您也需要設定一個限定hppd能服務客戶端程式的最大

值，預設是到150：

```
MaxClients 150
```

HTTP 在TCP傳輸協定(23.1)上執行，這意味著資料能被傳送前連線必須要被建立。當使用者點選一個連結(link)，一個新連線就被建立，因此增加了檔案的傳送時間。HTTP 1.1建議**keepalive** 功能，保持不斷的連線。這個不斷的連線被這些指令控制：

```
KeepAlive On                                          這個設定必須是on
KeepAliveTimeout 15
```

第二個設定意味著目前的連線在下一個需求來到時，將被保留15秒。在任何的狀況下，所有需求及回應確認必須在五分鐘內完成：

```
Timeout 300
```

網路中的擁塞狀況可以藉由在記錄檔內使用IP地址而非主機名稱做查詢來減緩：

```
HostnameLookups Off
```

這個設定意味著一個進入的連線將不呼叫DNS來做查詢，因此改善了伺服器回應的時間。

24.15　虛擬主機

所有在全球資訊網上的網站不一定是個別的主機，Apache在獨立的機器上提供虛擬主機(**virtual hosts**)，它經常被發現在一個容納超過上千個網站的主機上。主機可能有多個IP地址，且每一個IP地址表示一個網站。此外，它也可以是多個網域名稱共用一個IP地址。您能使用**<VirtualHost>** 指令為每一個虛擬主機來設定許多伺服器參數（這次請注意它使用<>符號）。

Velvet現在決定使用由Planets同意在它伺服器上提供20 MB的空間，並用來提供它們的網頁內容。我們假設網站*www.velvet.com* 有它自己獨立的IP地址，這個地址必須是出現在DNS中的A記錄，並指向*www.planets.com* 。一組的指令設定能被分別定義在**<VirtualHost>** 和 **</VirtualHost>** 標籤之間：

```
<VirtualHost www.velvet.com>
   ServerName www.velvet.com
   ServerAdmin webmaster@velvet.com
   DocumentRoot /home/velvet/www
   TransferLog /home/velvet/logs/access-log
   ScriptAlias /cgi-bin/ /home/velvet/www/cgi-bin/
</VirtualHost>
```

您能在第一個指令中使用IP地址來代替**www.velvet.com**。當伺服器接收到來自這網站的需求時，**ServerName**會把**www.planets.com**改變成**www.velvet.com**。這個網站也有它自己的**DocumentRoot** 和**cgi-bin**目錄。假設，Velvet有一個使用者帳號velvet在Planets的**/etc/passwd**中。除了可以有一個家目錄為基礎的架構(directory-base)安排之外，Planets也能分散管理，讓Velvet管理它們自己的網站。這通常意味著Velvet可以允許使用telnet及ftp來連結它的虛擬網域。

對於每一個虛擬主機而言，它必須要有像上面那樣的個別段落。除了一些像**ServerRoot**, **ServerType**和用來調整Apache的參數外，幾乎**httpd.conf**中的任何指令都能被用在這裡。

24.16 目錄存取控制

Apache經常藉由**access.conf**的設定來限制存取包含文件和CGI程式的目錄。此外，它允許使用者以個別目錄的方式覆寫泛系統(systemwide)的限制。存取限制被設定在**<Directory>**和**</Directory>**標籤之間，一個目錄中含有HTML檔案的典型例子，看起來像這樣：

```
<Directory /home/httpd/html>
  Options Indexes FollowSymLinks
  AllowOverride None                          不能使用.htaccess
  order deny,allow
  deny from all
  allow from hillinfo.com
</Directory>
```

Options敘述指定這個目錄和它全部的子目錄被允許的選項。如果它不能發現檔案**index.html**(**Indexes**)，這個伺服器會產生一個目錄中所有檔案的列表。它也使用一個符號連結並依照同樣的方式來處理一個普通的檔案 (**FollowSymLinks**)。**AllowOverride**可決定是否一個使用者被允許來覆寫這些選項。在這裡，沒有使用者可以這樣做(**None**)，但**cgi-bin**目錄允許每一個使用者來覆寫以下的這些選項：

```
<Directory /home/httpd/cgi-bin>
  AllowOverride All
  Options ExecCGI
</Directory>
```

在這裡的設定**AllowOverride All**意味著每一個使用者可以放置同樣的指令到他們自己的本地檔案**.htaccess**中，來取消這些限制。 當伺服器從任一個目錄中擷取一個檔案時，它首先找尋這個檔案。如果可以找到，且覆寫的這個功能有被設定，伺服器就會應用這檔案中的敘述。通常讓使用者設定他們自己的目錄是很合理的，因為當他們改變**.htaccess**檔案時，伺服器不需要重新開機。

注意到cgi-bin目錄使用一個特別的選項設定(ExecCGI)。任何目錄必需有ExecCGI設定才能被用來放置CGI程式。CGI程式在全球資訊網上面是一個很大的安全威脅，所以有必要明白的指出哪一個目錄有執行CGI程式的權限，而不是預設。然而，在這裡的敘述是多餘的，因為已被隱含在ScriptAlias敘述而且指向同樣的目錄。

您能設定一個目錄樹的存取權限，然後覆寫一個子目錄的權限：

```
Alias /web/docs /home/html/docs
<Directory /web/docs>
        Options Indexes FollowSymLinks
        AllowOverride None
</Directory>
<Directory /web/docs/dl>
        AllowOverride All
</Directory>
```

伺服器會查看在/web/doc/dl目錄中的.htaccess檔，但並不會查看在/web/docs/中任何其他的子目錄。這會改善伺服器的效能。

order，allow和deny敘述用來定義在這目錄中可被允許存取文件或是執行程式的主機，order決定存取規則被實行時的順序。這裡，deny的規則首先被應用然後才是allow的規則。當只允許存取少量主機時，通常先拒絕所有使用者的權限，然後再允許一些使用者。此處，只有*hillinfo.com* 被允許存取在目錄中的檔案。要允許全部的使用者，只要使用allow from all。

除了可以限制主機外，Apache也需要在使用者存取目錄時做認證。此時使用者必須提供使用者名稱和密碼來使用某些服務；您可能已經在全球資訊網看過這些了。Apache使用的htpasswd程式就具有使用者認證架構。它還有一些進階的功能，但是在本書中我們無法提到。


Tip

如果您不要一個特定目錄的伺服器選項被覆寫，只要在那個目錄中用AllowOverride None即可。因為Apache不需要在這目錄中的任何子目錄下搜尋.htaccess，所以能明顯提升伺服器的效能和增加它的安全性。

摘　要

網際網路使用網域名稱伺服器(DNS)來解析FQDN成為IP地址，反之亦然。這些訊息被保留在一組名稱伺服器上。一個網域被區分成區域(zone)，主要名稱伺服器是該區域官方的伺服器。次要名稱伺服器藉由區域傳送從主要伺服器下載它的訊

息。

查詢是由解析系統來完成，它是一組的程式館常式被建立到TCP/IP 的應用程式。它使用/etc/resolv.conf檔案。解析的方式是依據一個階層式的樣式，當區域名稱伺服器無法提供回答時，根伺服器就會被查詢。

BIND 使用named的服務程式並且是DNS最常用的實作程式。hints檔案包含了根名稱伺服器的IP地址和FQDN，而且被維護在每一個名稱伺服器的快取中。zone檔案包含了IP地址和主機名稱的對照表。反向查詢和localhost檔案執行反向解析，即把IP地址轉換成FQDN，或127.0.0.1成為localhost。DNS也指定一些伺服器來接收網域的郵件。

資料庫檔案使用名稱伺服器(NS)的記錄來指定名稱伺服器的IP地址，郵件交換器(MX)記錄用來提供指標指向郵件伺服器。地址(A)和指標(PTR)記錄對應地址到FQDN中，反之亦然。正式名稱(CNAME)記錄提供別名(aliases)。所有檔案的位置和名稱伺服器的類別被指定在/etc/named.conf檔。ndc 能啟動和停止named而nslookup 可以用來測試設定。

sendmail 是通用的傳輸代理程式，在Linux系統中，procmail 處理傳送工作。郵件可從遠端的網站用郵局傳輸協定(POP)取得，而通常是用撥號線路。

集中的hub（郵件伺服器）通常代表所有的主機寄送和接收郵件。它通常在郵件地址中被設定成隱藏主機名稱。hub可能轉送郵件到獨立的主機或提供可用NFS掛載的檔案系統。它也能扮演一個POP/IMAP伺服器。

sendmail 可執行成一個服務程式(-bd)在埠25中來接收郵件，也可成為SMTP的客戶端(-q)來清理郵件佇列。這個佇列被維護在/var/spool/mqueue中。sendmail 從/etc/sendmail.cf取得指令和使用/etc/aliases來執行郵件的轉送。別名(aliases) 被用來重新轉向郵件到一個或多個使用者、郵寄名單或一個檔案。

sendmail.cf 使用巨集(D)來定義一個聰明的中繼主機(DS)和執行偽裝(DM)。 小寫字母的巨集在內部被定義，級別(class)被用來定義sendmail 接收郵件的網域(Cw)。使用選項(O)，在sendmail 通知失敗前，您能設定讓它嘗試傳送一個訊息多少次數(O Timeout.queuereturn)。

Linux系統使用fetchmail 和.fetchmailrc來從POP/IMAP伺服器上接收郵件。您能決定是否保留郵件訊息(-k) 或全部取回(-a)。fetchmail 被用來在單一信箱中取得全部網域的郵件，然後將它分配到個別的信箱中。

Apache是網際網路上使用最廣的網頁伺服器，它執行httpd服務程式在埠80並使用HTTP傳輸協定。httpd 產生另一個無特權模式的httpd 程序來處理客戶端需求。它的配置被設定在httpd.conf檔案中，但很多系統也使用srm.conf 和access.conf。Apache有足夠的彈性來載入在執行時需要用到的模組（LoadModule

和 AddModule)。

建置檔案被保留在伺服器的根目錄(ServerRoot)中。HTML檔案位置被DocumentRoot所設定。CGI程式因安全理由被保留在一個個別的目錄中(ScriptAlias)。使用者能保留他們自己的檔案在個別的目錄中(UserDir)。

當URL不是以一個檔案結束，伺服器會檢查index.html檔案，除非有不同的指定 (DirectoryIndex)。Apache和最新的瀏覽器能即時的解壓縮.Z和.gz的檔案(AddEncoding)，這是因為在HTTP 1.1中的KeepAlive，所以它可支援不斷的連線直到KeepAliveTimeout設定的區間。

Apache在獨立的伺服器中，支援虛擬主機(<VirtualHost>) 讓多個網域可以被放在的單一的主機上，可以使用單一的IP地址或是不同的IP地址。每一個虛擬網域能有它自己的文件位置設定和取存權限。

每一個目錄(<Directory>)也能有它自己存取的限制，Options決定是否CGI程式可從一個目錄中來執行(ExecCGI)或一個目錄中的列表可被允許(Indexes)。一個使用者可以或是不可以被允許來覆寫這些選項(AllowOverride)。一個或更多網站可以被允許(allow from)或拒絕(deny from)存取一個目錄。

自我測試

24.1　次要名稱伺服器扮演什麼樣的角色，它如何獲取它想要的資料？

24.2　ftp 如何取得名稱伺服器的IP地址？

24.3　DNS如何指定網域中的郵伺服器來接收郵件？

24.4　您如何替相同的主機指定數個名字？

24.5　誰負責來做傳送？請舉出一些重要的傳送程式。

24.6　要送出的信件存在哪裡，您如何檢視這個佇列？

24.7　您如何通知sendmail，讓它來接收數百個網域的郵件？

24.8　您如何檢查在系統中sendmail或POP是否正在運作？

24.9　您在/etc/aliases檔案中定義轉送，但它無法運作。您可能犯了什麼錯誤？

24.10　在系統中有多少個httpd 程序要被保持在準備好的狀態？

24.11　您如何設定網頁伺服器上HTML文件的根目錄？

練　習

24.1　一個網域和一個區域之間有什麼不同？DNS如何處理它們？

24.2　什麼是只能快取名稱伺服器？

24.3　根名稱伺服器如何幫助解析動作的流程？

24.4 何時您需要轉換一個IP地址到一個FQDN？

24.5 次要名稱伺服器如何知道主要名稱伺服器的內容項目已被更改？

24.6 寄到*user@hillinfo.com*的郵件是由主機*mail.hillinfo.com*處理，但是那一台主機會接受寄到*user@www.hillinfo.com*的郵件？

24.7 如果您除了能用telnet edison存取一個網站之外，還有telnet edison.hillinfo .com，它的理由是什麼？假設在**/etc/hosts**沒有任何設定項目。

24.8 您如何建立sendmail 能接收郵件和每半小時清理佇列一次？

24.9 如果SMTP直接傳送郵件到另一個SMTP，為什麼郵件有時候要花數天才能到達？

24.10 一個使用者希望當送出郵件時能隱藏她的主機名稱並且希望她的ISP處理她的郵件，那麼她應該怎麼做？

24.11 您希望郵件在一小時內無法傳送時就得到通知。您需要改變什麼設定？

24.12 如果客戶端使用Netscape Messenger 時需要執行sendmail 來送信到hub嗎？

24.13 在一個接收端郵件伺服器上，**/etc/passwd**扮演什麼樣的角色？

24.14 您如何設定Apache來讓melinda在她的家目錄中存取網頁？URL如何做目錄的存取？

24.15 當URL不包含一個檔名時，您如何讓Apache伺服器用**default.html**檔？

24.16 您如何建置您的伺服器來保持不斷的連線？

24.17 **ScriptAlias**敘述的意義為何？

24.18 您如何讓*velvet.com*網域無法存取**cgi-bin**目錄？

24.19 您如何啟動在一個目錄中可執行CGI？