第二十三章

TCP/IP網路管理

在本書最後的兩個章中，我們要再一次看TCP/IP，但這次是從網路管理者的觀點來看。單一電腦的時代已經過去了，今天使用者甚至可將應用程式執行遍及在網路各處。網路管理成為不可或缺的部分和管理動作的專業組件。在大型系統的建置環境下，網路管理者通常不同於系統管理者。

一些TCP/IP的理論已提供在本章的開頭。我們將學習設定一個簡單的TCP/IP網路和建置它來存取另一個網路的主機。我們將用網際網路服務程式(daemon)設定基本的網際網路伺服器，以及用撥接線來連結網際網路。本章也會介紹兩個新加入TCP/IP的家族成員，即PPP和NFS。大量被使用在網路上的服務，像DNS，SMTP email 和HTTP會在最後章介紹。

目　標

- 了解一個TCP/IP網路的功能和IP地址系統。(23.1)
- 安裝網路介面卡和用ifconfig來配置它。（23.2 和 23.3）
- 用ping 來排除網路的問題。(23.4)
- 學習TCP/IP如何處理繞送 (routing)和用route來建立路由。(23.5)
- 用netstat顯示網路的統計。(23.6)
- 學習inetd在處理基本的TCP/IP服務程式，像是ftp和telnet時的角色。(23.7)
- 了解點對點傳輸協定(point-to-point protocols)的特色。(23.8)
- 用dip和chat來連結您的Linux電腦到網際網路上。(23.9)
- 了解使用在PPP上兩個重要的驗證傳輸協定，即PAP和CHAP。(23.10)
- 學習網路檔案系統的功能和配置。(23.11)

23.1　TCP/IP和地址系統

TCP/IP是一個階層式的產品，　不同的邏輯層處理不同的功能。它包括兩個傳輸協定來處理許多重要的工作：

- 傳輸控制協定(**Transmission Control Protocol** TCP)—— TCP是負責保證可靠性的傳輸和擁有健全的錯誤檢測及復元的能力。它劃分接收來自應用層的資料成為區段（或是封包，packets），並將每一個區段加上封包總合檢查碼(checksum)及序號做成包裝，以確保在另一端正確的順序重新組合。此封包也包含了來源及目的之埠編號(11.1.4)。如果它不能以完整的格式被另一端接收，它會再重送一個區段。

  TCP 也是連接導向(connection-oriented)的傳輸協定，它首先用一個交互(handshaking)訊號與對等的另一端(peer)溝通，確定遠端系統已準備好要來交換資料。當交互訊號成功時，一個連結被建立起來，然後TCP就能開始進行資料交換。大部分的應用都是使用TCP，但有些應用（像RealVideo）不能負荷這樣的交互動作，它使用另一種傳輸協定，像UDP來替代。UDP不能保證其可靠性，也不是連接導向傳輸。
- 網際網路傳輸協定(**Internet Protocol**)(IP)——一個封包經過TCP層移動到下一層的IP層。IP決定一個封包必須到哪裡和它來自何處。它提供了來源和目的IP地址給一個TCP區段，也扮演著一個不可或缺的角色，它繞送（routing, directing，即指引）封包到達適當的目的地。如果一個封包屬於在同一個網路中的一台主機，IP會直接送它到該網路中。如果它屬於一個不同的網路中的主機，IP指引封包到最近的路由器，「期望」這個路由器確實是最接近目的地的一個。

注意到兩個扮演不同角色的TCP和IP。TCP指定埠的編號，即需要被連結的應用程式，IP要知道一個封包要去那一台機器及它從何處來。

在使用TCP/IP網路前，您應該知道一些有關於在網路上機器被定址的方法。每一個機器有一張合適的網路介面卡，它藉由線材連結到另一部有同樣介面卡的機器上。所有主機的通訊通常只發生在這些網路介面卡之間。

每一個乙太網路卡有48個位元的實體地址，由硬體廠商把固定編碼(hard-coded)寫到電路板上，這個地址被稱為**MAC address**（媒體存取控制地址：Media Access Control）或乙太地址。它包含六個由冒號分隔的十六進位數字，它的形式看起來像這樣：

```
00:00:E8:2E:47:0C
```

除了這個地址，TCP/IP也了解一個IP地址，即一個邏輯的軟體地址（網際網路地址），而您已經把它用在一些TCP/IP的工具上。即使每一個主機名稱會被轉換到與它對應的IP地址，TCP/IP最後必須轉換此IP地址到主機對應的MAC地址。

這裡也有一些管理的規則來分配這些IP地址。考慮到下面32位元的地址，為了方便起見，把它分割成四個八位元組(octet)：

11000000 10101000 00000001 10100000

這個二進位形式可轉換為十進制的192.168.1.160。這個地址含有兩個部分，即一個網路地址和一個主機地址。每一個八位元組的最大數是255，網路地址通常是在同一個網路上的所有主機和使用一個到三個從左邊算起的八位元組，剩下的八位元組則被主機所使用。在這裡的192.168.1是網路地址，依照慣例，我們用零填滿主機的部分，因此它實際上應該讀作192.168.1.0。

有多少八位元組的數字要進入到網路部分是被子網路遮罩**(subnet mask)**所決定。這再一次是四個連續的八位元組，網路部分的個別位元的地址被設定成1。上面的網路應該是以255.255.255.0當作子網路遮罩。有多少八位元組數字可設定成255當遮罩是依據特定的規則來決定。TCP/IP使用這個遮罩來決定一個封包是否屬於目前的網路。稍後，我們會討論一些有關遮罩的事。

在網際網路上，地址可分屬於三個級別**(class)**中的一個。有多少八位元組數字保留給網路地址和第一個八位元組的值可決定網路屬於哪一個Class。三個Class的表示如表23.1。

您可以藉由觀看位於第一個八位元組，能簡單的說出網路是哪一種類型。舉例來說，148.27.3.12 是一個Class B的地址，192.142.3.67是一個Class C的地址。148.27.0.0是一個網路地址，192.142.3.0是另外一個網路地址。

當您有一個Class B的網路，您能有65,534個主機在此 Class 中，但您可能不需要那麼多的數量。如果您可以將網路切割子網(subnet)，藉由用完前三個八位元組，這樣就能有126個主機，就能降低您的需求。TCP/IP也允許從一個八進位組裡通常用來組成主機地址的元件中「借用」位元組。因此，如果從上面的Class B網路(140.27)的第三個八位元組借出前兩個字元，則子網路遮罩要變為255.255.192.0 (128 + 64 = 192)。

當分配IP地址時，您必須在您的頭腦中記住這些特點：

- TCP/IP需要為每一個機器保留一個分開的地址來廣播(broadcasting)訊息給所有的主機。 廣播地址是設定主機部分的IP地址成為255所得到的，這意味著我們的網路192.168.1.0 有192.168.1.255的廣播地址和子網路遮罩255.255.255.0。
- TCP/IP將**localhost**看做是沒有介面卡的一部機器，它需要一個個別的地址，即一個回授地址**(loopback address)**，這個地址是127.0.0.1，您能在一個獨立的主機上用它來執行網路的服務。
- 在網際網路上使用的地址不應該被用在區域網路上，即使區域網路沒有被連接到網際網路。每一個Class中的一個地址區塊，已被保留用在區域的網際網路中，它們顯示在表23.1中的最後一個欄位。

有了這個經歷，您應該能設定一個網路地址。舉例來說，您能選用Class C的地址192.168.5.0當作您的網路地址。這樣的話，您可以有主機地址設定在192.168.5.1和192.168.5.254之間，網路地址和廣播地址分別是192.168.5.0和192.168.5.255，它們無法再分配給其他的主機。

表 23.1 **網路的級別**(Class)**和保留地址**

網路級別	第一個八位元組的值	子網路遮罩	企業內部網路的網路地址
A	1-126	255.0.0.0	10.0.0.0 - 10.255.255.255
B	128-191	255.255.0.0	172.16.0.0 - 172.31.255.255
C	192-223	255.255.255.0	192.168.0.0 - 192.168.255.255


Tip

子網路遮罩和廣播地址各自的八位元組的值最多可到達255，如果您知道一個的值，您能簡單的計算另一個的值。

23.2 設定網路介面卡

您需要在每一部連接到網路的機器上放一片乙太網路卡。UNIX系統識別網路介面卡，就像開機的時候辨識硬碟和CD-ROM裝置。大部分的作業系統在安裝系統的同時，會一起安裝需要的介面卡和必要的驅動程式，但廠商也提供特別的工具在安裝後可再增加一片卡片。

當您設定您的網路介面卡時，系統可能自動地設定兩個硬體參數，如果它不能這樣做時，您就必須提供它們。通常，如果您的UNIX機器要在網路上正確的運作，這些參數必須被設定：

- I/O位址（一個硬體參數）。
- 中斷向量(interrupt vector)（一個硬體參數）。
- IP位址。
- 子網路遮罩。
- 廣播地址。
- 通訊閘道器地址。
- 主機名稱。
- 網域名稱。

I/O位址是一個十六進位 數字（通常是 0x300）。IRQ是一個整數（通常是2或9）。如果您的機器能自動的設定這些參數值，您就不用擔心了。不然，在您開始使用之前，您自己必須在卡片上設定它們。對於PC，在卡片插入機器後，您必須執

行一個廠商提供的一個小型的DOS程式。您必須確認系統設備使用的I/O地址和IRQ（像是鍵盤、硬碟和串列埠）不會跟您的設定衝突。

在這兩個參數被提供之後，機器必須在開機的時候正確地辨識卡片。很多系統提供支援像dmesg命令可簡易的列出開機訊息。如果介面卡被正確的安裝，您應該會看到像是這樣的dmesg 輸出：

```
ne2k-pci.c:vpre-1.00e 5/27/99 D. Becker/P. Gortmaker
ne2k-pci.c: PCI NE2000 clone ‘ RealTek RTL-8029 ’ at I/O 0x6200, IRQ 10.
eth0: RealTek RTL-8029 found at 0x6200, IRQ 10, 00:80:C8:02:24:78.
```

此處系統認出一個NE2000 PCI的網路卡，顯示了I/O地址設定**0x6200**和IRQ設為**10**。卡片的MAC地址顯示在最後。在系統中這個介面卡有被辨別出一個**eth0**的名字，很多網路工具在他們的輸出中明確的顯示出這個名稱。如果機器有另外一個介面，它可能會被取名為**eth1**。**eth** 是Linux用來命名介面卡而使用的開頭字串，所以這應該是在一個Linux系統中。其他的UNIX系統的名字樣式從製造廠商的卡片或晶片設定來取得。您可能會看到像**ie0**, **en0**, **le0**等等的名字。

這時引導我們進行第二個部分的配置，即軟體部分。您現在必須定義幾組的四個八位元組，包括IP地址，子網路遮罩，廣播地址和通訊閘道器地址。這些資訊被ifconfig和route命令使用在啟動的時候。機器也使用hostname 命令，從您提供的輸入來設定您的機器之FQDN。當您的網路沒有子網路時，也常由您提供IP的地址讓系統自動的決定子網路遮罩和廣播地址。

只有當您的機器連接到一個通訊閘道器和路由器時，您才必須提供通訊閘道器地址。TCP/IP沒有把這兩個由傳統網路技術做成的設備區分的很清楚。一個通訊閘道有時可能是唯一的方法能讓您的主機連接到網際網路或是另一個網路上，它通常是專用的設備，但一個機器使用兩片介面卡也能扮演通訊閘道器的角色，亦即每一片對應一個網路。事實上，如果一個機器被連接到四個網路，它必須有四張這樣的卡片。

23.3 ifconfig：設定網路介面

因為TCP/IP獨立於網路硬體，IP地址不會被內建到核心中，而是存在於網路軟體中。您必須使用ifconfig命令來設定您介面卡的IP地址，這命令的使用如下：

```
ifconfig eth0 192.168.0.3
```

命令的語法需要介面卡的名字（在這是**eth0**）和IP地址。這裡不但設定一個IP地址到介面卡中，而且也將它啟動。在非Linux的機器中，這個命令能適切的準備您的機器讓它可使用在網路的環境中。當ifconfig沒有改變系統的任何檔案時，它就便

無法永久的設定介面卡。所以每一次系統開機時，這命令必須從*rc*手稿中執行。

ifconfig可以選擇使用子網路遮罩和廣播地址當參數。如果網路不是子網路的話，就不需要它們，所以我們在此將不提供這兩個參數。如果您的網路遵照class規則（A, B或C），ifconfig會自動地計算子網路遮罩和廣播地址。對分割成子網路的網路而言，網路遮罩(netmask)和廣播(broadcast)參數必須被分別地提供，以下是我們如何使用ifconfig 來設定在一個子網路中的一個主機：

```
ifconfig le0 147.35.3.45 netmask 255.255.192.0 broadcast 147.35.63.255
```

我們這次使用一個不同的介面(**le0**)，廣播地址也已被指定，雖然它可從子網路遮罩中取得（主機地址的部分其對應的位元中兩者的最大值都是**255**）。很多舊的BSD 4.2系統使用**0**代替**1**來設定廣播位元，將它明白標示出來總是安全的作法。

要了解更多有關於您系統上所有的介面卡，依照您使用的系統，您可以簡單的使用ifconfig（像在Linux中）或是提供**-a**的選項。以下是從執行System V的機器上取得它的輸出：

```
# ifconfig -a
le0: flags=4043<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 147.35.3.45 netmask ffffff00 broadcast 192.168.0.255
      perf. params: recv size: 4096; send size: 8192; full-size frames: 1
      ether 00:20:18:62:47:e0
lo0: flags=4049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
      inet 127.0.0.1 netmask ff000000
      perf. params: recv size: 57344; send size: 57344; full-size frames: 1
```

這個輸出會因系統不同而有所變化，但您能了解所有從介面卡輸出的固定參數。ifconfig在這不僅告訴您機器的IP地址，而且也顯示它的狀態。系統是**UP**，支援**BROADCAST**和目前是**RUNNING**。注意到，它也顯示您介面卡的MAC地址。

看起來似乎有另一個介面卡**lo0**；這是每一部主機必須要有的回授介面，即使ifconfig沒有明確地將它的IP地址設定為**127.0.0.1**，它確實在系統裝機時就已經被設定了。

ifconfig也能顯示一個特定名稱的介面卡，Linux甚至會提供介面卡硬體參數的顯示（IRQ和基本的地址）：

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:20:18:62:47:E0
          inet addr:192.168.0.1  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0
```

```
          TX packets:9 errors:0 dropped:0 overruns:0
          Interrupt:9 Base address:0x300
```

ifconfig不僅僅設定或顯示介面卡的屬性，它被用來讓介面卡啟動或關閉。有時候，當IP地址必須被改變的時候就需要這樣做，命令的使用方法如下：

```
ifconfig eth0 down                          關閉介面卡
ifconfig eth0 up                            開啟介面卡
ifconfig eth0 192.168.0.5 up                介面卡的設定和啟動
```

ifconfig保證您在同一個網路上能連接到所有的主機（也提供他們正確的設定），但您仍不能連接到另一個網路的主機。為了處理這樣的狀況，您的系統必須把它的路徑(route)提供給網路，路由的理論會很快的被討論。

Linux 注意事項

如果您看一下*rc*的手稿（或是任何在開機時執行的手稿），它含有ifconfig敘述，您將常發現它配合使用的是變數參數而不是明確的值。這些手稿一般會被設計成通用的格式，所有的變數被定義在另一個檔案中。變數是由先執行手稿中的.或是source命令而被帶入手稿中(17.9.1)。這是一個合理的作法被大部分Linux的平台所遵循。

23.4 ping：檢查網路

一旦網路介面卡被安裝完成，您必須送個封包到一個已在網路上工作的機器。幾乎大部分的使用者用ping命令來排除網路的問題。這命令送出56個位元組的封包到一個遠端的目的地，它會回答發送端已收到此封包：

```
# ping 192.168.0.4
PING 192.168.0.4 (192.168.0.4): 56 data bytes
64 bytes from 192.168.0.4: icmp_seq=0 ttl=255 time=1.442 ms
64 bytes from 192.168.0.4: icmp_seq=1 ttl=255 time=0.735 ms
64 bytes from 192.168.0.4: icmp_seq=2 ttl=255 time=0.708 ms
64 bytes from 192.168.0.4: icmp_seq=3 ttl=255 time=0.711 ms
64 bytes from 192.168.0.4: icmp_seq=4 ttl=255 time=0.744 ms
[Ctrl-c]                                    中斷顯示
-- 192.168.0.4 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.708/0.868/1.442 ms
```

每一行顯示的「來回」時間是由一個特別的(ICMP)封包到達它的目的地和送回來的時間。此處，我們看到一個良好連接(well-connected) 的區域網路，有一個好的回應時間和無「封包遺失」。 在廣域網路中，來回時間大約是幾個毫秒左右，但仍在可接受的範圍內。如果封包隨機的到達，也就可能有一些封包的遺失。然而，當它在

網路上遺失時，TCP/IP是被設計成會重送資料，所以您不需要擔心封包的遺失，或是資料的遺失。

當主機是「關閉或無法到達」 時，在您按下中斷鍵時，您將看到這個訊息：

```
-- 192.168.0.3 ping statistics ---
50 packets transmitted, 0 packets received, 100% packet loss
```


Note

“Pinging”一個主機不需要任何的服務程序執行在另一端，然而，一個成功的ping輸出不一定意味著那些服務是被執行的。舉例來說，如果inetd沒有在遠端的機器上被啟動，即使ping報告成功了，ftp和telnet也將不會運作。


Tip

一個失敗的ping不代表連接上有問題，遠端的主機可能被關閉或是暫時從網路上移走了。去試著ping其他的主機看看。

23.5 路由

一個封包需要藉由一個路由器來處理，如果它的目的地主機是在另一個網路上。每一個在網路上的主機會在核心維護一個路由表(routing table)，這個表至少包含了三個欄位(field)—介面卡的名稱，目的地位址和路由的IP位址。當IP層接收到一個封包時，它從一個目的地位址中，使用子網路遮罩來取出網路部分的地址，然後用這個地址和這個表中的全部項目作比較（注意到一個IP封包不包括這個遮罩)。

如果IP發現這個地址在路由表中，它會經由表中設定的通訊閘道器繞送這個封包至指定的位置。如果不在路由表，接著IP必須送這個封包到預設路由(**default route**)中，它也是在表中有被指定到的項目。IP以這樣的方式，只有指定它到網路的路徑，而不是它到主機本身的路徑。這是IP其中一項最重要的特性。

即使事實上並沒有存在一個路由器，每一個網路都有一個最小的路由表。當到達一個主機的路徑是固定的時候，因為是小的網路，所以網路管理者會建立一個靜態路由(**static routing**)表。

然而，網際網路使用動態路由(**dynamic routing**)，因為在表中通常儲存到一個主機的數個路徑上。此處，路由服務程式（routing daemons，像routed）與另一個機器上的路由服務程式相互通訊來建立動態路由表。如果網路拓撲(topology)改變或部分的網路不運作時，路徑可以用動態方式被調整。路由傳輸協定路也可以決定到一個主機最有效率的路徑，他們讓IP看起來「很聰明」。

23.5.1 route：建立一個靜態路由表

ifconfig命令建立一個極小的路由表，對一群在相同網路上的主機已經夠好了

(Linux除外)。要轉送封包至另一個網路上的主機，通訊閘道器的路由訊息必須加到路由表中。如果沒有使用動態路由，系統管理者必須手動建立一個靜態路由表，可以用route命令來做到。

要示範如何建立一個靜態路由表，讓我們考慮一個Class C的網路**192.168.0.0**，它有一個*sunny*的主機地址**192.168.0.10**。這網路有兩個通訊閘道，包括*michael* (**192.168.0.1**)存取網際網路和*fredo* (**192.168.0.20**)連結至另一個網路**172.16.1.0**。我們必須在*sunny*主機上安裝經過兩個通訊閘道路徑的路由表。我們假設在*michael*和*fredo*上的網路介面卡已用ifconfig設定完畢。

我們在*sunny*主機上使用route 命令，首先安裝到非網際網路路徑的通訊閘道(在*fredo*)：

```
route add 172.16.1.0 192.168.0.20
```

我們新增一個路徑(**add**) **172.16.1.0**到路由表和指定要被連接並轉送封包的通訊閘道器是**192.168.0.20** (這是fredo的地址)。

因為*michael*扮演了到網際網路的通訊閘道，它明顯地可以處理更多的路徑(非必要的流量)，因此應該被設定成預設的通訊閘道。這麼做是用**default**關鍵字取代一個特定的路徑：

```
route add default 192.168.0.1
```

您一旦用這方法來設定預設路由，所有的封包不屬於區域網路和**172.16.1.0**會被轉送到這個通訊閘道(**192.168.0.1**)，您可以無礙的存取網路。當然，假設這通訊閘道的機器已被安裝成一個防火牆(firewall)來支援IP轉送或是一個代理伺服器(不包括在本書中)，來允許這主機轉送封包到此機器上。

要移除一個路徑，使用**delete**的關鍵字，上面建立的路徑可以簡單的用關鍵字**delete**取代 **add**來移除：

```
route delete 172.16.1.0 192.168.0.20
route delete default 192.168.0.1
```

Linux使用一個有些不太一樣的route命令的語法；它使用關鍵字**gw**來指定一個通訊閘道。route add 和route delete 命令，在Linux中應被輸入像：

*Linux**使用**gw*

```
route delete 172.16.1.0 gw 192.168.0.20
route add default gw 192.168.0.1
```

如預期中，route 不需要知道子網路遮罩，只要網路不是一個子網路。但是當它是子網時，子網路遮罩必須用**netmask**關鍵字來指定，有時，您必須使用關鍵字**net**和目的地位址讓route了解我們正在指定一個網路路徑。偶爾，您可能需要去指定一個主機的路徑，在這例子中，關鍵字 **host**就必須被使用。Linux和Solaris使用-

net和-host來替代。

當使用ifconfig時，這些route敘述將持續有效只要機器在運作的狀況下，他們會在每次系統開機時被執行，因此，它們總是被保留在系統的啟動手稿中。

23.6 netstat：顯示網路的參數

netstat -rn命令顯示路由表。-r選項是列出這個表，-n選項是列出編號型式的IP地址。在前一節中已執行過route add這兩個敘述，這就是為什麼在Linux系統中的表看起來像：

```
# netstat -rn
Kernel IP routing table

Destination   Gateway       Genmask           Flags    MSS Window      irtt Iface
172.16.1.0    192.168.0.20  255.255.255.255   UGH           00          0 eth0
192.168.0.10  0.0.0.0       255.255.255.255   UH            00          0 eth0
192.168.0.0   0.0.0.0       255.255.255.0     U             00          0 eth0
127.0.0.0     0.0.0.0       255.0.0.0         U             00          0 lo
0.0.0.0       192.168.0.1   0.0.0.0           UG            00          0 eth0
```

第一列表示任何目的地的網路為172.16.1.0的封包會被轉送到IP地址是192.168.0.20的機器上。*Flags*下面的G表示它一個閘道器；如果它不存在，那麼主機會直接地被連接。此處提供一個網路地址是有意義的，因為它可以減少路由表的大小。

第三行表示封包屬於網路192.168.0.0會被轉送至機器0.0.0.0，而這機器的路由表被顯示出來。這機器使用自己本身的介面卡，在同一個網路中連接另外一部機器。如果一個封包的目的地位址不能符合第一個欄位的任何項目（地址），那麼就由最後一行來決定。因為在這裡的目的位址是0.0.0.0，預設通訊閘道192.168.0.1將被使用。這裡的0.0.0.0表示地址不符合任何先前的地址（像shell的case敘述使用*的例子）。有時，您可能會看到字串default放在此位置上。

您可以看到在網路上全部的主機有兩個介面，即乙太網路地址（這裡是eth0）和回授地址(lo0)。所有的介面被啟動和運作（*Flag*顯示U）。H指出您可以到達單一主機的路徑。一個封包屬於區域本機(localhost) (127.0.0.0)使用相同的機器(0.0.0.0)當它的通訊閘道。如果您了解這全部，您應該能推算出執行這個命令的機器使用的IP地址為192.168.0.10。


Note

如果一部機器沒有連接到任何的主機，路由表中只顯示單筆的回授l(oopback)介面。這個路徑顯示在一個路由表的用意表示您只能在此主機上使用TCP/IP工具。

23.7 inetd：網際網路服務程式(daemon)

UNIX有大量的服務程式，它們每個都會收聽一個特別的埠編號以接收來自對應的客戶端需求。我們能負擔經常性地執行這些全部的服務程式，即使它們之中的一些大部分的時間都不會被用到？當然不行，比較合理的作法是當有需要時才執行它們。inetd服務程式解決了這個問題。

很多TCP/IP的服務程式，像是telnet和ftp服務被開啟的方式，既不是由客戶端程式也不是由**rc*n*.d**中系統開機的手稿，而是由主要的網際網路服務程式inetd。inetd收聽多重埠的任何連接需求。當它發現需求時，它開始執行在檔案**/etc/inetd.conf**中被定義給那個埠的程式。這個檔每一行包含一個服務：

```
ftp       stream    tcp    nowait    root    /usr/sbin/tcpd    in.ftpd -l -a
telnet    stream    tcp    nowait    root    /usr/sbin/tcpd    in.telnetd
talk      dgram     udp    wait      nobody  /usr/sbin/tcpd    in.talkd
pop-3     stream    tcp    nowait    root    /usr/sbin/tcpd    ipop3d
#imap     stream    tcp    nowait    root    /usr/sbin/tcpd    imapd
#tftp     dgram     udp    wait      root    /usr/sbin/tcpd    in.tftpd
```

第一個欄位顯示服務名稱。它使用的傳輸協定在第三個欄位。如果第四個欄位顯示**nowait**，這表示多個連結可以執行同樣的服務。最後兩個欄位顯示服務程式的絕對路徑和伺服器最後執行的完整命令列的程式。

全部這些行有一個共同的項目，它們被執行的方式，並不是直接藉由inetd，而是經由一個包裝(wrapper)程式叫做tcpd的程式來呼叫。tcpd首先檢查它的設定檔案，看客戶端是否被認可去使用這個服務。如果是，tcpd就執行相對應的服務程式。ftp的服務程式是伺服器程式in.ftpd，它執行時配合參數**-l -a**。tcpd也在另一個檔案中記錄(log)需求。經由tcpd的網路存取是藉由檔案**hosts.deny**和**hosts.allow**來控制。

UNIX系統啟動許多的服務，很多服務可能根本不需要。在這樣的例子中，最好的方式是關閉服務本身，將它所在行之前加上註解。您現在能了解為什麼大部分的UNIX系統預設不執行一般檔案傳輸協定(tftp - trivial file transfer protocol)。

現在，哪一個埠被ftp使用？這個數字是查驗 **/etc/services**檔案來決定。這個檔案是經由它們的第一個欄位服，即務名稱和**inetd. conf** 產生關連。這檔案含有兩個欄位：

```
ftp             21/tcp
telnet          23/tcp
smtp            25/tcp          mail
pop3            110/tcp          # POP第三版
pop3            110/udp
```

埠編號有傳輸協定的標籤附加在編號之後。查驗這個表格不只是決定埠編號，還有傳輸協定被哪一個服務所使用。在這個檔案中，很多服務有兩個項目，亦即一個是tcp，另一個是udp。


Tip

tcpd程式不一定存在所有的系統中。在一些系統上（像是Solaris），服務程式是直接由inetd啟動。例如ftp服務，是由inetd直接啟動in.ftpd（或在較舊的系統上之ftpd）而不是經由包裝程式tcpd。在這樣的狀況，/etc/inetd.conf中的最後兩欄使用同樣的命令名稱。

23.8 pppd：點對點傳輸協定(PPP)

您的電腦可能沒有一張網路卡，或是您只能用電話線路來維持與外面世界連絡。今天，您能經由序列埠（在DOS中的COM1和COM2）來連結兩部機器，然後在此連接上執行TCP/IP。一個特別的傳輸協定可以實現這種做法，即點對點傳輸協定(**Point-to-Point Protocol**-PPP)。在序列埠的通訊範圍中，PPP已經替代了SLIP和UUCP，它們之前也是風光一時。在今天，它變成一個標準方式，讓使用者在他們電腦的序列埠上用一個modem（譯注：**mod**ulator-**dem**odulator：調變與解調變器），數據機來存取網路（雖然它輸給了新的技術方法）。

PPP是一個奇特的傳輸協定。它在兩個主機間設定一個連結，使用它自己本身的一組IP地址，因此其他的傳輸協定像ftp和telnet能在之後執行。PPP之後在通訊過程中就沒有額外的參與除了確保持續的連線之外。它也沒有分開的客戶端和伺服器端組件。同樣的pppd命令必須執行正確的選項來扮演像是一個客戶端也像是一個伺服器端程式。在這節中，我們將討論BSD PPP的封包，它被Linux和很多UNIX系統所使用（但不是Solaris）。

pppd命令在Linux系統中位於/usr/sbin中，我們假設您是以一個系統管理者來登入及執行pppd，一個普通的使用者帳號需要一些設定才能執行它。如果您連接到您的ISP（譯注：Internet Services Provider：網際網路服務提供者），您必須在您的本地機器上啟動pppd 程序，像這樣：

```
/usr/sbin/pppd /dev/ttyS0 115200 crtscts modem defaultroute noipdefault -detach
```

/dev/ttyS0 是modem設備連接到第一個序列埠(COM1)，這埠的速度（這裡是115,200 鮑率(bauds)，換言之，每秒傳送幾個位元）被設定成至少是modem能處理最高速度的三倍，有一個理由被隱藏其中(23.9.2)。此命令設定了另一個介面(ppp0)，其屬性能用ifconfig和netstat -rn命令來顯示。

您必須執行PPP在您的電腦（客戶端）和您的ISP（伺服端）之間來存取網際網路。這意味著，一個類似的PPP程式也必須在另一端執行。如果客戶端的PPP介面

被設定成沒有單獨的IP地址（像上面），在建立連接前，伺服器端就必須提供這個地址。


Tip

如果您必須一直使用pppd**並使用一組固定的設定選項，那麼您可將大部分的這些選項放在**pppd**的設定檔/etc/ppp/options中，此檔案存放一行一個選項。**

pppd有一複雜的命令列，您最好了解它完整的語法，萬一您必須使用一個像chat的工具以手動啟動這個程式。您也必須了解為什麼上面的命令要用這些選項來執行：

- 因為PPP使用全部的八個位元，硬體流程控制**(hardware flow control)**必須被使用 (crtscts)，因此系統能調整資料流(stream)的流程。
- 軟體流程控制**(software flow control)**必須被關閉。我們不希望字元*[Ctrl-s]*和*[Ctrl-q]* 被modem解釋成開始和結束字元。這可以由pppd不要使用xonoff選項來確保。
- 不屬於區域網路的任何IP封包之預設路由(defaultroute)，它必須被繞送經過PPP介面。
- 大部分ISP支援動態地址**(dynamic addressing)**，這意味著伺服器必須提供客戶端的IP地址(noipdefault)。PPP也允許設定您自己的IP地址。
- pppd必須不允許從個別的終端機(-detach)被斷線，否則將不能持續連接。

當我們能安全的假設您的ISP知道它要做什麼，我們將不考慮伺服器端的選項。然而，如果您要設定一個PPP連接，將您家裡和公司的機器之間連線，您就必須知道伺服器的選項。


Note

PPP可能是TCP/IP**工具中唯一沒有分別的客戶端和伺服器端組件。它既不是用系統的啟始手稿來啟動，也不是用**inetd**來運作。**


Tip

如果您要用一般使用者帳號來用pppd**，您可能必須設定它的**SUID**位元**(22.4.1)**。這可以簡單的使用**chmod a+s /usr/sbin/pppd**來完成。Linux的一些版本只是求使用者屬於同樣的群組，像是**pppd**的擁有者。**

23.9 使用PPP來連接網際網路

您現在將使用一個Linux機器來連接到一個ISP，可經由它們的通訊閘道至網際網路。您必須連接一部modem到一個序列埠（假設COM1）和使用它來撥接出去連到ISP。在Linux中安裝modem是一件很直接的任務，只要在/dev/modem和/dev/ttyS0 之間（假設您使用第一個序列埠）設定一個符號連接。/dev/modem的列表顯示應該看起來像這樣：

```
lrwxrwxrwx   1 root     root           10 Dec  5 08:44 /dev/modem -> /dev/ttyS0
```

ISP通常使用UNIX機器，您必須使用一個已註冊過的使用者名稱和密碼來登入到他們的電腦中。一旦登入後，PPP程序必須在伺服器端啟動。如果ISP的機器不能自動地做這件事（通常它會做），那麼您就必須手動啟動它。在這之後，您必須離開ISP的機器，在您的機器上不用重設modem然後啟動PPP。

ISP的PPP程序通常會分配您的IP地址，且通常是動態的從一堆地址中給您一個。您已在您的機器上啟動pppd之後，此連結被建立在您的電腦和ISP的主機之間。這個連結也提供您一個預設的路由（在pppd內的**defaultroute**選項）連到網路。您的電腦於是變成網際網路的一部分。ISP的機器沒有進一步的任務來被使用，除了提供PPP連結，剩下的只是直到我們掛斷電話為止。

您的ISP也可能為您維護一個或更多的名稱伺服器**(name servers)** (24.2)。這些機器執行主機名稱解析到IP地址的工作。存取名稱伺服器是必要的，因為所有的網域名稱必須被解析而且解析出的IP地址要被加入PPP封包內，在這些封包從您的機器送出之前。ISP通常也提供電子郵件服務和維護一個新聞伺服器，因此您可以存取新聞群組(newsgroups)。

23.9.1 指定名稱伺服器和解析系統

您的ISP應該提供您它們的名稱伺服器的IP地址，現在，您需要去修改您的Linux系統中的兩個檔案，簡單的插入這兩行到**/etc/host.conf**中：

```
order hosts bind                                    用BIND啟動名稱伺服器
multi on
```

一個解析系統(resolver)是一個客戶端(24.2.1)，它代表應用程式提出需求，讓名稱伺服器來轉換一個FQDN 成為IP地址。在UNIX系統上的解析系統使用設定檔案**/etc/resolv.conf**。請放置您ISP的名稱伺服器地址在這檔案中（此處表示被作者的ISP使用的名稱伺服器）：

```
nameserver 202.54.1.30                              主要名稱伺服器地址
nameserver 202.54.9.1                               和第二個名稱伺服器
```

這樣完成了解析系統的設定，您的機器現在知道它必須使用BIND（在**/etc/hosts**查詢失敗之後）來解析FQDN，它也知道為此目的要去連絡的機器位置。

23.9.2 獲得手稿參數

要連結到網路，您將使用兩個以字元為基礎的(character-based)工具。最後，您將學習經由一個手稿傳遞全部的輸入至遠端系統，但您必須知道另一端系統預期接收什麼字串。因此，就撥號出去吧，注意到提示符號和輸入適當的字串。minicom是一個適合此用途的理想撥接工具，因為它能使用Hayes公司的“ AT ”命令。這些

字串都是用**AT**起頭，它能撥接一個modem，也能執行初始化。

圖 23.1 **登入到一個網際網路服務提供者的**PPP**伺服器**

```
Press ALT-Z for help on special keys

AT S7=45 S0=0 L1 V1 X4 &c1 E1 Q0
OK
atz
OK
atdt5599001
CONNECT 28800/ARQ/V34/LAPM/V42BIS
User Access Verification

Username:  sumit
Password:  *********
gicaro31>  ppp
Entering PPP mode.
Async interface address is unnumbered (Ethernet0)
Your IP address is 202.54.52.240. MTU is 1500 bytes
~~y}#.!}!q} }4}"}&} }*} } }%}&Tb..}'}"}(}"!.~~y}#.!}!r} }4}"}&} }*} } }%}&Tb..}~
~~y}#.!}!q} }4}"}&} }*} } }%}&Tb..}'}"}(}"!.~~y}#.!}!r} }4}"}&} }*} } }%}&Tb..}~
 .....
```

在您撥號出去前，確定您已設定modem的速度到它最高速處理的能力。大部分的modems和ISP提供資料壓縮，這結果增加了modem的速率，其通常是係數4。這意味著您能安全的使用115,200 bps （每秒傳輸元位數：bits per second），當作一個33.6 kbps（每秒傳輸千位元：kilo bits per second） modem的速度。如果您使用minicom和一個33.6或56 kbps的modem，執行minicom -s 然後設定序列埠速度到誇張的數字。離開minicom，然後以正常狀態重新執行它。

您現在必須使用一 些“ AT ” 命令。首先，用atz重設modem，然後用atdt撥號出去，接著是ISP的電話號碼。在需要驗證使用者的兩個提示需求中輸入一般的字串。我們假設ISP不能自動的啟動pppd程序，因此我們將自己做，這顯示在圖23.1中。

此圖表示動態IP地址在工作中；您的ISP已指定您的IP地址為202.54.52.240。您所看到好像亂碼的「垃圾(junk)」，其實是一個簡單的PPP封包被產生在伺服器端 。檢查這個輸出您會注意到三個字串是您應該預期的：Username:、Password: 和gicaro31>。您應該送出的回答字串必須要接在它們之後。我們現在將使用這個訊

息來開發手稿。

23.9.3 使用dip

在這些以字元為基礎的撥號工具中，dip被使用在許多不同的UNIX系統中，像SunOS, AIX, Ultrix和Linux。它有基本程式的架構並提供錯誤處理的邏輯。它也能被程式化來重新撥號，以下就是您應該使用的手稿加上由上面獲得的預期送出(expect-send)字串：

```
# cat dipdial.dip                          習慣是用.dip的副檔名
get $local 0.0.0.0                         區域IP地址是動態的
port modem                                 符號連接應該被建立
speed 115200                               建議用在一個33.6 kbps的modem
dialstart:
reset
flush
send atdt5599001\r                         也可使用dial 5599001
sleep 2
wait CONNECT
wait name: 20                              等候name:提示最多20秒
if $errlvl != 0 goto redial                如果沒有OK
send sumit\r
wait word:
send a9h4uil\r                             密碼能看見!!
wait >                                     這是處理gicar031>
send ppp\r                                 在伺服器端啟動pppd
mode PPP                                   在客戶端上啟動pppd
exit
redial:
sleep 1
goto dialstart
```

這個手稿只需要少許的解釋，因為它已經提供了大部分。注意在這裡的*[Enter]*鍵是用**\r**表示，即也可被echo命令認識的一個跳脫序列。我們在一個提示字串中使用次要字串(substrings)是允許一些較小的變化。舉例來說，**name:**可同時處理**username:**和**Username:**。當連結成功時，**$errlvl**變數被設定為**0**。如果遇到錯誤，if敘述保證modem會重新撥號。注意dip啟動pppd程序時在客戶端上只用一個簡單的敘述(mode PPP)。如此就有啟動pppd 的效果，而其對應的命令列之前已經顯示過。

請您儲存這些內容到檔案**dipdial.dip**中，您必須執行dip命令如下：

```
/usr/sbin/dip -v dipdial.dip
```

此處包含了所有要做的工作。它應該撥號modem，讓您登入，在兩端啟動PPP的程序後及回到您的提示符號。現在您已經在網際網路上，您應該能使用任何的網路工

具和在本章中所介紹的功能。使用dip -k 來中止pppd。如果這方法行得通，就可以省略**-v**（詳細：verbose）選項因為它將您的密碼顯示在螢幕上。

Note

如果伺服器本身就啟動PPP，這兩個敘述wait >**和**send ppp\r**必須從手稿中移除。**

Tip

如果您有一個脈衝撥號(pulse-dialing)**電話系統，使用**atdp**代替**atdt**。如果您使用一個較慢的**modem**，假定是**14.4 kbps**，設定速度到**38,400**。如果您的**modem**不能撥號，試著使用**atx3dt**代替**atdt**。如果以上全部失敗，試著用**wvdial**和**wvdialconf**命令，假如它們在您的**Linux**中系統可被找到。**wvdialconf **通常能為您建置這些設定。**

23.9.4 使用chat

如果您的系統沒有dip，您可以試試chat命令。它也使用一個手稿，在一個chat 手稿中的每一行（除了一些行含有**ABORT**外） 包含有一對預期送出的字串，chat使用一個更簡單易用的手稿：

```
# cat chatdial.chat
' ' atz                                  在執行atz前不預期任何事情發生
OK atdt5599001                           預期 OK 然後撥號
ABORT BUSY                               檢查您的modem是否有顯示 BUSY
ABORT 'RING - NO ANSWER'                 或 RING - NO ANSWER
CONNECT ' '                              預期 CONNECT 和不送出任何東西
name: sumit                              現在預期送出的三對字串
word: a9h4uil
> ppp                                    如果伺服器本身啟動pppd就不需要
```

一個空字串或回應表示為''，這個手稿注意到**BUSY**和**RING - NO ANSWER**的訊息，您的**modem**可能因此而放棄，並使用關鍵字**ABORT**來中止。注意到，此手稿不會在客戶端啟動**PPP**。我們在這裡採取一個不同的方法：執行pppd 命令和使用它的**connect**選項來執行chat手稿。整個動作能在背景中進行：

```
/usr/sbin/pppd /dev/ttyS0 115200 connect "/usr/sbin/chat -f chatdial.chat" \
crtscts modem defaultroute noipdefault -detach  &
```

不像dip，chat沒有選項來殺掉pppd程序。您能用一般的方法先執行ps然後殺掉pppd。但是因為pppd程序的PID會被存在一個文字檔**/var/run/ppp0.pid**中，您也能像這樣的殺掉它：

```
kill -9 `cat /var/run/ppp0.pid`          為它定義一個alias
```

在這裡也沒有重新撥號的機制，因此，如果您經常用單行指令以設法連結到您的ISP，那您就能使用chat。請把這個序列放在一個指令手稿中或是設計一個簡單容易管理的alias。

23.9.5 建立連結後

可能dip和chat在沒有任何障礙的狀況下就成功了，但您仍舊不能連結上一個遠端站台。在這種狀況中，您首先應該用netstat -rn或route -n測試您的PPP介面是否已啟動：

也可以使用 route -n

```
# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.3     0.0.0.0         255.255.255.255 UH    0      0        0 dummy0
202.54.1.30     0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        1 lo
0.0.0.0         202.54.1.30     0.0.0.0         UG    0      0        0 ppp0
```

您現在應該看到有關於**ppp0**的兩行。第二個還顯示使用pppd命令的**defaultroute**之選項結果。“Ping”伺服器，如果這可以動作，然後請試著ping一個已知的主機，假設是ping rs.internic.net。如果ping 不能產生一般的輸出，就要檢查設定在撥號手稿中的modem速度。

pppd在連結啟動之後會執行手稿/etc/ppp/ip-up和連結終止之前會執行/etc/ppp/ip-down手稿。使用者的活動被限定只能使用網頁時，通常可能不需要使用這些檔案。但ip-up用來自動化程序並從網路的一部伺服器寄送或下載郵件是有用的。如果您有自己的撥接帳號，您會想要節省連接時間，並藉由這個手稿來執行sendmail(24.7.1)和fetchmail(24.11.1)命令而不是手動。


Caution

您有時可能會得到modem**被鎖住**(is locked)**的訊息，在這種狀況下，錯誤訊息將告訴您刪除檔案/var/lock/LCK..modem。請依照指令動作。如果在/dev/modem的寫入位元**(write bit)**被關閉，可使用**chmod a+w /dev/modem**將它打開。**

23.10 PAP和CHAP認證

大部分的ISP使用某些型式的認證來檢查主機上一些使用者的身份。PPP提供兩個形式的認證：

- 密碼認證傳輸協定**(Password Authentication Protocol-**PAP)。在這系統中，文字式密碼和使用者名稱經由網路來傳送並進行認證。
- 查問交互訊號認證傳輸協定**(Challenge Handshake Authentication Protocol-**CHAP)。這是一個雙向認證的機制，密碼不會傳送到網路上。

PAP和CHAP利用分享的機密**(shared secret)**讓客戶端和伺服器端彼此都知道該

機密。它通常是使用者的密碼，而且PAP把它儲存在檔案/etc/ppp/pap-secrets中，檔案中的典型的一行如下：

```
henry       starisp.com         my:pass,word
```

第一個欄位是使用者的主機名稱。如果客戶端的機器不在網路上（像這例子使用撥號連接），ISP的機器不需要知道這個名字。PPP能被執行來翻譯這個欄位成為已註冊在ISP上的使用者名稱而不是一個主機名稱。第二個欄位是ISP的FQDN；一個*在這裡表示符合所有的ISP。第三個欄位包含分享的機密，在這例子中就是密碼。第四個欄位通常是保持空白。

當一個使用者撥號到ISP時，密碼用明碼(clear text)送到ISP端。只有當密碼符合ISP /etc/ppp/pap-secrets檔案中儲存的內容時連線才會啟動。

PAP的防護比根本沒有防護還好一點，因為一個封包偵測器就能攔截傳送的文字密碼。明白地說，在乙太網路上比在電話線路中還容易被攔截，但它仍然很容易的就會遭受攻擊。即使PAP仍舊廣泛的被使用，CHAP提供一個更安全的認證。

PPP使用CHAP當作預設的認證傳輸協定，CHAP實際上並不會交換密碼，而是伺服器送一個查問字串(**challenge string**)給客戶端。客戶端回應時藉由hashing（編碼）查問字串和它自己維護並儲存在/etc/ppp/chap-secrets中一個類似形式之分享機密。伺服器也會在它這端做同樣的動作，因為它也知道分享的機密，然後它會比較它自己計算出的編碼及從客戶端接收來內容，當兩者比對成功時連線就會被建立。

使用任一種形式認證，在伺服器端的pppd必須加上**auth**選項來執行。不過，客戶端這邊必須用 **user** （此處的 **user henry**)選項來執行。因為認證是由電腦計算而來不是經由登入，之前的dip和chat手稿必須被修改成刪除全部有關於預期傳送的字串。關於dip，您能移除**CONNECT**之後的所有敘述，但請保留mode PPP命令。 您也需要使用/etc/ppp/options檔案來放置更多的選項。chat手稿在**CONNECT**之後不需要有敘述。


Caution

因為，檔案pap-secrets和 chap-secrets含有密碼或查問字串，他們必須讓全部的使用者除了root以外都沒有讀取權限。否則，系統將不安全，像是沒有使用任何的認證一樣。

23.11 網路檔案系統

使用TCP/IP登入遠端系統和傳輸檔案的機制，通常是不夠用的，要傳輸一個大型資料庫是非常不切實際。更好的作法是如果遠端檔案系統能被掛載(mount)到本

地檔案系統，所以使用者不需要用任何特別的命令來存取遠端系統。Sun Microsystems（昇陽）覺得有這需要並創造了網路檔案系統**(Network File System)** (NFS)的觀念，如今它幾乎已經移植到所有UNIX的平台上。

一個網路檔案系統將一個遠端檔案系統到掛載一個本地的目錄上。讓他們被當作是本地的連接。舉例來說，如果/datab檔案系統位在主機 *fredo*上，而被掛載到本地的目錄/oracle上，您將不知道/oracle是一個本地或一個遠端的檔案系統。去存取遠端（掛載）檔案系統的這些檔案並不需要特別的命令。

NFS的機制非常有用是有一些理由的。與其鼓勵使用者建立檔案的拷貝（像是ftp），NFS允許使用者工作在數個電腦中並能分享檔案。這意味著您能有一些大容量的磁碟放在幾個電腦中，而被其他使用者存取。它也使得系統維護和備份更容易，因為系統管理者需要去備份和維護一組的檔案，而不是去備份多個機器上的相同副本。

不像本地掛載，NFS並沒有限制只能掛載檔案系統。它允許您掛載任何遠端目錄，即使目錄不能構成一個分割的檔案系統。使用者可能被允許在一個目錄上的權限為讀取或是讀寫。您也能指定哪些主機被允許使用這些功能。然而，NFS不能在使用者層級設定權限。 如果您已允許一個主機可來存取，那就是您信任在這部主機上的所有使用者。這讓NFS有點不安全，特別是當提供可寫入服務時。


Tip

當您在一個過載的系統上用完磁碟空間時，與其增加一個單獨的硬碟，您可加入一個單獨機器本身，然後遠端掛載它的檔案系統，它的好處是機器可以一直處在運作狀態。

23.11.1 安裝NFS

NFS主要靠伺服器端的兩個服務程式mountd和nfsd 來管理（在Linux中的rpc.mountd和rpc.nfsd)。nfsd實際執行客戶端裝載和卸下(umount)檔案系統的工作，mountd驗證使用者的需求。

用來管理NFS檔案系統所使用命令是一樣的（mount和umount），但使用在此處時做了一些修改。大部分的系統使用BSD型式的NFS，它會export檔案系統而不像Sun Solaris是 share它們。在本章中，我們將討論BSD系統，它被一群主機製造廠商所使用，像是HP-UX, IRIX , SCO, 和Linux。

當客戶端發佈一個裝載需求時，mountd檢查伺服器端的/etc/exports檔案來驗證客戶端存取的權限。這檔案每一行中包含能被遠端裝載的每個匯出(export)目錄。以下是 /etc/exports中一些典型的項目：

```
/                                          所有主機可讀寫根目錄！
/project3/doc
/java/programs    -ro                      所有主機可唯讀取用
```

```
/hrd/html   -access=fredo:tessio                    只有主機fredo和tessio可存
/prog/html  -rw=michael                             只有主機Michael可讀寫
```

Linux的格式是不同的，在目錄名稱之後是以空白分隔的一個或數個主機列表。每一個主機名稱之後有圓括號()，其中包含權限形式。前兩行在這系統的設定也是相同的，所以就不顯示它們：

```
/java/programs  (ro)                                全部使用唯讀
/hrd/html  fredo(rw) tessio(rw)
/projects       *.planets.com(rw)                   所有在planets.com網域的主機
```

第一行指定每一個主機只有唯讀權限，而第二行提供讀寫存取權限給*fredo*和*tessio*主機，最後一行允許所有在 *planets.com*網域的主機有讀寫權限。

一旦目錄被裝載，就由nfsd來做存取控制。確定mountd 和充足數目的nfsd程序有被執行，然後用exportfs命令匯出**/etc/exports** 中的目錄：

```
exportfs -a                                         Red Hat使用同樣的命令
```

如果您的系統沒有exportfs命令，此時您需要找到手稿中包含啟動mountd和nfsd服務程式的敘述。如果這個手稿沒有一個重新啟動(**restart**)的功能，就用 ” stop ” 來停止服務和用 “ start ” 來再一次啟動它們，藉由把這兩個字串當成手稿的參數來執行。

一旦伺服器準備好了，客戶端只需簡單的執行mount命令。這次我們必須指定檔案系統的形式為 **nfs**。此處，我們設定一個軟性(**soft**) 裝載並在背景(**bg**)執行和唯讀模式：

```
mount -r -F nfs -o soft,bg sunny:/project3/doc /fredo/project3
```

如果裝載時失敗，軟性裝載確保客戶端不會重試這個動作。這個模式在讀取文件的狀況被推薦使用。此處，即使目錄可被裝載成讀寫模式，mount的**-r**選項確保**/etc/exports**中的設定被重新定義。除了mount選項外，您也能使用**remount**選項，它們在(21.10.3)中已被討論過。

像本地檔案系統一樣，NFS檔案系統也能被指定在**/etc/fstab**中。之後的mount指令就不需要再使用這樣複雜的設定選項。如果**fstab**包含這個輸入，上面的裝載動作能在開機的時候就被自動地執行：

```
sunny:/project3/doc  /fredo/project3 nfs ro,soft,bg 0 0
```

NFS是經由網路分享檔案的方法中最常被使用的。它能相當有成本效益的把經常使

用之單一軟體複製到一個已裝載的檔案系統上。這裡的優點是讓軟體更新只要在一個地方發生。一些組織使用NFS來匯出郵件佇列目錄/var/mail來集中管理它們的郵件。


Tip

如果您使用NFS來存取一個遠端檔案系統上您自己擁有的檔案，確認您擁有的UID和GUID與在這部伺服器上的符合。如果它們不符合，用ls -l的列表就會顯示它們的編號而不是名稱。此外，對您自己擁有的檔案可能沒有該有的存取權利。在這樣的情況下，您最好修改 /etc/passwd來符合UID和GUID。

摘　要

TCP/IP網路的功能主要靠兩個傳輸協定來控制，即傳輸控制協定(TCP)和網際網路傳輸協定(IP)。TCP是一個可靠的傳輸協定。如果不能及時的在另一端被接收，它會重送遺失的區段。如果必要的話，IP會注意到封包位址和轉送封包到最近的一個路由器。

多少八位元組數字來代表網路地址和第一個八位元組的值來決定一個網路的class（A, B 和 C)。一組分別的網路地址被預留給區域網際網路來使用，這些地址不會被使用在網際網路上。

每一個網路需要個別的IP地址來當回授地址和廣播地址。子網路遮罩轉譯一個IP地址，它能反映出多少個位元是從主機地址中借出，以形成網路地址的部分。

網路介面卡有唯一的MAC地址，它的IP位址必須被轉換成此地址才能讓一個訊息到達一個主機。當設定介面卡上時，您必須指定IRQ和I/O地址，它們應該和機器上使用的卡片沒有衝突。

ifconfig設定網路介面的IP地址和子網路遮罩。它也能啟用和關閉一個介面。使用ping和netstat來檢查網路的連接。

核心會維護一個路由表，它由IP查驗以作為繞送之依據。route可設定路徑到一個網路和顯示路由表。這表指出要被用來繞送封包到外部網路的閘道器。預設路徑被提供在路由表中以備萬一查驗失敗時使用。

很多TCP/IP服務，像telnet, ftp及pop3是經由inetd 執行，而它讀取在/etc/inetd.conf檔案的項目。這些服務程式通常藉由包裝程式來啟動（像tcpd)而不是直接地被啟動。被這些服務程式所使用的埠編碼指定在/etc/services中。

點對點傳輸協定(PPP)讓TCP/IP的功能可運作在一條電話線上。pppd使用硬體流程控制，但不是軟體流程控制。大部分的ISP使用pppd來提供動態IP地址在連接的兩端。在啟動和中止前，pppd分別執行/etc/ppp/ip-up和/etc/ppp/ip-down手稿

命令。

要連結到網際網路，名稱伺服器地址必須被設定在/etc/resolv.conf中。dip和chat是兩個手稿形式的工具，它使用預期傳送字串對來執行自動登入的程序。pppd 呼叫chat的手稿， 而dip能從手稿本身啟動pppd服務。PPP連接問題應該用netstat -rn和ping來檢查。

大部分的ISP不是使用PAP就是使用CHAP來驗證使用者。PAP傳送儲存在/etc/ppp/pap-secrets中的明碼密碼，CHAP使用儲存在/etc/ppp/chap-secrets中的分享秘密和查問字串（伺服器送出）來做計算。這些檔案都被保留在連接的兩端。CHAP比PAP更加安全。

網路檔案系統(NFS)讓您掛載一個遠端檔案系統或一個目錄到一個本地目錄上。目錄及它們的存取權限被維護在/etc/exports檔和用exportfs來做匯出。NFS沒有提供使用者層級的存取。/etc/fstab, mount和umount 被使用的方式幾乎一樣，就像在獨立的主機一般。

自我測試

23.1 請說出這些IP地址是屬於網路Class中的哪一個： (i) 202.54.9.1, (ii) 107.35.45.78, (iii) 34.67.102.34。

23.2 您能用11.23.34.45和172.26.0.6的主機IP地址在網際網路上嗎？

23.3 有哪兩個網路介面卡的硬體參數，在您的機器上設定TCP/IP之前可能需要手動設定？

23.4 您如何在您的系統上，找出網路介面卡的IRQ和I/O地址？

23.5 您如何關閉ftp服務？

23.6 哪一個傳輸協定最常被使用在TCP/IP的撥號連接上，用什麼命令來啟動此服務？

23.7 用哪一個Hayes-modem命令您能做撥號動作？

23.8 /etc/exports包含什麼，在您改變它之後，您如何讓mountd重新讀取這個檔案？

練 習

23.1 您如何找出您的機器的MAC地址？

23.2 什麼是廣播地址， 它使用哪一個IP地址？

23.3 回授地址的重要性是什麼？

23.4 如果您設定一個Class C的內部網路，假如您使用正式的指導原則，那麼前兩

個八位元組會是什麼？

23.5 您如何關閉您的網路而不使用任何的*rc*手稿？

23.6 如果ping顯示一些“ packet loss ”，它說明了什麼？

23.7 您的機器連接到網際網路上另一個IP地址為**192.168.0.1**的主機，假設閘道機器允許您傳送封包至網際網路，您如何在您的SVR4和Linux機器上設定路由？

23.8 您如何找出您路由器的IP地址？

23.9 您如何在您的機器上關閉telnet服務？

23.10 您不能從您的無特權使用者帳號來執行/usr/sbin/pppd。可能的原因是什麼？

23.11 您只能使用IP地址連結至一個在網際網路上的ftp站，但不是FQDN。名稱伺服器已執行在您的網路中，那是什麼設定您忘了考慮到？

23.12 為什麼CHAP認證優於PAP？

23.13 您需要用一個手稿連結到網際網路上的ftp站，但在執行ftp命令前必須先確認文字**ppp0**被顯示在netstat -rn的輸出。您要如何做？

23.14 使用單一行perl程式來改變所有您**.chat**和**.dip**手稿中的密碼，從**s1o3n5y8**更改為**j2n98d0k2**。

23.15 您如何在SVR4系統上使用mount指令以唯讀模式來存取一個遠端主機*uranus*上的目錄**/usr/doc**？