PPP HOWTO 中譯版

作者: Robert Hart *hartr@interweft.com.au*
譯者: *Asd L. Chen* & *C.W.Huang* v3.0, 31 March 1997. 翻譯日期: 25 August - 9 October 1997

這份文件展示如何把你的 Linux PC 連接到一台 PPP 伺服器上，如何使用 PPP 把區域網路連結在一起並且提供一種把你的 Linux 電腦設定爲 PPP 伺服器的方法．本文同時提供幫助偵錯無法正常運作的 PPP 連結．

目錄

1 簡介 **1**

1.1 客户端以及伺服端 . . . 2

1.2 不同 Linux 套件間的差異 . . . 2

1.3 套件相關的 PPP 配置工具 . . . 3

2 IP 號碼 **3**

3 這份文件的目的 **4**

3.1 設立 PPP 客户端 . . . 4

3.2 使用 PPP 連結兩個區域網路或連結一個區域網路到網際網路上 . . . 5

3.3 設立 PPP 伺服器 . . . 5

3.4 使用 PPP 於直接纜線(direct null modem)連結 . . . 5

3.5 這份文件目前尚未涵蓋的... . . . 5

4 涵蓋的軟體版本 **5**

5 其它有用的重要的文件 **6**

5.1 有用的 Linux 電子郵遞列表 . . . 7

6 將 PPP 設定爲客户端的工作概要 **7**

6.1 取得/安裝軟體 . . . 7

6.2 編譯核心加入 PPP 支援 . . . 7

6.3 從你的 ISP 取得資訊 . . . 8

6.4 設定你的數據機及串列埠 . . . 8

6.5 建立名稱到位址解析(DNS) . . . 8

6.6 PPP 與 root 權限 . . . 9

6.7 檢查你的 PPP 套件檔案及設定 PPP 選項 . . . 9

6.8 如果你的 PPP 伺服器使用 PAP 認證(Password Authentication Protocol) . . . 9
6.9 以手動方式連接至 PPP 伺服器 . . . 9
6.10 使你的 PPP 連結自動化 . . . 10
6.11 結束連結 . . . 10
6.12 如果你遭遇問題 . . . 10
6.13 建立連結之後 . . . 10
6.14 動態 IP 連接提供標準 IP 服務的問題 . . . 10

7 配置你的 Linux 核心 **10**
7.1 安裝 Linux 核心原始程式碼 . . . 11
7.2 瞭解你的硬體 . . . 11
7.3 編譯核心 — Linux 1.2.13 核心 . . . 11
7.4 編譯核心 - Linux 1.3.x 及 2.0.x 核心 . . . 12
7.5 PPP-2.2 與 `/proc/net/dev` 的注意事項 . . . 13
7.6 配置核心的一般考量 . . . 13

8 取得你所需的 PPP 伺服器資訊 **14**

9 設定你的數據機及串列埠 **14**
9.1 串列埠與速度相容性的注意事項 . . . 15
9.2 串列埠名稱 . . . 16
9.3 配置你的數據機 . . . 16
9.4 串列流量控制注意事項 . . . 17
9.5 測試你的數據機的撥出功能 . . . 17

10 建立名稱到位址解析(DNS) **18**
10.1 `/etc/resolv.conf` 檔案 . . . 19
10.2 `/etc/host.conf` 檔案 . . . 20

11 使用 PPP 與 root 權限 **20**

12 設定 PPP 連接檔案 **21**
12.1 替代的 options.tpl 檔案 . . . 22
12.2 我應該使用什麼選項? (無 PAP/CHAP) . . . 28

13 如果你的 PPP 伺服器使用 PAP 認證(Password Authentication Protocol) **29**
13.1 使用 MSCHAP . . . 29

13.2 PAP/CHAP 密碼檔案 29
13.3 PAP 密碼檔 30
13.4 CHAP 密碼檔 31
13.5 處理多重 PAP 認證連接 31

14 以手動方式建立 PPP 連結 **32**

15 使你的連結自動化 - 創造連結指令稿 **35**
15.1 以使用者名稱密碼認證的指令稿 35
15.2 ppp-on 指令稿 35
15.3 編輯這裡所提供的起始指令稿 37
15.3.1 ppp-on 指令稿 37
15.3.2 ppp-on-dialer 指令稿 38
15.4 一份 chat 指令稿其中的含意... 38
15.4.1 在伺服器端啓動 PPP 40
15.5 使用於 PAP/CHAP 驗認連線的 chat 指令稿 40
15.6 pppd 的 debug 與 file option_file 選項 41

16 測試你的連線指令稿 **41**

17 結束 PPP 連結 **43**

18 偵錯 **43**
18.1 我把 PPP 編譯進去但是 Linux 說我沒有... 44
18.1.1 你啓動的是正確的核心嗎? 44
18.1.2 你將 PPP 核心支援編譯爲模組嗎? 44
18.1.3 你是否你用正確的 PPP 版本配合你的核心? 44
18.1.4 你是否以 root 身份執行 pppd? 44
18.2 我的數據機連上了但 PPP 並未啓動 44
18.3 系統記錄說 “serial line is not 8 bit clean...” 45
18.3.1 你沒有正確地簽入伺服器 45
18.3.2 你並未啓動伺服器上的 PPP 45
18.3.3 遠端的 PPP 過程啓動很慢 45
18.4 不能設立預設遞送路徑 46
18.5 其它問題 46

19 當完全陷於膠著時如何取得幫助 **46**

20 連結運作之後的普遍問題 **46**
20.1 我無法看見我連線的 PPP 伺服器以外的世界 47
20.2 我可以寄送電子郵件，但不能接收 47
20.3 爲什麼人們不能來我的機器 finger, WWW, gopher, talk 等等？ 47

21 以動態 IP 提供網際網路服務 **47**
21.1 設定電子郵件 48
21.2 設定本地的名稱伺服器 48

22 使用 PPP 連結兩個網路 **49**
22.1 設定 IP 號碼 49
22.2 設定遞送路徑 49
22.3 網路安全 50

23 建立連結之後 - /etc/ppp/ip-up 指令稿 **50**
23.1 特殊遞送路徑 50
23.2 處理電子郵件佇列 50
23.3 一個 /etc/ppp/ip-up 指令稿的例子 51
23.4 處理電子郵件 52

24 使用 /etc/ppp/ip-down **53**

25 區域網路上的遞送問題 **53**
25.1 安全上的注意事項 54

26 建立 PPP 伺服器 **54**
26.1 編譯核心 55
26.2 伺服器系統的概觀 55
26.3 所需的全部軟體 55
26.4 設定標準的撥接(shell access) 56
26.5 設立 PPP 選項檔案 56
26.6 設定 pppd 以便讓使用者（成功地）執行它 56
26.7 爲 pppd 設定一個全域的別名(alias) 57

27 在 null modem (直接連線)上使用 PPP **57**

散布

旦這份文件有新的版本產生就會貼到 comp.os.linux.answer 討論群． 也可以在此取得 HTML 格式的文件:

- *Linux Howto Index* <http://sunsite.unc.edu/LDP/HOWTO/>
- *PPP-HOWTO* <http://www.interweft.com.au/other/ppp-howto/ppp-howto.html>

其它格式(SGML, ASCII, postscript, DVI)的文件可以從這裡取得: *Howtos - other formats* <ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/other-formats>.

因爲 sunsite.unc.edu 的負載非常地重，請使用最靠近你的適當映射(mirror)節點．

感謝

不斷地有許多人提供我寫作此文件的協助． 特別感謝 Al Longyear 對 PPP 本身的指導(如果這裡有任何錯誤，那是我的而不是他的)， Greg Hankins(Linux HOWTOs 的維護人)以及 Debi Tackett (MaximumAccess.com) 許多在形式，內容順序，邏輯與解釋清晰度的有用建議．

最後，對於許多以電子郵件提供我意見的人，感謝您． 就像所有 HOWTO 的作者一樣，幫助他人所獲得的滿足感就是我們所收到的報酬，而這已足夠． 經由寫作這份 HOWTO 文件我償還一小部份人情債，這是我，以及所有其餘 Linux 的使用者， 所積欠給爲我們撰寫與維護這個作業系統的人的．

1 簡介

PPP （點對點協定）是在串列連結上跑 IP （網際網路協定）以及其它網路協定的一種機制， 串列連結可以是直接的串列連接（使用 null-modem 纜線）或是使用數據機以及電話線路所造出的連結 (當然也包括如 ISDN 的數位線路)．

使用 PPP, 你可以把你的 Linux PC 連接到一台 PPP 伺服器上並存取該伺服器所連接的網路資源（幾乎）就如同你是直接連接在該網路上一般．

你也可以把你的 Linux PC 設爲一台 PPP 伺服器，這樣一來其它電腦就可以撥入你的電腦並且存取在你區域網路裡的資源．

因爲 PPP 是一種點對點 (peer-to-peer) 的系統，因此你也可以使用兩台 Linux PC 上的 PPP 把網路連結在一起（或是把區域網路連結到網際網路上）．

最主要的差異當然是速度 — 標準的乙太網路連線是以 10 Mbps（每秒百萬位元）這個理論上的最大貫通量在動作， 而在數據機這方面最大是以 56 kbps（每秒千位元）的速度在動作．

同時，依據 PPP 連線的型態，某些應用以及服務在使用上可能會有些限制．

1.1 客户端以及伺服端

PPP 是一種完完全全是點對點的協定；撥接的機器以及接受撥接的機器之間（在技術上）並沒有差異． 然而，爲了清楚明白的緣故，以伺服端與客户端的方式來思考是很有用的．

當你撥入一個節點要建立 PPP 連線時，你是客户端．你所連線的那台機器是伺服端．

當你在設定一台 Linux 機器使其接收並處理撥入的 PPP 連線時，你正在設立一台 PPP 伺服端．

任何 Linux PC 都可以是 PPP 伺服端以及客户端 — 甚至於如果你有一個以上的串列埠（以及數據機，如果有必要）的話還可以同時扮演這兩種角色． 如同前面所說的，就 PPP 而言，一旦連線建立那麼客户端與伺服端之間並不眞的有什麼差異．

爲了清楚明白的關係，這份文件把啓始呼叫（即 "撥入"）的那台機器稱作客户端， 而把回應電話，核對撥入請求之驗認（利用使用者代號，密碼以及其它可能的機制）的那台機器稱作伺服端．

使用 PPP 做爲客户端把某個地方的一台或多台機器連結到網際網路上可能是大部份人都感興趣的一點 - 這是將他們的 Linux 機器作爲一客户端．

在這份文件中所描述的程序將會讓你能夠建立並自動化你的網際網路連線．

這份文件也將在把你的 Linux PC 設爲 PPP 伺服器以及使用 PPP 來連結兩個區域網路（有完整的遞送設定）這些方面給你指引（這常常被稱作建立廣域網路連結）．

1.2 不同 Linux 套件間的差異

有這多不同的 Linux 套件，他們有自己的特質與行爲模式．

特別地，Linux (以及 Unix)電腦有兩種不同的起始方式，界面設定等等．

這兩種分別是 **BSD system initialisation** 與 **System V system initialisation**． 如果你進入某些 Unix 的新聞討論群，你會發現經常有雙方支持者的戰爭． 如果你覺得有趣，就加入這個浪費網路頻寬的行列!

最普遍使用的套件可能是

- Slackware
  使用 BSD 形式的系統起始
- Red Hat (and its former associate Caldera)
  使用 SysV 系統起始(有些許的修改)
- Debian
  使用 SysV 系統起始

BSD 形式的系統起始通常將它的啓動檔案放在 `/etc/...`，這些檔案是:

```
        /etc/rc
        /etc/rc.local
        /etc/rc.serial
                (也可能有其它檔案)
```

最近，一些 BSD 系統的起始模式將使用 `/etc/rc.d...` 做爲啓動檔案的存放目錄，而不是將所有的東西都丟到 `/etc` 下．

System V 起始模式將啓動檔案存放在 `/etc/...` 或 `/etc/rc.d/...` 及其下的一堆子目錄中:

```
drwxr-xr-x   2 root     root         1024 Jul  6 15:12 init.d
-rwxr-xr-x   1 root     root         1776 Feb  9 05:01 rc
-rwxr-xr-x   1 root     root          820 Jan  2  1996 rc.local
-rwxr-xr-x   1 root     root         2567 Jul  5 20:30 rc.sysinit
```

```
drwxr-xr-x   2 root     root         1024 Jul  6 15:12 rc0.d
drwxr-xr-x   2 root     root         1024 Jul  6 15:12 rc1.d
drwxr-xr-x   2 root     root         1024 Jul  6 15:12 rc2.d
drwxr-xr-x   2 root     root         1024 Jul 18 18:07 rc3.d
drwxr-xr-x   2 root     root         1024 May 27  1995 rc4.d
drwxr-xr-x   2 root     root         1024 Jul  6 15:12 rc5.d
drwxr-xr-x   2 root     root         1024 Jul  6 15:12 rc6.d
```

如果你嘗試尋找你的乙太網路卡及網路遞送設定眞正的配置地方，你需要追蹤到這些檔案裡找出配置指令的所在．

1.3 套件相關的 PPP 配置工具

在某些安裝套件(如 Red Hat 與 Caldera)，有一個以 X window 設定的 PPP 撥接系統． 這份 HOWTO 並未包含這種與最件相關的工具．如果你在使用上有問題，直接和套件的發行人詢問．

對於 Red Hat 4.x 的使用者，可以在 Linux 的資源區找到 *Red Hat PPP-TIP* <http://www.interweft.com.au>， 也可在支援區找到 *Red Hat Software* <http://www.RedHat.com>．

2 IP 號碼

每一個連上網際網路的設備都必須擁有它們自己的，唯一的 IP 號碼．這些是由每個國家的有關單位集中指定的．

如果你要把區域網路連上網際網路，在你的區域網路裡你所擁有的所有電腦及設備都必須使用從指定給你的網路範圍裡分配的 IP 號碼． 即使連接另一個區域網路（在網際網路之外）你也絕不能憑空取用 IP 號碼． 如果你'偷來'的 IP 號碼開始干擾其它已經使用這個 IP 的電腦的話，最糟時不僅不會動，還將會引起全體性的大破壞．

請注意在這整份文件裡所使用的 IP 號碼（有一些個例外）都是從保留給沒有（不曾）連上網際網路的網路所使用的‘未連線網路號碼’系列來的．

有一些特別留給不連接網際網路的網路使用的 IP 號碼．這些 IP 號碼依序是:

- 1 個 A-等級 的位址
  10.0.0.0 (netmask 255.0.0.0)
- 16 個 B-等級 的位址
  172.16.0.0 - 172.31.0.0 (netmask 255.255.0.0)
- 256 個 C-等級 的位址
  192.168.0.0 - 192.168.255.0 (netmask 255.255.255.0)

如果你的網路還沒從你國家的權責單位分配到 IP 號碼，那麼你的機器應該使用前面所列這些網路號碼中的一個．

這些號碼永遠不該使用在網際網路上．

然而，它們仍可使用在一部連接在網際網路的機器上的內部乙太網路卡上． 因爲 IP 號碼是分配給網路界面，而不是電腦本身的． 因此雖然你的乙太網路界面使用 10.0.0.1 (例如)，當你以 PPP 連上網際

網路時，你的 PPP 界面將由伺服器分配給另一個(合法的) IP 號碼． 你的 PC 將有網際網路的連接能力，但其它區域網路內的電腦則沒有．

過，使用 Linux 以及 ipfwadm 軟體的 IP 偽裝能力(也叫做 NAT - Network address Translation 網路位置轉換)， 你就可以把你的區域網路上不具合法 IP 號碼的機器連上網際網路（提供的服務會有些限制）．

想得到更多資訊請參考 *Linux IP Masquerade mini HOWTO* `<http://sunsite.unc.edu/mdw/HOWTO/mini/IP-Masquerade>`．

對於大部分的使用者，經由 PPP 連接單一機器到網際網路服務提供者(Internet Service Providers: ISP)， 取得一個 IP 號碼（或更正確的說，一個網路號碼）是沒有必要的．

如果你希望連接一個小型的區域網路到網際網路上，許多網際網路服務提供者可以從他們已有的 IP 位址空間中提供給你一個次網路（特定的一段 IP 號碼）． 或者，使用 IP 偽裝能力．

對於經由 ISP 連接單一 PC 到網際網路上的使用者，大部分的提供者使用動態的 IP 指定． 這是說，連線程序的一部份中，你所聯繫的 PPP 服務將會告訴你的機器這次連線的期間裡該 PPP 界面應該使用什麼 IP 號碼． 每次你連上 ISP 所得到的號碼不會一樣．

使用動態的 IP 號碼，你每次的連線不會都取得相同的 IP 號碼． 這對在你 Linux 機器上的伺服型態應用程式像是 sendmail, ftpd, httpd 以及諸如此類的軟體而言會產生問題． 這些服務根基於這部提供服務的電腦能以同樣的 IP 號碼取用．

由於動態的 IP 號碼指定而有所限制的服務（以及解決的方法，可能的話）稍後會在這份文件中討論．

3 這份文件的目的

3.1 設立 PPP 客户端

這份文件提供想要使用 Linux 以及 PPP 撥接到某一台 PPP 伺服器並且使用 PPP 設立 IP 連線的人們一些指引． 文件中假設 PPP 已經編譯好並且安裝在你的 Linux 機器上(但簡短地涵蓋重新配置重新編譯你的核心以便包含 PPP 支援的部份)．

雖然 DIP（建立 SLIP 連線的標準方式）可以用來創造 PPP 連線，但 DIP 指令稿一般都相當地複雜．因爲這個緣故，這份文件並不涵蓋使用 DIP 來建立 PPP 連線的部份．

取而代之的是，這份文件描述標準的 Linux PPP 軟體(chat/pppd)．

3.2 使用 PPP 連結兩個區域網路或連結一個區域網路到網際網路上

這份文件提供關於連結兩個區域網路或連結一個區域網路到網際網路上的（基本）資訊．

3.3 設立 PPP 伺服器

這份文件提供如何配置你的 Linux PC 做爲 PPP 伺服器的指引（允許其他人撥接到你的 Linux PC 並且建立 PPP 連線）．

你應該注意的是設定 Linux 作爲 PPP 伺服器的方法有無數種．這份文件（目前）給你一種方式 — 作者用來它設立一台小型 PPP 伺服器（共有 16 台數據機）．

這個方式已知運作良好．不過，它不一定是最好的方式．

3.4 使用 PPP 於直接纜線(direct null modem)連結

本文件提供使用 PPP 連結於兩部 Linux PCs 於 null modem 纜線的簡短描述． 以此法連結其它作業系統於 Linux 也是可能的． 如果要這樣做，你必須參考你要連結的作業系統的文件．

3.5 這份文件目前尚未涵蓋的...

- 編譯 PPP 軟體
  參見隨附你使用的 pppd 版本而來的文件．
- 連接以及配置數據機到 Linux 上(細節)
  請參閱 Serial-HOWTO，以及數據機相關的起始設定，參閱 *Modem Setup Information* `<http://www.in.net/info/modems/index.html>` 裡的資訊可能對你設定數據機有所幫助．
- 使用 DIP 創造 PPP 連線
  使用 chat 取代...
- 使用 socks 或是 IP Masquerade
  已經有涵蓋這兩個套件的絕佳文件．
- 使用 `diald` 以建立自動連線
  請參閱 `diald` 文件裡的資訊．
- 使用 EQL 將兩部數據機合成單一 PPP 連結．
- 與套件相關的 PPP 連線方法(例如 Red Hat 4.x 網路設定工具)
  請參閱該套件中的文件．
- 不斷增加的自動建立 PPP 工具
  請參閱適當的文件．

4 涵蓋的軟體版本

這份使用說明假設你使用 Linux 1.2.x 核心配合 PPP 2.1.2 軟體或 Linux 1.3.X/2.0.x 以及 PPP 2.2 版．

在本文件撰寫之時，Linux 的 PPP 軟體最新正式版本是 ppp-2.2f． 最新的版本(ppp-2.3)還在 beta 階段．

使用 PPP 2.2.0 配合核心 1.2.13 是有可能的．然而，要這樣做需要核心修補． 建議核心 1.2.13 版本的使用者升級到 ppp-2.2 因為它包含了許多錯誤修正及功能加強．

還有，你應特別注意不能將 **PPP 2.1.2** 與 **Linux** 核心 **2.0.X** 合用．

請注意這份文件不包括使用 Linux 核心 2.0.x 的可載入模組所引起的問題． 請參閱 kerneld mini-HOWTO 以及核心模組 2.0.x 的文件（在 Linux 2.0.x 的原始程式目錄 `/usr/src/linux/Documentation/...` 裡）．

因為這份文件是設計來協助新手的，強烈建議你使用已知能穩定地共同運作的 **Linux** 版本及適當的 **PPP** 版本．

5 其它有用的重要的文件

鼓勵使用者閱讀:-

- 隨附於 PPP 套件的文件;
- pppd 以及 chat 的線上使用手冊;
(使用 `man chat` 以及 `man pppd` 來閱讀)
- Linux 網路管理指引 (NAG);
參閱 *The Network Administrators' Guide* <http://sunsite.unc.edu/mdw/LDP-books/nag-1.0/nag.html>
- Net-2/3 HOWTO;
參閱 *Linux NET-2/3-HOWTO* <http://sunsite.unc.edu/mdw/HOWTO/NET-2-HOWTO.html>
- 在 `/usr/src/linux/Documentation` 裡的 Linux 核心文件;
- 數據機設定資訊 - 參閱 *Modem Setup Information* <http://www.in.net/info/modems/index.html>
- 由 O'Reilly and Associates 出版的最佳 Unix/Linux 書籍（參閱 *O'Reilly and Associates On-Line Catalogue* <http://www.ora.com/> 裡的目錄）· 如果你是 Unix/Linux 的新手，立刻跑（不要用走的）到離你最近的電腦書店去投資幾樣這些資料·
- 由 Al Longyear 所維護的 PPP-FAQ，可在 *Linux PPP-FAQ* <ftp://sunsite.unc.edu/pub/Linux/docs/faqs> 取得·
裡面包括了大量有用的資訊以問答的形式出現，在試圖找出為何 PPP 不能(正常)工作時相當有用·
- 不斷新增由不同出版社與作者出品的書籍;
鼓勵你去查閱這些流通的書籍·Linux 的發展與套件的演進相當快速，而書籍的改版速度(一般來說)相當的慢! 買一本已經過期的好書(現在有許多)可能使得新使用者感到困惑與失望·

Linux 文件的最佳起點就是 *The Linux Documentation Project Home Page* <http://sunsite.unc.edu/mdw/>. HOWTOs 傾向於定期地更新·

雖然即使不必閱讀這些資料中的任何一份你也可以使用這份文件來建立你的 PPP 連結， 但是如果你去閱讀這些文件那麼你對於你所做的究竟發生了些什麼將會了解更多! 你將能夠精確的提出自己的問題(或至少在 comp.os.linux... 新聞討論群或 Linux mailing lists 中問出較有深度的問題)·

這些文件（與各種其它文件，包括相關的 RFCs 等）提供額外的以及比這份使用說明所能包含的更多細節解釋·

如果你要使用 PPP 連接區域網路到網際網路上，那麼你將得知道一些關於 TCP/IP 網路運作的知識· 補充一點這份文件前面提到的資料，你將會發現 O'Reilly 出版的 "TCP/IP Network Administration" 以及 "Building Internet Firewalls" 很有好處!

5.1 有用的 Linux 電子郵遞列表

有許多在不同使用者等級之間流通的電子郵遞列表(mailing lists)· 選擇你有興趣以及對你專業知識及觀點有所幫助的列表來訂閱·

一些忠告: 有些列表是給那些"專家級"使用者或是特殊主題的． 雖然沒有人會抱怨你"潛伏"(訂閱但不發表文章)在裡面， 不過如果你在不適合的列表中提出"新手"的問題你可能會收到一些憤怒的意見．

這並不是因爲專家級的使用者討厭新進的使用者，而是因爲這些列表是用來處理具有一定困難度的特殊主題的．

入那些可以公開訂閱的列表，但請局限你的意見於列表的相關主題上面．

Linux 電子郵遞列表一個很好的起點是 *Linux Mailing List Directory* <http://summer.snu.ac.kr/~djshin/linux/mail-list/index.shtml>．

6 將 PPP 設定爲客户端的工作概要

這份文件包含了許多資訊 - 而且每次改版又不斷增加!

因此，本節的目的在於提供一簡短的概要說明要將你的 Linux 系統作爲客户端連接到 PPP 伺服器所需的工作．

6.1 取得/安裝軟體

如果你的 Linux 套件並未包含 PPP 軟體，你可從此取得: *the Linux PPP daemon* <ftp://sunsite.unc.edu/pub/Linux/system/network/serial/ppp/ppp-2.2.0g.tar.gz>.

這是本文件撰寫的時候最新正式版本．然而，儘量從該站取的最新的版本(本文件撰寫時 ppp-2.3 尚在 beta 階段而且應該很快會發行)．

這份 PPP 套件包含了如何編譯及安裝所需的指引，因此本文並不包含此部份!

6.2 編譯核心加入 PPP 支援

Linux 的 PPP 運作包含兩部份

- 上面提到的 PPP daemon
- PPP 的核心支援

許多套件似乎在它們預設的安裝核心中提供 PPP 的支援，但其餘則否．

如果在啓動時核心報告這樣的訊息

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

你的核心已編譯 PPP 支援在内．

那就是說，不論你使用什麼套件你可能希望自行編譯核心以在你的特定硬體配置之下得到最有效的系統資源利用． 請記住核心不能被置換出記憶體因此儘量讓核心保持最小在記憶體有限的機器上相當有用．

本文件在 7 (配置你的 Linux 核心) 一節提供了重新編譯核心的最少說明．

細節部份請參閱 *The Linux Kernel HOWTO* <Kernel-HOWTO.html>．

6.3 從你的 ISP 取得資訊

幾乎有無數種設定 PPP 伺服器的方法． 爲了連上你的 ISP, 你必須取得 PPP 伺服器如何運作的資訊．

爲你用的是 Linux, 在一些僅知道 MS Windows 客户端的 ISP 諮詢部門可能會遭遇困難．

然而，愈來愈多的 ISP 使用 Linux 提供服務 - 而且 Linux 也滲入公司環境，如果你眞的遇到問題你也可能會很幸運．

8 (取得你所需的 PPP 伺服器資訊) 一節告訴你需要知道你連接的 PPP 伺服器的什麼資訊 - 以及如何找出這些資訊．

6.4 設定你的數據機及串列埠

爲了連上 PPP 伺服器並取得最佳的資料傳輸速率，你的數據機必須設定正確．

同樣地，連接數據機及電腦的串列埠也必須正確地設定．

9 (設定你的數據機及串列埠) 一節提供這方面的資訊．

6.5 建立名稱到位址解析(DNS)

除了執行 PPP 及自動簽入 PPP 伺服器的檔案之外，還有一些文字配置檔案必須設定， 以便讓你的電腦能將像這樣的名稱 `www.interweft.com.au` 解析爲 IP 位址以便眞正與那台電腦連線． 這些檔案是:-

- `/etc/resolv.conf`
- `/etc/host.conf`

10 (建立名稱到位址解析) 一節提供這方面的設定細節．

特別地，你並不需要在你的 Linux PC 上執行名稱伺服器以連接到網際網路上(雖然或許你會想這麼做)． 你所需的只是知道至少一部你能使用的名稱伺服器 IP 位址(最好是你的 ISP 所提供的)．

6.6 PPP 與 root 權限

因爲在你的 Linux 電腦與另一個 PPP 伺服器建立連結需要操作網路界面(PPP 界面是一個網路界面)與核心的路由表，pppd 需要使用 root 的權限．

細節部份，參閱 11 (使用 PPP 與 root 權限) 一節．

6.7 檢查你的 PPP 套件檔案及設定 PPP 選項

有好幾個配置及撥號檔案需要建立以便讓 PPP 運作．在 PPP 套件中有一些範例，而本節告訴你應有的這些檔案是什麼:-

```
/etc/ppp/options
/etc/ppp/scripts/ppp-on
/etc/ppp/scripts/ppp-on-dialer
/etc/ppp/options.tpl
```

看你想用 PPP 達到的目標是什麼你可能需要一些額外的檔案:-

```
/etc/ppp/options.ttyXX
/etc/ppp/ip-up
/etc/ppp/pap-secrets
/etc/ppp/chap-secrets
```

除此之外，PPP daemon 有大量的線上選項，正確地使用是很重要的; 因此本節帶你看這些標準 PPP 選項並幫助你選擇你應使用的選項·

細節部份，請參閱 12 (設定 PPP 連接檔案) 一節·

6.8 如果你的 PPP 伺服器使用 PAP 認證(Password Authentication Protocol)

許多 ISP 設立 PPP 伺服器使用 PAP 認證·如果你的伺服器並不使用 PAP (如果你可以用手動簽入並收到標準的使用者名稱/密碼提示符號那麼它並未使用 PAP)，那麼你可以跳過此節·

取代在進入伺服器時出現提示符號輸入使用者名稱及密碼的是，使用 PAP 的 PPP 伺服器並不需要文字模式的簽入方式·

使用者認證是連接控制協定(link control protocol, LCP)的一部份，在建立 PPP 連線之初期交換的資訊·

13 (如果你的 PPP 伺服器使用 PAP 認證(Password Authentication Protocol)) 一節提供你使用 PAP 建立 PPP 連結所需檔案的資訊·

6.9 以手動方式連接至 PPP 伺服器

已經建立完基本的檔案之後，透過手動(使用 minicom 或 seyon)的方式連接及啓動你 Linux PC 的 pppd 來測試是個不錯的主意·

參見 14 (以手動方式建立 PPP 連結) 說明設定的細節部份·

6.10 使你的 PPP 連結自動化

一旦你成功地用手動方式簽入，你可以試著建立一指令稿以自動建立此連結·

15 (使你的連結自動化 - 創造連結指令稿) 一節包含了建立所需的指令稿，以及相當多的注意力集中在 chat 命令於 PPP 伺服器的簽入過程·

這節也討論了使用指令稿於 PAP/CHAP 認證的伺服器上·

6.11 結束連結

一旦你的連結能夠建立並正常工作，你必需能夠結束連結·

這在 17 (結束 PPP 連結) 一節說明·

6.12 如果你遭遇問題

許多人無法讓 PPP 正常地工作．PPP 伺服器的種類以及和它們建立連線的方法有非常多種． 類似地，PPP 有許多的選項 - 而有些選項是不能一起用的．

除了簽入並起始 PPP 服務的問題之外，數據機本身及電話線路也可能會有問題．

18 (解決問題) 一節提供一般錯誤的基本資訊，以及如何分離並解決的方法．

這裡提供的不過是基本的資訊．Al Longyear 所維護的 PPP-FAQ 包含了更多這一方面的資訊!

6.13 建立連結之後

一旦 PPP 連結可以運作了(特別地，一旦 IP 層可以運作)，Linux 的 PPP 可以自動執行(以 root 身份)一指令稿完成任何你想達成的功能．

23 (建立連結之後) 一節提供 /etc/ppp/ip-up 指令稿的資訊， 它如何接收 PPP 傳來的參數以及如何利用它來做一些像自 ISP 取回你的電子郵件， 送出在你機器佇列中存放的郵件等等工作．

6.14 動態 IP 連接提供標準 IP 服務的問題

就像在簡介一節裡指出的，動態 IP 號碼影響你的 Linux PC 作爲網際網路伺服器的能力．

21 (動態 IP 連接提供標準 IP 服務的問題) 一節提供(主要)受影響的服務以及你所能做(如果有的話)的克服辦法．

7 配置你的 Linux 核心

爲了要能夠使用 PPP，你編譯的 Linux 核心必須包含 PPP 的支援． 如果你還沒有 Linux 的原始程式碼請先取得 — 在 Linux 的檔案系統標準中它是放在 /usr/src/linux 裡面．

檢查這個目錄 - 許多的 Linux 套件會在安裝過程中安裝此原始程式碼(檔案及次目錄)．

在啓動時，你的 Linux 核心會印出大量的訊息．如果核心支援 PPP 的話會包含此一訊息． 要觀看這些訊息的話，看看你的系統記錄檔或用 dmesg | less 指令將這些訊息顯示在螢幕上． 如果核心支援 PPP 的話你將會看到像這樣的幾行

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

(這是 Linux 2.0.x 系列核心的訊息)．

Linux 核心的原始程式可以從 sunsite.unc.edu 或其映射節點取得．

7.1 安裝 Linux 核心原始程式碼

下面是簡短的取得及安裝 Linux 核心原始程式碼的指示．詳細的資訊可以在 *The Linux Kernel HOWTO* <Kernel-HOWTO.html> 中找到．

爲了安裝及編譯核心，你必須以 root 的身份簽入．

1. 切換到 /usr/src 目錄下
   cd /usr/src
2. 檢查 /usr/src/linux 看看你是否已經安裝了原始程式碼．
3. 如果沒有的話，由 *Linux kernel source directory* <ftp://sunsite.unc.edu/pub/Linux/kernel/v2.0> 或離你最近的映射節點取回．
   如果你想找更舊版本的核心(例如 1.2.X)，可到 *Old Linux kernel source directory* <ftp://sunsite.unc.edu/pub/Linux/kernel/old> 找找．
4. 選擇適當的核心 - 通常是最近的版本．取回後將此 'tar' 壓縮檔放在 /usr/src．
   注意: 一個 'tar' 檔案是一種壓縮檔 - 可能壓了許多檔案和目錄在裡頭．它相當於 DOS 下的 zip 檔案．
5. 如果你已經安裝了原始程式，但想要升級到新版的核心，你必須先移除舊的程式碼．用這個指令
   rm -rf /usr/src/linux
6. 用這個指令解壓縮
   tar xzf linux-2.0.XX.tar.gz
7. 然後，cd /usr/src/linux 並看看 README 檔．裡頭包含了完整的解釋如何配置及編譯新核心．仔細閱讀這個檔案(將它印出來放在手邊是個好主意，直到你已做了足夠多次知道怎麼辦)．

7.2 瞭解你的硬體

如果你要重新編譯你的核心你必須知道在你 PC 裡面的界面卡設備!! 對於某些設備（像是音效卡）你也得知道各種設定（像是硬體中斷號碼，輸出入位址以及諸如此類的資訊）．

7.3 編譯核心 － Linux 1.2.13 核心

要開始配置程序，首先依照在解說檔裡的說明適當地安裝原始程式碼．然後以下面的指令開始核心的配置程序

```
make config
```

為了要能夠使用 PPP，你必須配置核心包含 PPP 的支援（使用 PPP 需要 pppd 以及核心的 PPP 支援兩者）．

```
PPP (point-to-point) support (CONFIG_PPP) [n] y
```

根據在你 PC 裡面的硬體以及你想要的 Linux 作業系統特色回答其它的配置問題．然後繼續依照該解說檔來編譯並安裝你的新核心．

核心 1.2.13 只要製造四個 PPP 裝置．對於多埠卡(multi-port serial cards)來說，你必須修改核心 PPP 的原始程式式以便得到更多埠的支援．(參見隨附 PPP-2.1.2 套件而來的 README.linux 檔案說明你需要做的細節部份)．

注意: 核心 1.2.13 的配置對話不允許你回頭 - 因此如果你在 make config 對話中答錯問題，用 CTRL-C 跳出來重新開始．

7.4 編譯核心 - Linux 1.3.x 及 2.0.x 核心

對於 Linux 2.0.x 的核心，你可以使用類似 Linux 1.2.13 的程序． 再一次，依照在解說檔裡的說明適當地安裝原始程式碼．然後以下面的指令開始核心的配置程序

```
make config
```

然而，你也有另外的選擇

```
make menuconfig
```

這提供一個以選單爲基礎配合線上輔助的配置系統，允許你在配置程序中隨意所之．

也有一個以 X windows 爲基礎的配置界面

```
make xconfig
```

你可以直接編譯 PPP 的支援到核心裡去或者是做成可載入模組．

如果你只在你 Linux 機器運作的一部份時間裡使用 PPP 的話，那麼建議你把 PPP 的支援編譯爲可載入模組． 使用 'kerneld'，你的核心將會在你開始你的 PPP 連結程序時自動載入提供 PPP 支援所需的模組． 這節省了寶貴的記憶體空間（因爲核心沒有任何一個部份可以被置換到記憶體外，但可載入模組在未使用時會自動被移除）．

要這樣做的話，你需要打開可載入模組的支援:

```
        Enable loadable module support (CONFIG_MODULES) [Y/n/?] y
```

要加入 PPP 的核心支援，回答下列這個問題:-

```
        PPP (point-to-point) support (CONFIG_PPP) [M/n/y/?]
```

要做成可載入模組的話，回答 **M** 即可，否則就編譯成核心的一部份，回答 **Y** 就是．

與 1.2.13 的核心不同的是，2.0.x 的核心視需要自動建立 PPP 設備並且一點都不需要去研究原始程式碼以增加可用的 PPP 設備．

7.5 PPP-2.2 與 /proc/net/dev 的注意事項

如果你在使用 PPP-2.2，你會發現建立 PPP 裝置的一個副作用就是如果你觀察 /proc/net 檔案系統看不到 PPP 的裝置， 直到開始使用 pppd 裝置才會被建立起來:-

```
[hartr@archenland hartr]$ cat /proc/net/dev
Inter-|   Receive                      |  Transmit
 face |packets errs drop fifo frame|packets errs drop fifo colls carrier
    lo:  92792    0    0    0    0    92792    0    0    0     0     0
  eth0: 621737   13   13    0   23   501621    0    0    0  1309     0
```

一旦你有一個(或更多) PPP 服務啓動後，你會看到像這樣的項目(從一部 PPP 伺服器上取得):-

```
[root@kepler /root]# cat /proc/net/dev
Inter-|   Receive                  |  Transmit
 face |packets errs drop fifo frame|packets errs drop fifo colls carrier
    lo: 428021    0    0    0    0   428021    0    0    0     0    0
  eth0:4788257  648  648  319  650  1423836    0    0    0  4623    5
  ppp0:   2103    3    3    0    0     2017    0    0    0     0    0
  ppp1:  10008    0    0    0    0     8782    0    0    0     0    0
  ppp2:    305    0    0    0    0      297    0    0    0     0    0
  ppp3:   6720    7    7    0    0     7498    0    0    0     0    0
  ppp4: 118231  725  725    0    0   117791    0    0    0     0    0
  ppp5:  38915    5    5    0    0    28309    0    0    0     0    0
```

7.6 配置核心的一般考量

如果你要把你的 Linux PC 設爲一台 PPP 伺服器的話，那麼你必須把 IP forwarding 的支援編譯進去．如果你想使用 Linux 連接兩個區域網路或將區域網路連接到網際網路上的話這個選項也是需要的．

如果你要連接一個區域網路到網際網路上（或者甚至只是連接兩個區域網路），你應該要關心安全上的問題．在核心中加入 IP firewalls 的支援可能是必須的!

如果你想要用 IP 僞裝連接使用前述任何一個‘未連線’IP 網路號碼的區域網路那麼你也需要這個配置．

要使用 IP 僞裝與 IP firewalling，你必須在 make config 過程中的第一個問題上回答 yes:-

```
Prompt for development and/or incomplete code/drivers (CONFIG_EXPERIMENTAL)?
```

雖然這聽起來對新使用者有些困難，許多人仍愉快地使用 Linux 2.0.X 核心的 IP 僞裝與 IP firewalling 功能而沒有問題．

一旦你完成新核心的安裝以及重新啓動之後，你就可以開始配置並測試你的 PPP 連結．

8 取得你所需的 PPP 伺服器資訊

在你可以建立到某台伺服器的 PPP 連線之前，你需要取得下列的資訊（從 PPP 伺服器的系統管理使用者支援人員取得）:-

- 撥接服務的電話號碼
  如果你是在私用交換機(PABX)系統下，你還需要撥出外線信號的私用交換機號碼 — 通常是 0 或 9．
- 伺服器使用動態或靜態的 IP 號碼?
  如果伺服器使用靜態的 IP 號碼，那麼你需要知道在 PPP 連線中你這端要使用哪個 IP 號碼． 如果你的 ISP 提供你一合法 IP 號碼的次網路，你需要知道你能使用的 IP 號碼及網路遮罩．

  大部分的網際網路服務提供者都使用動態的 IP 號碼．如前述，這對於你可以使用的服務會有些限制．

然而，即使你使用靜態的 IP 號碼，大部份的 PPP 伺服器也不允許(爲了安全理由)客户端自行指定 IP 號碼，因爲這有安全風險． 你仍然必須知道這些資訊．

- ISP 的領域名稱伺服器 IP 號碼是什麼?
  雖然只需要一個但最少應該得要有兩個．

  這裡可能會有一個問題．微軟的 Windows 95 PPP 設定允許讓 DNS 的地址在連線過程中傳到客户端． 因此你的 ISP (或公司的諮詢室)可能會告訴你不需要知道 DNS 伺服器的 IP 位址．

  對於 Linux 來說，你必須至少知道一個 DNS 的位址． 目前 Linux 的 PPP 實作中不允許在連立連線之時動態地指定 DNS 的 IP 號碼 - 很可能將來也不會．

  注意: 雖然，Linux (作爲 PPP 的客户端)不能從伺服器中接受 DNS 的位址，然而，作爲 PPP 伺服器時， 它可以用 pppd 的選項 `dns-addr` 爲客户端指定這個資訊．

- 該伺服器是否需要使用 PAP/CHAP?
  如果是這樣你需要知道你用來連線 "id" 以及 "secret" ．（這大概會是你的在這個 ISP 的使用者名稱以及密碼）．

- 伺服器會自動起動 PPP 或者在一旦你簽入之後需要發出任何指令來起動伺服端的 PPP?
  如果你必須下某個指令來起動 PPP 的話，它是什麼?

- 如果伺服器是微軟的 Windows NT 系統，它是否使用微軟的 PAP/CHAP 系統?
  許多公司的區域網路似乎是使用 Windows NT 的系統以增加安全性．

仔細地注意這些資訊 — 你馬上要使用它們!

9 設定你的數據機及串列埠

你應該確認你的數據機設定正確而且你知道它連接到哪個串列埠．

謹記:-

- DOS com1: = Linux /dev/cua0 (and /dev/ttyS0)
- DOS com2: = Linux /dev/cua1 (and /dev/ttyS1)
  等等

還要提醒你就是如果你有四個串列埠，標準 PC 上讓 com1 與 com3 共用 IRQ4 且讓 com2 與 com4 共用 IRQ3．

如果你有其它機裝置和串列埠共用一個 IRQ 的話可能會有問題． 你必須確定你的數據機串列埠擁有它自己，唯一的一個 IRQ．許多現在的串列埠卡(與品質較佳的主機板上的串列埠)允許你將串列埠上的 IRQ 移開．

如果你在用 Linux 2.0.X 核心，你可以用 `cat /proc/interrupts` 檢查使用中的 IRQ，你會看到像這樣的輸出結果

```
 0:    6766283   timer
 1:      91545   keyboard
 2:          0   cascade
 4:     156944 + serial
 7:     101764   WD8013
10:     134365 + BusLogic BT-958
13:          1   math error
15:    3671702 + serial
```

這裡顯示了一個串列埠在 IRQ4 (一個滑鼠)與一個串列埠在 IRQ15． (也有一串列埠在 com2, IRQ3 與 com4 是在 IRQ14，但並未使用，沒有顯示出來)．

請小心 - 如果你要調你的 IRQ 的話你要確實知道你在做些什麼! 你不只需要打開你的電腦外殼，將界面卡拔下並調整 jumper，你還需要知道那個佔用什麼 IRQ． 以我的例子來說，這完全是以 SCSI 爲主的 PC，因此我將主機板上的通常會佔用 IRQ14 與 15 的 IDE 界面關掉．

你還需要注意如果你的 PC 還跑了其它的作業系統，更動 IRQ 的位址可能使那個作業系統啓動不正常，或根本開不了機!

如果眞的將串列埠移到了非標準的 IRQ，那麼你需要告訴 Linux 每一個埠所用的 IRQ 位址． 這可以用 `setserial` 指令達成，而且最好將它放到 `rc.local`，或在 SysV 系統中由 `rc.local` 所呼叫的 `rc.serial` 裡面成爲開機過程的一部份． 以我上面所列的機器來說，用這樣的指令

```
/bin/setserial -b /dev/ttyS2 IRQ 11
/bin/setserial -b /dev/ttyS3 IRQ 15
```

然而，如果你使用由 `kerneld` 行程所負責的動態載入串列模組，你不能像這樣每次開機時設定一次 IRQ 後就把它忘了． 這是因爲如果串列模組被移除，Linux 就會忘了這些特殊的設定．

因此，如果你以動態方式載入串列模組，你必須要在每入載入模組時重新設定 IRQ 才行．

9.1 串列埠與速度相容性的注意事項

如果你使用高速(外接式)數據機(14,400 baud 或更高)，你的串列埠速度必須能夠處理這種數據機所產生的資料量，特在是當數據機在壓縮資料時．

你的串列埠需要一種新型的 UART (Universal Asynchronous Receiver Transmitter)，像是 16550(A)． 如果你在使用一部舊的機器(或舊的串列卡)，很可能串列埠上只有舊的 8250 UART，這在你使用高速數據機時可能會發生問題．

使用這個指令

```
setserial -a /dev/ttySx
```

讓 Linux 回報你使用的 UART 類型．如果你沒有 16550A 型的 UART，去買一塊新的串列卡(不到 $50)． 當你買一塊新卡時，確定你能調整上面的 IRQ!

注意: 最早版本的 16550 UART 晶片上有瑕疵．這點很快就被發現了而且也發行了新版的晶片 - 16550A UART． 但仍有少部份有瑕疵的晶片流入了市面．雖然不太可能剛好就讓你買到，但你還是應該看一下晶片類型是不是 16550A，特別是在一些較早期的串列卡上．

9.2 串列埠名稱

以前，Linux 用 `cuaX` 表示撥出的串列埠名而 `ttySx` 表示撥入的名稱．

核心 2.0.X 之後這樣的要求已經改變了，所以你應該可以用 `ttySx` 同時表示撥入及撥出的名稱． 據我了解 `cuaX` 的裝置名稱在未來版本的核心中可能會取消．

9.3 配置你的數據機

爲使用 PPP 你將得正確地配置你的數據機 — 要完成這件工作請閱讀你的數據機使用手冊! 大部分的數據機都有 PPP 所需求的的出廠預設選項． 最基本的配置是:-

- 硬體流量控制 (RTS/CTS) (&K3 在許多採用 Hayes 指令集的數據機)

你應該要研究的其它設定（使用標準 Hayes 指令）是:

- E1 開啓指令的本地回應 (chat 運作所需)
- Q0 回報執行結果代碼 (chat 運作所需)
- S0=0 關閉自動回應 (除非你想讓你的數據機接聽電話)
- &C1 只在連線之後偵測載波
- &S0 Data Set Ready (DSR)永遠設爲開啓
- (看情況) Data Terminal Ready

有一個站台提供你不斷增加的數據機款式的設定，可能會對你有所幫助，在 *Modem setup information* `<http://www.in.net/info/modems/index.html>`．

在你的電腦與數據機之間的數據機串列面界面如何運作也值得研究了解． 大部分現代的數據機都允許你以固定的速度使用串列界面，即使電話線路界面切換到最高的速度它跟遠端數據機都能夠處理．

這稱爲 split speed operation．如果你的數據機能支援這項功能，把數據機的串列界面鎖定在它的最高速度（通常是 115,200 baud，但對 14,400 數據機來說可能是 38,400 baud）．

使用你的通訊軟體（例如 minicom）找出關於你數據機的配置資訊並且設爲 PPP 所需的設定． 許多數據機會在 AT&V 指令的回應中回報它們目前的設定狀況，但是你應該查看一下你的數據機使用手冊．

如果你把設定全都弄亂了，那麼你可以藉由發出 AT&F 指令回到穩定狀態（通常可以） — 回到出廠設定． （在我遇過的大部分現代數據機中，出廠的設定包含所有使用 PPP 所需的設定 — 但是你應該做個檢查）．

一旦你找出了必要的設定字串就把它寫下來．然後你必須做個決定: 你可以將這些設定值存在數據機上的非揮發性記憶體中這樣可以用適當的 AT 指令就可叫出． 另一個做法是將這些正確的設定在 PPP 的撥號過程中傳入．

如果你只從 Linux 下使用數據機連接到 ISP 伺服器，最簡單的做法就是將這些設定值存入數據機上的非揮發性記憶體(non-volatile RAM)．

另一方面，如果你要還在其它的應用程式或作業系統上使用數據機，最保險的做法是在每次撥號時才將這些資訊傳入， 如此可以確保在每次撥號時數據機總是在正確的狀態．(記下這些設定字串還有個好處就是在數據機遺失了記憶體的內容時，這確實可能發生)．

9.4 串列流量控制注意事項

當資料在串列通訊線路上流動的時候，可能會發生資料到達的比電腦能處理的還要快這種情況（電腦可能忙著做其它的事 — 記得，Linux 是一個多使用者，多工的作業系統）· 爲了確保資料不會漏失（在緩衝區中的資料不會超載而因此漏失掉），需要某些控制資料流量的方法·

有兩種方法可以在串列線路上達成這個目的:-

- 使用硬體信號(Clear To Send/Request to Send - CTS/RTS)
- 使用軟體信號(control S and control Q, also known as XON/XOFF).

雖然後者用在終端機（文字）連結上可能很好，但是在 PPP 上的資料使用整個 8 bits 編碼空間 — 而且在資料中的某個地方相當可能存在會被轉成 control S 以及 control Q 的位元組· 所以，如果數據機設成使用軟體流量控制的話，那麼傳輸很容易被擾亂!

對於使用 PPP 的高速連結（使用 8 bits 資料編碼）來說硬體流量控制是很重要的，因此你必須使用硬體流量控制·

9.5 測試你的數據機的撥出功能

在你完成了串列埠與數據機的設定之時試著撥到你的 ISP 上看看能否連上以確定設定是否有誤·

使用你的終端機通訊軟體（像是 minicom），撥到你想使用 PPP 連線的那台 PPP 伺服器·

（注意: 在這個階段我們並不嘗試建立 PPP 連線 — 只是要證實我們擁有正確的電話號碼以及爲了簽入並且起動 PPP 而確實地找出伺服器究竟傳送給我們什麼訊息）·

在這個過程中，可以截取（記錄到一個檔案裡）整個簽入的過程或者是小心地（非常小心地）確實寫下遠端伺服器所給你提示輸入你的使用者名稱以及密碼的訊息（以及任何建立 PPP 連線需要下達的指令）·

如果你的伺服器使用 PAP，你可能不會看見簽入的提示符號，而是(以文字表示的)連結通訊協定(看起來像是垃圾)出現在你的螢幕上·

一些警告:-

- 某些伺服器相當聰明: 你可以用文字模式的使用者名稱/密碼方式或使用 PAP 簽入· 因此如果你的 ISP 或公司站台使用 PAP 但你沒有立刻在螢幕上看到垃圾，也並不表示你做錯了·
- 某些站台要求你先輸入一些起始的文字然後才啓動標準的 PAP 程序·
- 某些 PPP 伺服器是被動的 - 它們就坐在那裡等待而不送出任何資料，直到客户端撥入並送出合法的連結控制協定· 如果你要連接的 PPP 伺服器是被動模式的，你將不會看到任何垃圾!
- 某些伺服器在你按下 `ENTER` 前不會啓動 PPP - 因此如果你正確簽入但未看到垃圾的話可以試試!

這最少值得撥接兩次 — 某些伺服器會在每次你簽入時改變提示（例如依時間改變!）· 你的 Linux 機器在每次你撥入的時候要能夠辨認的兩個主要的提示是:-

- 要求你輸入使用者名稱的提示;
- 要求你輸入密碼的提示;

如果你必須發出某個指令來起動伺服端的 PPP，那麼你也需要找出一旦你簽入之後伺服器所給你輸入指令以啓動 PPP 的提示．

果你的伺服器自動起動 PPP 的話，一旦你簽入完成，你將開始見到螢幕上出現垃圾 — 這是 PPP 伺服端傳送給你的機器以起動並且配置 PPP 連線的資訊．

這看起來應該會是像這樣的東西:-

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

(而且它還會一直進來!)

在某些系統中 PPP 必須在伺服端上明確地加以起動．這通常是因爲該伺服器被設定成允許使用相同使用者名稱密碼配對來做 PPP 簽入及一般使用界面(shell) 的簽入． 如果是這樣，一旦你簽入就發出該指令．再一次，你將會見到 PPP 連線伺服端起動的垃圾資訊 — 所以現在你可以掛斷了．

如果你沒有在連上時立即在螢幕上看到這些垃圾(如果必要的話先簽入並啓動 PPP 伺服器)， 按下 **Enter** 試試能否啓動 PPP 伺服器...

這個時候，你就可以掛斷你的數據機（通常是，快速地鍵入 +++ 一旦你的數據機回應 OK 然後接著下 ATHO 指令）．

如果你不能使你的數據機運作的話，閱讀你的數據機使用手冊，你通訊軟體的線上使用手冊以及 Serial HOWTO! 一旦你解決該問題，繼續進行前面所說的．

10 建立名稱到位址解析(DNS)

雖然我們人類喜歡給事物取個名稱，電腦喜歡數字． 在 TCP/IP 網路上（這也就是網際網路），我們藉由特定的名稱呼叫某台機器，而且每台機器都存在於一個特定的領域標記裡面． 舉例而言，我的 Linux 工作站稱爲 **archenland** 而它是存在於 **interweft.com.au** 這個領域裡． 所以它在人們心目中的位址就是 **archenland.hedland.edu.au**． (稱爲 FQDN - fully qualified domain name．)

然而，爲了使網際網路上的其它電腦找的到這台機器，事實上電腦在網際網路上溝通時它是藉由其 IP 號碼來認知的．

把機器（以及領域）的名稱轉譯（解析）成爲在網際網路上所實際使用的 IP 號碼是提供領域名稱服務之機器的工作．

它是這樣子運作的:-

- 你的機器需要知道某部特定電腦的 IP 位址．需要這項資訊的應用程式向你的 Linux PC 上的'名稱解析器'詢問此資訊;
- 名稱解析器查詢機器內部的主機名稱檔案(可能是 `/etc/hosts` 以及/或是知道這項資訊的領域名稱伺服器，而名稱解析器實際的行爲由 `/etc/host.conf` 檔案決定);
- 如果答案在主機名稱檔案中，就以此答案回答;
- 如果是在指定的領域名稱伺服器中，你的 PC 就向該機器詢問;
- 如果這部領域名稱伺服器已經知道被詢問名稱的 IP 號碼，就以此回答．否則，它會透過網際網路向其它的名稱伺服器找尋資訊． 最後這部名稱伺服器會將資訊傳回給原來詢問的名稱解析器，然後傳給要求此資訊的應用程式．

當你創造出 PPP 連線時，你需要告訴你的 Linux 機器它可以在哪裡找到主機名稱對照到 IP 號碼（位址解析）的資訊，如此你就可以使用機器的名稱而你的電腦可以把這些名稱轉譯成它運作所需的 IP 號碼．

一個方法是輸入所有你想聯絡的主機到 /etc/hosts 檔案裡（如果你是連上網際網路的話這眞的是完全不可能的事）；另一個方法是使用相對於機器名稱的 IP 號碼（要記得全部的 IP 位址是不可能的除非在最小的區域網路上）．

最好的方法是設定你的 Linux 使它知道該到哪取得這項名稱到號碼的資訊 — 自動地．這項服務是由領域名稱伺服器系統所提供．需要做的全部工作是把 IP 號碼輸入到你的 /etc/resov.conf 檔案裡．

10.1 /etc/resolv.conf 檔案

你的 PPP 伺服器系統管理使用者支援人員應該要提供給你兩個 DNS 的 IP 號碼（只需要一個 — 但是兩個在出問題時可以多些幫助）．

如前所述，Linux 不能以像 MS Windows 95 一樣的方式來設定它的名稱伺服器的 IP 號碼．所以你必須(有禮貌地)堅持你的 ISP 提供你這項資訊!

你的 /etc/resolv.conf 看起來應該會像這樣:

```
domain your.isp.domain.name
nameserver 10.25.0.1
nameserver 10.25.1.2
```

編輯這個檔案（如果沒有的話就建一個新的）以呈現你的 ISP 提供的資訊．它的擁有者以及權限應該像下面這樣:-

```
-rw-r--r--   1 root     root           73 Feb 19 01:46 /etc/resolv.conf
```

如果你因爲本來就在區域網路上所以已經設立 /etc/resolv.conf 檔案的話，只要簡單的把 PPP 連線的 DNS 伺服器 IP 號碼加到你已有的檔案裡即可．

10.2 /etc/host.conf 檔案

你也應該檢查你的 /etc/host.conf 檔案是否設定正確．它看起來應該像這樣

```
order hosts,bind
multi on
```

它告訴你的名稱解析器在向名稱伺服器詢問之前先使用主機名稱檔案中的資訊．

11 使用 PPP 與 root 權限

因爲 PPP 需要設定網路設備，變更核心遞送表格以及諸如此類的動作，所以它需要以 root 的權限來做這些事．

如果 root 以外的使用者要能建立 PPP 連線，那麼 pppd 程式應該設爲以 root 的身分執行(setuid):-

```
-rwsr-xr-x   1 root     root        95225 Jul 11 00:27 /usr/sbin/pppd
```

如果 /usr/sbin/ppd 不是設定爲這樣，那麼以 **root** 的身分下這個指令:

```
chmod u+s /usr/sbin/pppd
```

這樣是讓 pppd 以 root 的權限執行，即使是由一般使用者所執行的． 這能讓一般使用者執行的 pppd 具有必要的權限建立網路界面及核心遞送表格．

以 root 的身分執行的程式在安全上是潛在的漏洞，所以你對於設定爲 'setuid' 的程式必須非常地小心． 許多地程式(包括 pppd)已經被小心地撰寫以將用 root 的身分執行的危險降到最低，所以這樣做應該是安全地(不過不保證)．

根據你希望你的系統如何運作而定，特別是如果你希望你系統裡的任何使用者都能啓始 PPP 連結，你應該把你的 ppp-on/off 指令稿設定爲全部的人都可以讀取執行．（如果你的 PC 只由你使用那麼這樣大概不錯）．

然而，如果你不希望任何人都能起始 PPP 連線（例如，你的孩子在你的 Linux PC 上有帳號而你不希望他們在沒有你的監督下連上網際網路），你將得建立一個 PPP 群組（編輯 /etc/group 檔案）並且:

- 將 pppd 設定爲以 root 的權限執行，擁有者是 root 而群組是 PPP，而其它的權限都關閉．它看起來應該像這樣

```
-rwsr-x---   1 root     PPP         95225 Jul 11 00:27 /usr/sbin/pppd
```

- 使 ppp-on/off 指令稿由使用者 root 以及群組 PPP 所擁有
- 使 ppp-of/off 指令稿能由群組 PPP 讀取執行

```
-rwxr-x---   1 root     PPP           587 Mar 14  1995 /usr/sbin/ppp-on
-rwxr-x---   1 root     PPP           631 Mar 14  1995 /usr/sbin/ppp-off
```

- 關閉其它的存取權限
- 把能夠起動 PPP 的使用者加入 /etc/group 檔案的 PPP 群組裡

即使如此，一般使用者仍然無法以軟體的方式中止連結． 執行 ppp-off 指令稿需要 root 的權限．然何，任何使用者都可以關掉數據機(或將電話線由內接式數據機拔下)．

另外一種(更好的)辦法，允許使用者使用 sudo 來起動 ppp 連結． 這樣可以提供更佳的安全性並且可以讓你設定讓任何(可信任的)使用者使用指令稿來啓動/結束連結． 使用 sudo 可以讓一位可信任的使用者乾淨而安全地啓動/結束 PPP 連結．

12 設定 PPP 連接檔案

你必須以 **root** 身份簽入來建立這些目錄並且編輯這些設立 PPP 連線所需的檔案，即使你想讓所有的使用者都能使用 PPP．

PPP 使用幾個檔案來建立並設定 PPP 連結．這些檔案在 PPP 2.1.2 與 2.2 中的名稱與位置都不同．

在 PPP 2.1.2 中這些檔案是:-

```
/usr/sbin/pppd               # PPP 執行檔
/usr/sbin/ppp-on             # 撥號/連線指令稿
/usr/sbin/ppp-off            # 斷線指令稿
/etc/ppp/options             # 所有連線所使用的選項
/etc/ppp/options.ttyXX   # 給某一特定通訊埠使用的選項
```

在 PPP 2.2 中這些檔案是:-

```
/usr/sbin/pppd                  # PPP 執行檔
/etc/ppp/scripts/ppp-on         # 撥號/連線指令稿
/etc/ppp/scripts/ppp-on-dialer  # 撥號的 chat 指令稿部份
/etc/ppp/scripts/ppp-off        # 斷線指令稿
/etc/ppp/options                # 所有連線所使用的選項
/etc/ppp/options.ttyXX          # 給某一特定通訊埠使用的選項
```

Red Hat Linux 的使用者應注意標準的 Red Hat 4.X 安裝將這些指令稿放在 /usr/doc/ppp-2.2.0f-2/scripts．

在你的 /etc 目錄裡應該要有個目錄:

```
drwxrwxr-x   2 root     root         1024 Oct  9 11:01 ppp
```

如果它不存在的話 — 以這樣的權限建立它．

如果這個目錄已經存在，它應該會包含一個稱爲 **options.tpl** 的選項檔案樣板．這個檔案包括在下面．

因爲它包含所有 PPP 選項的解釋所以請你把它印出來（配合 pppd 的線上使用手冊來閱讀將會是很有用的）． 雖然你可以使用這個檔案作爲 /etc/ppp/options 檔案的基礎，但是建立你自己的，沒有包含所有在這個樣板裡的指令的選項檔案可能會更好 — 它會短得多而且比較容易閱讀維護．

如果你有多個串列線路數據機（典型的例子是 PPP 伺服器），那麼建立一個一般化的 /etc/ppp/options 檔案，其中包含每個你提供支援撥入的串列埠所使用的共同選項並且爲每一個需要個別設定以建立 PPP 連線的串列線路設立個別的選項檔案．

這些檔案名爲 options.ttyx1, options.ttyx2 依此類推（其中 x 是你串列埠的適當代碼）．

然而，對於單一 PPP 連線，你可以直接使用 /etc/ppp/options 這個檔案．另外一種辦法，你可以把所有的選項放進 pppd 指令作爲參數．

使用 /etc/ppp/options.ttySx 檔案的設定方式會比較容易加以維護． 如果你使用 PPP 來連線到好幾個不同的節點去的話，那麼你就可以在 /etc/ppp/options.site 裡面爲每個節點建立選項檔案然後在你連線時指定選項檔案作爲 PPP 指令的參數． (使用 file option-file 參數於 pppd 的指令列)．

12.1 替代的 options.tpl 檔案

某些個 PPP 的發行套件似乎漏失了 options.tpl 這個檔案，所以在這裡有個完整的檔案． 我建議你不要直接編輯這個檔案來建立你自己的 /etc/ppp/options． 把它拷貝到一個新的檔案然後編輯該檔案比較好．如果你弄亂了你編輯的檔案，你可以回頭從原始檔案再度開始．

```
# /etc/ppp/options -*- sh -*- general options for pppd
# created 13-Jul-1995 jmk
# autodate: 01-Aug-1995
# autotime: 19:45

# Use the executable or shell command specified to set up the serial
# line.  This script would typically use the "chat" program to dial the
# modem and start the remote ppp session.
#connect "echo You need to install a connect command."

# Run the executable or shell command specified after pppd has
# terminated the link.  This script could, for example, issue commands
# to the modem to cause it to hang up if hardware modem control signals
# were not available.
#disconnect "chat -- \d+++\d\c OK ath0 OK"

# async character map -- 32-bit hex; each bit is a character
# that needs to be escaped for pppd to receive it.  0x00000001
# represents '\x01', and 0x80000000 represents '\x1f'.
#asyncmap 0

# Require the peer to authenticate itself before allowing network
# packets to be sent or received.
#auth

# Use hardware flow control (i.e. RTS/CTS) to control the flow of data
# on the serial port.
#crtscts

# Use software flow control (i.e. XON/XOFF) to control the flow of data
# on the serial port.
#xonxoff

# Add a default route to the system routing tables, using the peer as
# the gateway, when IPCP negotiation is successfully completed.  This
# entry is removed when the PPP connection is broken.
#defaultroute

# Specifies that certain characters should be escaped on transmission
# (regardless of whether the peer requests them to be escaped with its
# async control character map).  The characters to be escaped are
# specified as a list of hex numbers separated by commas.  Note that
# almost any character can be specified for the escape option, unlike
# the asyncmap option which only allows control characters to be
# specified.  The characters which may not be escaped are those with hex
# values 0x20 - 0x3f or 0x5e.
#escape 11,13,ff
```

```
# Don't use the modem control lines.
#local

# Specifies that pppd should use a UUCP-style lock on the serial device
# to ensure exclusive access to the device.
#lock

# Use the modem control lines.  On Ultrix, this option implies hardware
# flow control, as for the crtscts option.  (This option is not fully
# implemented.)
#modem

# Set the MRU [Maximum Receive Unit] value to <n> for negotiation.  pppd
# will ask the peer to send packets of no more than <n> bytes. The
# minimum MRU value is 128.  The default MRU value is 1500.  A value of
# 296 is recommended for slow links (40 bytes for TCP/IP header + 256
# bytes of data).
#mru 542

# Set the interface netmask to <n>, a 32 bit netmask in "decimal dot"
# notation (e.g. 255.255.255.0).
#netmask 255.255.255.0

# Disables the default behaviour when no local IP address is specified,
# which is to determine (if possible) the local IP address from the
# hostname. With this option, the peer will have to supply the local IP
# address during IPCP negotiation (unless it specified explicitly on the
# command line or in an options file).
#noipdefault

# Enables the "passive" option in the LCP.  With this option, pppd will
# attempt to initiate a connection; if no reply is received from the
# peer, pppd will then just wait passively for a valid LCP packet from
# the peer (instead of exiting, as it does without this option).
#passive

# With this option, pppd will not transmit LCP packets to initiate a
# connection until a valid LCP packet is received from the peer (as for
# the "passive" option with old versions of pppd).
#silent

# Don't request or allow negotiation of any options for LCP and IPCP
# (use default values).
#-all

# Disable Address/Control compression negotiation (use default, i.e.
```

```
# address/control field disabled).
#-ac

# Disable asyncmap negotiation (use the default asyncmap, i.e. escape
# all control characters).
#-am

# Don't fork to become a background process (otherwise pppd will do so
# if a serial device is specified).
#-detach

# Disable IP address negotiation (with this option, the remote IP
# address must be specified with an option on the command line or in an
# options file).
#-ip

# Disable magic number negotiation.  With this option, pppd cannot
# detect a looped-back line.
#-mn

# Disable MRU [Maximum Receive Unit] negotiation (use default, i.e.
# 1500).
#-mru

# Disable protocol field compression negotiation (use default, i.e.
# protocol field compression disabled).
#-pc

# Require the peer to authenticate itself using PAP.
# This requires TWO WAY authentication - do NOT use this for a standard
# PAP authenticated link to an ISP as this will require the ISP machine
# to authenticate itself to your machine (and it will not be able to).
#+pap

# Don't agree to authenticate using PAP.
#-pap

# Require the peer to authenticate itself using CHAP [Cryptographic
# Handshake Authentication Protocol] authentication.
# This requires TWO WAY authentication - do NOT use this for a standard
# CHAP authenticated link to an ISP as this will require the ISP machine
# to authenticate itself to your machine (and it will not be able to).
#+chap

# Don't agree to authenticate using CHAP.
#-chap
```

```
# Disable negotiation of Van Jacobson style IP header compression (use
# default, i.e. no compression).
#-vj

# Increase debugging level (same as -d).  If this option is given, pppd
# will log the contents of all control packets sent or received in a
# readable form.  The packets are logged through syslog with facility
# daemon and level debug. This information can be directed to a file by
# setting up /etc/syslog.conf appropriately (see syslog.conf(5)).  (If
# pppd is compiled with extra debugging enabled, it will log messages
# using facility local2 instead of daemon).
#debug

# Append the domain name <d> to the local host name for authentication
# purposes.  For example, if gethostname() returns the name porsche,
# but the fully qualified domain name is porsche.Quotron.COM, you would
# use the domain option to set the domain name to Quotron.COM.
#domain <d>

# Enable debugging code in the kernel-level PPP driver.  The argument n
# is a number which is the sum of the following values: 1 to enable
# general debug messages, 2 to request that the contents of received
# packets be printed, and 4 to request that the contents of transmitted
# packets be printed.
#kdebug n

# Set the MTU [Maximum Transmit Unit] value to <n>. Unless the peer
# requests a smaller value via MRU negotiation, pppd will request that
# the kernel networking code send data packets of no more than n bytes
# through the PPP network interface.
#mtu <n>

# Set the name of the local system for authentication purposes to <n>.
# This will probably have to be set to your ISP user name if you are
# using PAP/CHAP.
#name <n>

# Set the user name to use for authenticating this machine with the peer
# using PAP to <u>.
# Do NOT use this if you are using 'name' above!
#user <u>

# Enforce the use of the host name as the name of the local system for
# authentication purposes (overrides the name option).
#usehostname

# Set the assumed name of the remote system for authentication purposes
```

```
# to <n>.
#remotename <n>

# Add an entry to this system's ARP [Address Resolution Protocol]
# table with the IP address of the peer and the Ethernet address of this
# system.
#proxyarp

# Use the system password database for authenticating the peer using
# PAP.
#login

# If this option is given, pppd will send an LCP echo-request frame to
# the peer every n seconds. Under Linux, the echo-request is sent when
# no packets have been received from the peer for n seconds. Normally
# the peer should respond to the echo-request by sending an echo-reply.
# This option can be used with the lcp-echo-failure option to detect
# that the peer is no longer connected.
#lcp-echo-interval <n>

# If this option is given, pppd will presume the peer to be dead if n
# LCP echo-requests are sent without receiving a valid LCP echo-reply.
# If this happens, pppd will terminate the connection.  Use of this
# option requires a non-zero value for the lcp-echo-interval parameter.
# This option can be used to enable pppd to terminate after the physical
# connection has been broken (e.g., the modem has hung up) in
# situations where no hardware modem control lines are available.
#lcp-echo-failure <n>

# Set the LCP restart interval (retransmission timeout) to <n> seconds
# (default 3).
#lcp-restart <n>

# Set the maximum number of LCP terminate-request transmissions to <n>
# (default 3).
#lcp-max-terminate <n>

# Set the maximum number of LCP configure-request transmissions to <n>
# (default 10).
# Some PPP servers are slow to start up. You may need to increase this
# if you keep getting 'serial line looped back' errors and your are SURE
# that you have logged in correctly and PPP should be starting on the server.
#lcp-max-configure <n>

# Set the maximum number of LCP configure-NAKs returned before starting
# to send configure-Rejects instead to <n> (default 10).
#lcp-max-failure <n>
```

```
# Set the IPCP restart interval (retransmission timeout) to <n>
# seconds (default 3).
#ipcp-restart <n>

# Set the maximum number of IPCP terminate-request transmissions to <n>
# (default 3).
#ipcp-max-terminate <n>

# Set the maximum number of IPCP configure-request transmissions to <n>
# (default 10).
#ipcp-max-configure <n>

# Set the maximum number of IPCP configure-NAKs returned before starting
# to send configure-Rejects instead to <n> (default 10).
#ipcp-max-failure <n>

# Set the PAP restart interval (retransmission timeout) to <n> seconds
# (default 3).
#pap-restart <n>

# Set the maximum number of PAP authenticate-request transmissions to
# <n> (default 10).
#pap-max-authreq <n>

# Set the CHAP restart interval (retransmission timeout for
# challenges) to <n> seconds (default 3).
#chap-restart <n>

# Set the maximum number of CHAP challenge transmissions to <n>
# (default 10).
#chap-max-challenge

# If this option is given, pppd will re-challenge the peer every <n>
# seconds.
#chap-interval <n>

# With this option, pppd will accept the peer's idea of our local IP
# address, even if the local IP address was specified in an option.
#ipcp-accept-local

# With this option, pppd will accept the peer's idea of its (remote) IP
# address, even if the remote IP address was specified in an option.
#ipcp-accept-remote
```

12.2 我應該使用什麼選項? (無 PAP/CHAP)

嗯，完全視情況而定（唉）．這裡所提供的應該適用於大部份的伺服器．

而，如果它無法運作的話，閱讀樣板檔(/etc/ppp/options.tpl) 以及 pppd 的線上使用手冊並且告訴你所連線之伺服器的系統管理使用者支援人員．

你還應該注意這裡所展示的連結指令稿也使用了一些給 pppd 的命令列選項以便讓事情容易調整些．

```
# /etc/ppp/options (NO PAP/CHAP)
#
# 避免 pppd 進入背景執行
-detach
#
# 使用數據機控制線
modem
# 使用 uucp 形態的鎖定檔以避免它人取用串列裝置
lock
# 使用硬體流量控制
crtscts
# 在遞送表格中將此連結建立爲預設遞送裝置
defaultroute
# 不使用任何"逸出"控制序列
asyncmap 0
# 最大傳送封包大小爲 552 bytes
mtu 552
# 最大接收封包大小爲 552 bytes
mru 552
#
#-------END OF SAMPLE /etc/ppp/options (no PAP/CHAP)
```

13 如果你的 PPP 伺服器使用 PAP 認證(Password Authentication Protocol)

如果你連接的伺服器需要 PAP 或 CHAP 認證，你需要多一點的工作．

在上面的選項檔案中，加上下面幾行

```
#
# 告訴 pppd 使用你的 ISP 名稱做爲認證過程中的"主機名稱"
name <your ISP user name>       # 你需要改變這一行
#
# 如果你執行 PPP *伺服器* 並且需要使用 PAP 或 CHAP 認證，適當除去下面的
# 註解．不要在你做爲客户端連上伺服器時使用此選項(即使它使用 PAP 或 CHAP)
# 因爲這是告訴伺服器必須爲它自己在你的機器上提供認證(這幾乎是不可能的，
# 所以連結會失敗)．
#+chap
```

```
#+pap
#
# 如果你使用在 /etc/ppp/pap-secrets 檔案中經編碼的密碼，除去下面一行的註解．
# 注意：這和 Windows NT 上的遠端存取服務裡的微軟編碼密碼是不同的．
#+papcrypt
```

13.1 使用 MSCHAP

微軟的 Windows NT 遠端存取服務(RAS)可以被設定爲使用一種 CHAP(Challenge/Handshake Authentication Protocol) 的變種． 在你的 PPP 原始程式碼中，你可以找到一個叫 README.MSCHAP80 的檔案討論這件事．

你可以開啓 pppd 的偵錯功能判斷伺服器是否要求使用此認證協定． 如果伺服器使用 MS CHAP 認證，你會看到這樣的一行

```
rcvd [LCP ConfReq id=0x2 <asyncmap 0x0> <auth chap 80> <magic 0x46a3>]
```

關鍵的訊息是 **auth chap 80**．

爲了使用 MS CHAP，你必須重新編譯 pppd 以支援此協定．請參見 PPP 原始程式碼中的 README.MSCHAP80 說明如何編譯以使用此變種協定．

你還要注意這裡的程式碼只是讓 Linux PPP 做爲客户端連接上 MS Windows NT 伺服器用的． 它並不支援將 Linux 設立爲讓客户端使用 MSCHAP80 認證 PPP 伺服器．

13.2 PAP/CHAP 密碼檔案

如果你使用 PAP 或 CHAP 認證，你必須建立這些密碼檔案．它們是:

```
/etc/ppp/pap-secrets
/etc/ppp/chap-secrets
```

這些檔案擁有者必須爲 root，群組爲 root，同時爲了安全檔案權限應爲 740．

關於 PAP 與 CHAP 第一點要指出的是它們是設計用來認證電腦系統而不是使用者的．

"嗯? 有什麼不同呢?" 我聽到你在問．

哦，一旦你的電腦建立與伺服器建立 PPP 連線，你系統上的任何使用者都可以使用此連線 - 不只是你而已． 這也是爲何你可以使用 PPP 將兩個區域網路(LAN)連接起來建立一廣域網路(WAN)的原因．

PAP (還有 CHAP 也是)需要雙向認證 - 也就是每一電腦需要一個合法的名稱與密碼在另一電腦上． 然而，這並不是大部份提供撥接式 PAP 認證連接的 PPP 伺服器運作的方式．

也就是說，你的 ISP 可能會給你一個使用者名稱與密碼以允許你連接到他們的系統進而連接到網際網路上． 你的 ISP 對於你電腦的名稱一點也不感興趣，因此你可能必須使用你在 ISP 上的使用者名稱做爲你電腦的名稱．

這點要使用 pppd 選項的 name user name 來達成． 因此，如果你使用你 ISP 給你的使用者名稱，加上這一行

```
name your_user name_at_your_ISP
```

到你的 /etc/ppp/options 檔案中·

技術上來說，PAP 其實應該要使用 user our_user name_at_your_ISP 這個選項，但是如果使用到 PAP 的話那麼 pppd 聰明的足以把 name 解譯爲 user· 使用 name 這個選項的好處是這對於 CHAP 也是有效的·

因爲 PAP/CHAP 是用來驗認電腦的，在技術上你也需要指定遠端電腦的名稱· 然而，大部分的人只接一家 ISP 所以你可以在密碼檔裡的遠端主機名稱使用通用字元(*) ·

許多 ISP 有多個數據機撥接中心連接到不同的終端機伺服器 — 每個終端機伺服器都有不同的名稱，但都從單一的（自動轉線）號碼撥接· 在某些情形下可能因此而很不容易預先知道遠端電腦的名稱!

13.3 PAP 密碼檔

/etc/ppp/pap-secrets 這個檔案看起來像

```
# Secrets for authentication using PAP
# client        server      secret     acceptable_local_IP_addresses
```

這四個欄位是以空白分界而且最後一項可以是空白(如果你使用動態 IP 或由你的 ISP 所提供的靜態 IP)·

假定你的 ISP 給你的使用者名稱是 fred 而密碼是 flintstone 的話那麼你應該在 /etc/ppp/options[.ttySx] 裡設定 name fred 這個選項並且把你的 /etc/ppp/pap-secrets 檔案設成下面這樣

```
# Secrets for authentication using PAP
# client        server  secret          acceptable local IP addresses
fred            *       flintstone
```

這是說對於 fred 這個本地機器名稱（我們已經告訴過 pppd 使用這個名稱，即使這不是我們本地機器的名稱也一樣）以及任何的伺服器，都使用 flintstone 這個密碼（暗碼）·

注意到我們並不需要指定本地的 IP 位址，除非有要求要我們使用一個特定的，固定的 IP 位址· 即使你試著用一固定的 IP，這也不大可能行得通因爲對於大部份的 PPP 伺服器(爲了安全的理由)不允許遠端系統自行指定 IP 號碼·

13.4 CHAP 密碼檔

目前的 pppd 版本要求你要有互相驗認的方法 — 這是說你必須能夠讓從你的機器到遠端伺服器以及從遠端伺服器到你的機器這兩種驗認都能進行·

所以，如果你的機器是 fred 而遠端是 barney 的話，則在它們個別的 /etc/ppp/options.ttySx 裡你的設爲 name fred remotename barney 而遠端機器應設爲 name barney remotename fred ·

fred 這台機器的 /etc/ppp/chap-secrets 檔案看起來會像

```
# Secrets for authentication using CHAP
# client        server  secret          acceptable local IP addresses
fred            barney  flintstone
barney          fred    wilma
```

而 barney 的是

```
# Secrets for authentication using CHAP
# client        server  secret          acceptable local IP addresses
barney          fred    flintstone
fred            barney  wilma
```

特別要注意的是雙方機器都必須具有雙向認證的項目·這允許本地的機器向遠端機器認證它自己並且讓遠地機器向本地機器認證它自己·

13.5 處理多重 PAP 認證連接

一些使用者擁有多於一部使用 PAP 的伺服器可連線· 如果你的使用者名稱在每部你想連接的機器上都不同，這並不是問題·

然而，許多使用者在兩個系統(或者甚至是全部)上具有相同的使用者名稱· 這在從 /etc/ppp/pap-secrets 中正確地選擇適當一行時會有問題·

如你所料地，PPP 提供一種機制以克服此因難· PPP 允許你使用 **remotename** 選項爲遠端伺服器設定一個"假想名稱"·

假設你連接到兩部 PPP 伺服器都使用名稱 fred·你以此方式建立 /etc/ppp/pap-secrets 檔案

```
fred    pppserver1      barney
fred    pppserver2      wilma
```

如此，在與 pppserver1 連線時你在 pppd 選項中使用 `name fred remotename pppserver1` 並在連線 pppserver2 時使用 `name fred remotename pppserver2`·

因爲你可以使用在 pppd 使用 `file filename` 以指定 ppp 選項，你可以建立一指令稿以連接到每部 PPP 伺服器，正確地取用選項檔案並且選擇正確的 `remotename` 選項·

14 以手動方式建立 PPP 連結

現在你已經建立你的 /etc/ppp/options 以及 /etc/resolv.conf 檔案（以及，如果有需要，還有 /etc/ppp/pap|chap-secrets 檔案），你可以藉由手動設立 PPP 連線來測試這些設定（一旦我們使手動連線能運作，我們將會自動化該過程）·

要做這件事，你的通訊軟體必須能在不重置你的數據機的情況下結束· Minicom 可以做到 — ALT Q（或者在比較舊的 minicom 版本是使用 CTRL A Q）·

確定你以 root 簽入·

啓動你的通訊軟體（像是 minicom），撥接到 PPP 伺服器並且像平常一樣的簽入． 如果你需要發出某個指令來起動伺服端上的 PPP，那麼就做吧．你將會看見你以前看過的垃圾．

如果你使用 PAP 或 CHAP 的話，那麼只是連線到遠端系統應該就會起動在遠端上的 PPP 而且你將在沒有簽入的情況下就見到這些垃圾（雖然這在某些伺服器上不會發生 - 試著按下 **Enter** 看看能否看到這些垃圾）．

現在，在不要重置數據機的情況下結束通訊軟體(在 minicom 使用 `ALT Q` 或 `CTL A Q`)並且在 Linux 的提示符號下（以 root 的身分）鍵入

```
pppd -d -detach /dev/ttySx 38400 &
```

`-d` 這個選項開啓偵錯功能 — ppp 連線起動時的”交談”將會記錄到你的系統記錄裡去 — 如果你遇上麻煩這會很有用的．

你數據機的燈現在應該要閃爍，因爲要建立 PPP 連線．PPP 連線的建立將會花點時間．

此時你可以觀察 PPP 界面，藉由發出這個指令

```
ifconfig
```

在你所擁有的任何乙太網路以及回授設備之外，你應該會看見像這樣的資訊:

```
ppp0      Link encap:Point-Point Protocol
          inet addr:10.144.153.104  P-t-P:10.144.153.51 Mask:255.255.255.0
          UP POINTOPOINT RUNNING  MTU:552  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0
          TX packets:0 errors:0 dropped:0 overruns:0
```

其中

- inet addr:10.144.153.10 是該連結中你這端的 IP 號碼．
- P-t-P:10.144.153.5 是伺服端的 IP 號碼．

(自然， ifconfig 不會報告這些 IP 號碼，而是你的 PPP 伺服器上使用的．)

注意: ifconfig 也告訴你該連結已經起動並且正在執行!

如果沒有 ppp 裝置列出或得到的是像這樣的資訊:

```
ppp0      Link encap:Point-Point Protocol
          inet addr:0.0.0.0  P-t-P:0.0.0.0  Mask:0.0.0.0
          POINTOPOINT  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0
          TX packets:0 errors:0 dropped:0 overruns:0
```

你的 PPP 連線並沒有建立... 參閱稍後有關錯誤排除的章節!

你還可以看到這部遠端機器(及其後的)的遞送路徑．要做到這點，下這個指令

```
route -n
```

你應該看到像這樣的:-

```
Kernel routing table
Destination     Gateway         Genmask         Flags MSS    Window Use Iface
10.144.153.3    *               255.255.255.255 UH    1500   0        1 ppp0
127.0.0.0       *               255.0.0.0       U     3584   0       11 lo
10.0.0.0        *               255.0.0.0       U     1500   0       35 eth0
default         10.144.153.3    *               UG    1500   0        5 ppp0
```

特別注意這裡，我們有兩個項目指到我們的 ppp 裝置。

第一項是主機遞送(以 H 旗號表示)並且允許我們看到這部我們正在連接的主機 - 但再來就沒有了。

第二項是預設遞送路徑(由 pppd 的 `defaultroute` 選項所建立的)。 這份遞送路徑告訴我們的 Linux PC 將任何不在內部乙太網路的封包送到那裡 - 到我們指定的網路 - 也就是到 PPP 伺服器本身。 而 PPP 伺服器有責任爲我們將封包遞送到網際網路並將回應的封包送回給我們。

如果你在遞送表格中沒看到兩個項目，有些事不對勁。 特別是如果你的系統記錄(syslog)顯示一訊息告訴你 pppd 無法取代已存在的預設遞送路徑，那麼你已有一預設遞送路徑指向你的乙太網路界面 - 而這必須由指定的網路遞送來取代: 你只能有一個預設遞送路徑!!!

你必須檢查一下你系統的起始檔案以找出預設遞送是在那裡建立的(這是由 `route add default...` 指令指定的)。 將它改成像這樣的 `route add net...` 。

現在藉由 ifconfig 之輸出所回報的 IP 號碼來 'ping' 一下該伺服器以測試這個連結，像這樣

```
ping 10.144.153.51
```

你應該會接收到像這樣的輸出

```
PING 10.144.153.51 (10.144.153.51): 56 data bytes
64 bytes from 10.144.153.51: icmp_seq=0 ttl=255 time=328.3 ms
64 bytes from 10.144.153.51: icmp_seq=1 ttl=255 time=190.5 ms
64 bytes from 10.144.153.51: icmp_seq=2 ttl=255 time=187.5 ms
64 bytes from 10.144.153.51: icmp_seq=3 ttl=255 time=170.7 ms
```

這將會一直列印下去 — 按下 CTRL C 終止它，那時你將會接收到更多訊息:

```
--- 10.144.153.51 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 170.7/219.2/328.3 ms
```

到目前都很好。

現在試著藉由名稱 ping 某台主機（不是 PPP 伺服器它自己的名稱，而是在另一個地方的一台你知道可能已經起動而且正在運作的主機）。例如

```
ping sunsite.unc.edu
```

這時會有一點停頓，因爲 Linux 從你在 /etc/resolv.conf 裡指定的 DNS 取得你要 'ping' 的完整主機名稱所對應的 IP 位址 — 所以不要擔心（但是你將會看見你的數據機燈號閃爍）· 在短暫的時間後你將會接收到像這樣的輸出

```
 PING sunsite.unc.edu (152.2.254.81): 56 data bytes
64 bytes from 152.2.254.81: icmp_seq=0 ttl=254 time=190.1 ms
64 bytes from 152.2.254.81: icmp_seq=1 ttl=254 time=180.6 ms
64 bytes from 152.2.254.81: icmp_seq=2 ttl=254 time=169.8 ms
64 bytes from 152.2.254.81: icmp_seq=3 ttl=254 time=170.6 ms
64 bytes from 152.2.254.81: icmp_seq=4 ttl=254 time=170.6 ms
```

再一次，藉由按下 CTRL C 終止輸出並取得統計資料...

```
--- sunsite.unc.edu ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 169.8/176.3/190.1 ms
```

如果你沒有收到任何回應，試著 ping 一下你的 ISP 提供的 DNS 的 IP 位址· 如果你從這裡得到肯定的結果，那麼看起來問題在你的 /etc/resolv.conf 檔案裡·

如果這樣會沒有結果，問題出在網路遞送上，或者是你的 ISP 在將封包遞送回給你時出了問題· 檢查上面的遞送表格，如果沒問題，和你的 ISP 聯絡·一個測試 ISP 的好方法是用另一種作業系統來連接· 如果這樣就可以動，那麼問題就在你這邊·

如果一切正常，藉由鍵入這個指令終止該連線

```
ppp-off
```

一段短暫的停頓之後，數據機應該會自己掛斷電話

如果這個指令無法動作的話，那麼不是關掉你的數據機就是起動你的通訊軟體並且以 +++ 中斷數據機然後當你接收到數據機的 OK 提示時以 ATH0 掛斷電話·

你可能也得清除 pppd 所建立的鎖定檔

```
rm -f /var/lock/LCK..ttySx
```

15 使你的連結自動化 - 創造連結指令稿

一旦你可以用上述的方法用手動方式簽入，更好的做法是建立一些指令稿來自動爲你進行這些工作·

一份 chat 指令稿可以自動化簽入以及起動 PPP 的程序，因此所有你得做的（以 root 或是 ppp 群組組員的身分）只是發出單一個指令來起動你的連線·

15.1 以使用者名稱密碼認證的指令稿

如果你的 ISP 並不要求使用 PAP/CHAP 的話，這些就是爲你準備的!

如果 ppp 套件安裝正確，你應該會有兩個範例檔案． 在 PPP 2.1.2 中是在 /usr/sbin 而在 PPP 2.2 裡是在 /etc/ppp/scripts．這些是

PPP-2.1.2

```
ppp-on
ppp-off
```

而在 PPP-2.2

```
ppp-off
ppp-on
ppp-on-dialer
```

現在，如果你正在使用 PPP 2.1.2，我強烈建議你刪除這些範例檔案． 它們有潛在的問題 - 不要告訴我它們工作得很好 - 我也用了很長一段時間(而且還在這份 HOWTO 的第一版中推薦它們)!

爲了 ppp 2.1.2 使用者的權益，這裡有比較好的樣板檔案，這些是從 ppp 2.2 發行套件裡拿來的． 我建議你拷貝並使用這些指令稿而不要再用舊的 PPP-2.1.2 指令稿．

15.2 ppp-on 指令稿

這是實際啓動連線的一對指令稿中的第一個．

```
#!/bin/sh
#
# Script to initiate a PPP connection. This is the first part of the
# pair of scripts. This is not a secure pair of scripts as the codes
# are visible with the 'ps' command.  However, it is simple.
#
# These are the parameters. Change as needed.
TELEPHONE=555-1212      # The telephone number for the connection
ACCOUNT=george          # The account name for logon (as in 'George Burns')
PASSWORD=gracie         # The password for this account (and 'Gracie Allen')
LOCAL_IP=0.0.0.0        # Local IP address if known. Dynamic = 0.0.0.0
REMOTE_IP=0.0.0.0       # Remote IP address if desired. Normally 0.0.0.0
NETMASK=255.255.255.0   # The proper netmask if needed
#
# Export them so that they will be available to 'ppp-on-dialer'
export TELEPHONE ACCOUNT PASSWORD
#
# This is the location of the script which dials the phone and logs
# in.  Please use the absolute file name as the $PATH variable is not
# used on the connect option.  (To do so on a 'root' account would be
# a security hole so don't ask.)
```

```
#
DIALER_SCRIPT=/etc/ppp/ppp-on-dialer
#
# Initiate the connection
#
#
exec /usr/sbin/pppd debug /dev/ttySx 38400 \
        $LOCAL_IP:$REMOTE_IP \
        connect $DIALER_SCRIPT
```

而這是 ppp-on-dialer 指令稿:-

```
#!/bin/sh
#
# This is part 2 of the ppp-on script. It will perform the connection
# protocol for the desired connection.
#
/usr/sbin/chat -v                                       \
        TIMEOUT         3                               \
        ABORT           '\nBUSY\r'                      \
        ABORT           '\nNO ANSWER\r'                 \
        ABORT           '\nRINGING\r\n\r\nRINGING\r'    \
        ''              \rAT                            \
        'OK-+++\c-OK'   ATH0                            \
        TIMEOUT         30                              \
        OK              ATDT$TELEPHONE                  \
        CONNECT         ''                              \
        ogin:--ogin:    $ACCOUNT                        \
        assword:        $PASSWORD
```

在 PPP-2.2 中，ppp-off 指令稿看起來像:-

```
#!/bin/sh
######################################################################
#
# Determine the device to be terminated.
#
if [ "$1" = "" ]; then
        DEVICE=ppp0
else
        DEVICE=$1
fi

######################################################################
#
# If the ppp0 pid file is present then the program is running. Stop it.
```

```
if [ -r /var/run/$DEVICE.pid ]; then
        kill -INT `cat /var/run/$DEVICE.pid`
#
# If the kill did not work then there is no process running for this
# pid. It may also mean that the lock file will be left. You may wish
# to delete the lock file at the same time.
        if [ ! "$?" = "0" ]; then
                rm -f /var/run/$DEVICE.pid
                echo "ERROR: Removed stale pid file"
                exit 1
        fi
#
# Success. Let pppd clean up its own junk.
        echo "PPP link to $DEVICE terminated."
        exit 0
fi
#
# The ppp process is not running for ppp0
echo "ERROR: PPP link is not active on $DEVICE"
exit 1
```

15.3 編輯這裡所提供的起始指令稿

因為這份新的指令稿分成兩部份，我們分別編輯它們．

15.3.1 ppp-on 指令稿

你將得編輯這個指令稿以反映你在 ISP 的使用者名稱，你在 ISP 的密碼，你 ISP 的電話號碼．

每一行像是 TELEPHONE= 實際上設定了這個變數包含在等號右邊的資訊 (當然不包括註解部份)． 因此編輯每一行使它對你的 ISP 與連線是正確的．

而且，如果你(需要)在 /etc/ppp/options 檔案裡設定了 IP 號碼，刪除這行

```
$LOCAL_IP:$REMOTE_IP \
```

還有，確定變數 DIALER_SCRIPT 指向你將要使用的撥號指令稿的全部路徑名稱． 因此，如果你移動或改變這個指令稿的名稱，確定你正確地改變 ppp-on 指令稿裡的這一行．

15.3.2 ppp-on-dialer 指令稿

這是第二個指令稿實際建立起我們的 PPP 連結．

注意: 一份 chat 指令稿一般全都放在同一行裡．倒斜線是用來讓一行可以跨過數行的實體行（便於人們閱讀）而並不是指令稿本身組成的一部份．

然而，仔細地看看它的細節部份以便讓我們了解它實際(假設的)要做什麼是非常有用的!

15.4 一份 chat 指令稿其中的含意...

一份 chat 指令稿是一系列“期待字串”“送出字串”的配對． 特別是，注意到我們總是在送出某些訊息之前先期待某些訊息出現．

如果我們在沒有先接收到任何訊息的情況下要送出某些訊息的話，我們必須使用空的期待字串（藉由 “” 指明）而在沒有送出任何訊息的情況下要期待某些訊息作法也類似! 而且，如果字串包含好幾個字（例如 NO CARRIER），你必須把字串用引號括住，這樣 chat 會把字串當作是單一個項目．

在我們樣板裡的 chat 指令行是:

```
exec /usr/sbin/chat -v
```

起動 chat，-v 告訴 chat 將其所有的輸出入拷貝到系統記錄裡（通常是 /var/log/messages）． 一旦你覺得 chat 指令稿的運作穩定而可靠，編輯這一行把 -v 移除以便減少你系統記錄的混雜．

```
TIMEOUT         3
```

這設定接收所期待的輸入逾時時限爲三秒．如果你使用眞的是很慢的數據機那麼你可能得要增加這個設定到五秒或是十秒!

```
ABORT           '\nBUSY\r'
```

如果接收到 BUSY 字串，中止執行．

```
ABORT           '\nNO ANSWER\r'
```

如果接收到 NO ANSWER 字串，中止執行．

```
ABORT           '\nRINGING\r\n\r\nRINGING\r'
```

如果接收到（重複的）RINGRING 字串，中止執行．這是因爲某人正打電話給你!

```
"              \rAT
```

不期待數據機傳送任何訊息並且送出 AT 字串．

```
OK-+++\c-OK    ATH0
```

這一個比較複雜一點，因爲它使用了某些 chat 的錯誤回復功能．

它所代表的是...期待 OK，如果沒有收到（因爲數據機並不在指令模式下）那麼送出 +++ （使數據機返回指令模式的標準 Hayes 相容數據機字串）並且期待 OK；接著送出 ATH0 （數據機掛斷字串）．這讓你的指令稿可以應付你的數據機黏在線上無法斷線的情況．

```
TIMEOUT         30
```

設定指令稿其餘部份執行的逾時時限爲 30 秒．如果你曾經因爲逾時而經驗過 chat 指令稿被終止的問題，把它增加爲 45 秒或更久．

```
OK              ATDT$TELEPHONE
```

期待 OK （數據機對 ATH0 指令的回應）並且撥接到我們想要呼叫的號碼．

```
CONNECT         ''
```

期待 CONNECT 字串（當遠端數據機有回應時我們的數據機送出的）並且不送出任回覆訊息．

```
ogin:--ogin:    $ACCOUNT
```

再一次，這裡面我們有一些錯誤回復設定．期待簽入提示(...ogin:)，但是如果我們在逾時前沒有接收到，送出一個返回字元(return)然後再次找尋簽入提示． 當接收到提示時，送出使用者名稱（存放在使用者界面的 $ACCOUNT 環境變數裡）．

```
assword:        $PASSWORD
```

期待密碼提示並且送出密碼（再一次，存放在使用者界面的一個環境變數裡）．

這個 chat 指令稿有合理的錯誤回復功能． Chat 還有許多重要的特色，比這裡所展示還要更多． 更詳細的資訊請參閱 chat 的線上使用手冊(`man 8 chat`)．

15.4.1 在伺服器端啓動 PPP

雖然這份 `ppp-on-dialer` 指令稿對於在你簽入後會自動啓動 pppd 的伺服器來說非常的好用，然而某些伺服器需要你明白地在伺服器上啓動 PPP．

如果你需要在伺服器上下指令以啓動 PPP，你必須要編輯 `ppp-on-dialer` 指令稿．

在這指令稿的尾端(在 password 這行後面)加上 期待 送出 字串對 - 這是尋找你的 shell 提示符號 (特別小心對 Bourne shell 有特殊意義的字元 - 像是 $ 與 [或] 左右方括號等)．

一旦 chat 找到了 shell 提示符號，chat 必須下指令以啓動你 ISP 的 PPP 伺服器．

以我的例子而言，我的 PPP 伺服器使用標準的 Linux Bash 提示符號

```
[hartr@kepler hartr]$
```

而且需要我打

```
ppp
```

以啓動伺服器上的 PPP．

在此允許一點錯誤回復是個好主意，所以在我的例子中我用

```
        hartr--hartr    ppp
```

這就是說，如果我在逾時前沒收到提示符號，送出 return 並且再次等待提示符號．

一旦收到提示符號，就送出 ppp 字串．

注意: 別忘了在前一行結束前加上 \ 以便讓 chat 認爲這整個指令稿是在一行裡!

幸的是，某些伺服器會產生非常多變的提示符號! 你可能必須使用 minicom 多簽入幾次以便了解什麼可以用來做爲確定的“期待”字串．

15.5 使用於 PAP/CHAP 驗認連線的 chat 指令稿

如果你的 ISP 使用 PAP/CHAP 驗認，那麼你的 chat 指令稿會簡單得多． 你的 chat 指令稿所要做的全部只是撥接電話，等待接上線，然後就讓 pppd 去處理簽入事宜!

```
#!/bin/sh
#
# This is part 2 of the ppp-on script. It will perform the connection
# protocol for the desired connection.
#
exec /usr/sbin/chat -v                                       \
        TIMEOUT         3                                    \
        ABORT           '\nBUSY\r'                           \
        ABORT           '\nNO ANSWER\r'                      \
        ABORT           '\nRINGING\r\n\r\nRINGING\r'         \
        ''              \rAT                                 \
        'OK-+++\c-OK'   ATH0                                 \
        TIMEOUT         30                                   \
        OK              ATDT$TELEPHONE                       \
        CONNECT         ''                                   \
```

15.6 pppd 的 debug 與 file option_file 選項

就像我們已經看過的，你可以對 pppd 使用 -d 選項打開偵錯訊息記錄． 'debug' 選項和這個是一樣的．

因爲我們正在使用新的指令稿建立新的連線，先放著此選項不管． (警告: 如果你的磁碟空間不大夠，記錄 pppd 的訊息可能會大量增加你的系統記錄檔大小並使你遇到麻煩 - 不過在此你必定會連線失敗並嘗試許久)．

一旦你高興地認爲一切工作正常，你可以除去此選項．

如果你不是從 /etc/ppp/options 或 /etc/ppp/options.ttySx 檔案中叫出 ppp 選項， 可以在 pppd 上用 file 選項指定檔案名稱 - 例如

```
exec /usr/sbin/pppd debug file options.myserver /dev/ttyS0 38400 \
```

16 測試你的連線指令稿

開啓一個新的 Xterm（如果你在 X 底下）或開啓一個新的虛擬主控台並且以 root 簽入．

這個新的階段作業中，發出這個指令

```
tail -f /var/log/messages
```

（或是你系統記錄檔案的任何名稱）．

在第一個視窗（或是虛擬主控台中）發出這個命令

```
ppp-on &
```

（或是任何你所編輯的 /usr/sbin/ppp-on 版本之名稱）． 如果你沒有藉由在這個指令的尾端指定 & 使得指令稿進入背景執行的話，那麼在 ppp 結束（當連線結束時）之前你都不能回到終端機提示下

現在切換到追蹤系統記錄檔的視窗．

你將會見到某些像下列的東西（假定你對 chat 指定 -v 並且對 pppd 指定 -d 選項）.... 這是緊跟在 pppd 起動資訊之後記錄到系統記錄檔案裡的 chat 指令稿及其回應:

```
Oct 21 16:09:58 hwin chat[19868]: abort on (NO CARRIER)
Oct 21 16:09:59 hwin chat[19868]: abort on (BUSY)
Oct 21 16:09:59 hwin chat[19868]: send (ATZ^M)
Oct 21 16:09:59 hwin chat[19868]: expect (OK)
Oct 21 16:10:00 hwin chat[19868]: ATZ^M^M
Oct 21 16:10:00 hwin chat[19868]: OK -- got it
Oct 21 16:10:00 hwin chat[19868]: send (ATDT722298^M)
Oct 21 16:10:00 hwin chat[19868]: expect (CONNECT)
Oct 21 16:10:00 hwin chat[19868]: ^M
Oct 21 16:10:22 hwin chat[19868]: ATDT722298^M^M
Oct 21 16:10:22 hwin chat[19868]: CONNECT -- got it
Oct 21 16:10:22 hwin chat[19868]: send (^M)
Oct 21 16:10:22 hwin chat[19868]: expect (ogin:)
Oct 21 16:10:23 hwin chat[19868]: kepler login: -- got it
Oct 21 16:10:23 hwin chat[19868]: send (hartr^M)
Oct 21 16:10:23 hwin chat[19868]: expect (ssword:)
Oct 21 16:10:23 hwin chat[19868]:  hartr^M
Oct 21 16:10:23 hwin chat[19868]: Password: -- got it
Oct 21 16:10:23 hwin chat[19868]: send (??????^M)
Oct 21 16:10:23 hwin chat[19868]: expect (hartr)
Oct 21 16:10:24 hwin chat[19868]: [hartr -- got it
Oct 21 16:10:24 hwin chat[19868]: send (ppp^M)
Oct 21 16:10:27 hwin pppd[19872]: pppd 2.1.2 started by root, uid 0
Oct 21 16:10:27 hwin pppd[19873]: Using interface ppp0
Oct 21 16:10:27 hwin pppd[19873]: Connect: ppp0 <--> /dev/cua1
Oct 21 16:10:27 hwin pppd[19873]: fsm_sdata(LCP): Sent code 1, id 1.
Oct 21 16:10:27 hwin pppd[19873]: LCP: sending Configure-Request, id 1
```

```
Oct 21 16:10:27 hwin pppd[19873]: fsm_rconfreq(LCP): Rcvd id 1.
Oct 21 16:10:27 hwin pppd[19873]: lcp_reqci: rcvd MRU
Oct 21 16:10:27 hwin pppd[19873]: (1500)
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: lcp_reqci: rcvd ASYNCMAP
Oct 21 16:10:27 hwin pppd[19873]: (0)
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: lcp_reqci: rcvd MAGICNUMBER
Oct 21 16:10:27 hwin pppd[19873]: (a098b898)
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: lcp_reqci: rcvd PCOMPRESSION
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: lcp_reqci: rcvd ACCOMPRESSION
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: lcp_reqci: returning CONFACK.
Oct 21 16:10:27 hwin pppd[19873]: fsm_sdata(LCP): Sent code 2, id 1.
Oct 21 16:10:27 hwin pppd[19873]: fsm_rconfack(LCP): Rcvd id 1.
Oct 21 16:10:27 hwin pppd[19873]: fsm_sdata(IPCP): Sent code 1, id 1.
Oct 21 16:10:27 hwin pppd[19873]: IPCP: sending Configure-Request, id 1
Oct 21 16:10:27 hwin pppd[19873]: fsm_rconfreq(IPCP): Rcvd id 1.
Oct 21 16:10:27 hwin pppd[19873]: ipcp: received ADDR
Oct 21 16:10:27 hwin pppd[19873]: (10.144.153.51)
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: ipcp: received COMPRESSTYPE
Oct 21 16:10:27 hwin pppd[19873]: (45)
Oct 21 16:10:27 hwin pppd[19873]:  (ACK)
Oct 21 16:10:27 hwin pppd[19873]: ipcp: returning Configure-ACK
Oct 21 16:10:28 hwin pppd[19873]: fsm_sdata(IPCP): Sent code 2, id 1.
Oct 21 16:10:30 hwin pppd[19873]: fsm_sdata(IPCP): Sent code 1, id 1.
Oct 21 16:10:30 hwin pppd[19873]: IPCP: sending Configure-Request, id 1
Oct 21 16:10:30 hwin pppd[19873]: fsm_rconfreq(IPCP): Rcvd id 255.
Oct 21 16:10:31 hwin pppd[19873]: ipcp: received ADDR
Oct 21 16:10:31 hwin pppd[19873]: (10.144.153.51)
Oct 21 16:10:31 hwin pppd[19873]:  (ACK)
Oct 21 16:10:31 hwin pppd[19873]: ipcp: received COMPRESSTYPE
Oct 21 16:10:31 hwin pppd[19873]: (45)
Oct 21 16:10:31 hwin pppd[19873]:  (ACK)
Oct 21 16:10:31 hwin pppd[19873]: ipcp: returning Configure-ACK
Oct 21 16:10:31 hwin pppd[19873]: fsm_sdata(IPCP): Sent code 2, id 255.
Oct 21 16:10:31 hwin pppd[19873]: fsm_rconfack(IPCP): Rcvd id 1.
Oct 21 16:10:31 hwin pppd[19873]: ipcp: up
Oct 21 16:10:31 hwin pppd[19873]: local  IP address 10.144.153.104
Oct 21 16:10:31 hwin pppd[19873]: remote IP address 10.144.153.51
```

（注意 — 我使用靜態的 IP 號碼 — 因此我的機器送這些東西到 PPP 伺服器去 — 如果你使用動態的 IP 號碼那麼你將不會看見這些．） 還有，這個伺服器需要指定的命令來啓動 PPP．

這看起來沒什麼問題 — 所以如同前面的測試般， ping 一些 IP 號碼以及主機名稱．

動你的全球資訊瀏覽器或是任何其它東東並且開始逛 — 你上線了!

17 結束 PPP 連結

當你完成 PPP 連結之後，可以使用標準的 ppp-off 指令終止它（要記得 — 你得是 root 或者是 PPP 群組的組員!）

你將會在你的系統記錄裡看見一些像這樣的的東西:

```
Oct 21 16:10:45 hwin pppd[19873]: Interrupt received: terminating link
Oct 21 16:10:45 hwin pppd[19873]: ipcp: down
Oct 21 16:10:45 hwin pppd[19873]: default route ioctl(SIOCDELRT): Bad address
Oct 21 16:10:45 hwin pppd[19873]: fsm_sdata(LCP): Sent code 5, id 2.
Oct 21 16:10:46 hwin pppd[19873]: fsm_rtermack(LCP).
Oct 21 16:10:46 hwin pppd[19873]: Connection terminated.
Oct 21 16:10:46 hwin pppd[19873]: Exit.
```

別擔心那個 SIOCDELRT - 這不過是 pppd 表示要結束了而沒什麼好擔心的．

18 偵錯

你的連線有各種可能的原因無法運作 — chat 無法正確地完成，你的線路雜訊很大等等． 所以，檢查你的系統記錄找尋線索．

18.1 我把 PPP 編譯進去但是 Linux 說我沒有...

一個非常常見的問題是人們已經將 PPP 編譯到核心之中並且嘗試執行 pppd，但核心仍然抱怨說它不支援 PPP! 有許多原因可能導致此事發生．

18.1.1 你啓動的是正確的核心嗎?

雖然你已經重新編譯核心以支援 PPP，你卻沒有啓動新的核心． 這可能是因爲你沒有更新 /etc/lilo.conf 並重跑 lilo．

檢查的方法是下這個指令 uname -a，將產生像這樣的結果

```
Linux archenland 2.0.28 #2 Thu Feb 13 12:31:37 EST 1997 i586
```

它給出了核心的版本及核心編譯的日期 - 這樣你就知道到底發生了什麼事．

18.1.2 你將 PPP 核心支援編譯爲模組嗎?

如果你將 PPP 核心支援編譯爲模組，但卻沒有編譯及安裝模組，你就會得到這個錯誤· 看一下 *Kernel-HOWTO* 以及在 `/usr/src/linux` 下的 README 檔案!

另一個模組連結的可能問題是你期望需要的模組自動地被載入，但卻沒有執行 `kerneld` (會自動載入並移除模組的工具)· 看一下 *kerneld mini-HOWTO* 裡的資訊說明如何設定 `kerneld`·

18.1.3 你是否你用正確的 PPP 版本配合你的核心?

你必須使用 ppp-2.2 以配合核心 2.0.X· 你可以在核心 1.2.X 使用 ppp-2.2 (如果你修補過核心)否則你必須使用 ppp-2.1.2·

18.1.4 你是否以 root 身份執行 pppd?

如果你不是以 root 身份執行 pppd (並且 pppd 並未設定爲以 root 身份執行)，你就會收到此訊息·

18.2 我的數據機連上了但 PPP 並未啓動

同樣也有許不盡的原因(參考一下 comp.os.linux...)·

一個最常見的錯誤是在你的指令稿裡你少打了某些東西· 這裡唯一可做的是你將 Linux PC 與伺服器的對話記到你的系統記錄中(`/var/log/messages`)然後一行一行地看個仔細· 你可能還需要再次以手動方法撥入伺服器檢查一遍·

你得要從頭到尾小心地檢查 — 而且心裡要記得我們人類有種傾向，閱讀的是我們認爲我們鍵入的 — 而不是眞的在那裡的!

18.3 系統記錄說 “`serial line is not 8 bit clean...`”

這也有許多的類似情形 - 像是 `serial line looped back` 等等，導致的原因可能是許多事情中的一件(或一系列)·

要知道到底發生了什麼，必須對 pppd 背後做了些什麼有點了解·

當 pppd 啓動後，它會送出連結控制協定(link control protocol)封包到遠端機器· 如果它收到合法的回應才會走到下一階段(使用 IPCP - IP 控制封包)而且只有在這協商完成之後實際的 IP 層才會建立因此你才能使用 PPP 連結·

如果當你的 PC 送出協商封包時在遠端沒有 PPP 伺服器在運作，這些封包在遠端簽入過程中將被彈回來· 因爲這些封包是使用 8 bits，彈回來時會將第八個位元截掉(記任，ASCII 是七位元的碼)· PPP 因此而抱怨此訊息·

有許多原因會造成協商封包被彈回·

18.3.1 你沒有正確地簽入伺服器

當你的 chat 指令稿完成後，你的 PC 會啓動 pppd·然而，如果你並未完成在伺服器的簽入過程(包括送出任何必要在伺服器上啓動 PPP 的指令)，PPP 就不會開始·

因此連結控制協定封包被彈回你也因此收到這個錯誤·

你必須小心地檢查並修正(必要的話)你的 chat 指令稿(參見上面說明)·

18.3.2 你並未啓動伺服器上的 PPP

某些 PPP 伺服器在你完成簽入過程後需要你輸入指令或按下 RETURN 才會在遠端啓動 PPP·

檢查你的 chat 指令稿(參見上面說明)·

如果你以手動方式簽入時發現你必須送出 RETURN 才會在遠端啓動 PPP，簡單地在你的 chat 指令稿尾端加上空白的期待/送出字串對(空的送出字串實際上會送出 RETURN)·

18.3.3 遠端的 PPP 過程啓動很慢

這有點兒技巧!

預設的情況下你的 Linux pppd 被編譯成最多送出十個連線控制要求封包· 如果伺服器啓動有點慢，十個連線封包可能在遠端 PPP 準備好接收前就全部送出了·

於是在你的機器上，pppd 看到十個封包被彈回(第八位元被截去)而結束·

有兩個方法可以解決此事:-

在你的 PPP 選項中加上 `lcp-max-configure 30`· 這增加 pppd 在放棄前送出的連線封包的最大數目·對一個眞的很慢的伺服器來說，你可能還需要更多·

或者，你可以回過來用一些技巧·你或許會注意到當你以手動簽入 PPP 伺服器並讓 PPP 啓動時，收到的垃圾的第一個字完總是 tilde(~) 字元·

利用此點我們可以在 chat 指令稿尾端加上新的期待/送出字串對，期待 tilde 字元並不送出任何東西·這看起來像這樣:-

```
\~          ''
```

注意: 因爲 tilde 字元對 shell 來說有特殊意義，必須加逸出符號(就是前面的倒斜線)·

18.4 不能設立預設遞送路徑

如果 pppd 拒絕建立預設遞送路徑，這是因爲(應該沒錯)它拒絕移除或取代已有的預設遞送路徑·

通常的原因是因爲某些套件將你的乙太網路卡設爲預設遞送路徑而不是設爲指定的網路遞送·

參見 Linux NAG 與 Net2/3 HOWTOs 裡的資訊以正確地設定你的乙太網路卡及相關的遞送·

另一可能原因是你的區域網路已使用了閘道器或路由器而且你的遞送表格已設定爲將預設遞送路徑指向這裡·

要修正這種情況需要更多的網路知識而已經超出此份 HOWTO 的範圍了· 建議你取得一些專家的意見(經由新聞組群或你周圍可以問的人)·

18.5 其它問題

還有許多原因導致 PPP 無法連接或是無法正確運作·

現在仔細看看 PPP FAQ （這眞的是一系列的問題與回答）·這是一份非常詳實的文件而且答案就在裡面! 以我自己（很爛）的經驗，如果你的問題答案不在其中，那麼該問題就不是 PPP 的錯! 以我爲例我使用 ELF 核心而且沒有升級適當的核心模組·在曙光出現之前我僅僅浪費了大概兩天（以及一個晚上的大部分時間）詛咒去那個事實上已非常良好的 PPP 伺服器·

19 當完全陷於膠著時如何取得幫助

如果你不能使你的 PPP 連結運作，回頭重新閱讀這份文件並檢查每一件事 — 同時配合 "chat -v..." 以及 "pppd -d" 在你的系統記錄裡建立的輸出作爲參考·

也要參考 PPP 本身的文件以及 FAQ 並加上在此曾經提及的文件!

如果你仍然陷於膠著狀態的話，那麼有些能夠在 PPP 這方面幫你的人會合理的、有規律的掃描 comp.os.linux.misc 與 comp.os.linux.networking 新聞討論群，以及 comp.protocols.ppp·

你可以嘗試寄個人的電子郵件給我，但我眞的有自己的工作（和生活）而且我不保證會很快地回覆（即使不完全會這樣）因爲這得視我目前的工作負擔以及我私人的生活狀態而定!

特別是 — 不要張貼剪下來的錯誤偵測輸出到新聞討論群上也不要藉由電子郵件寄給我 — 前者浪費大量的網路頻寬而後者將轉給 `/dev/null` (除非我特別要求將它寄給我!)

20 連結運作之後的普遍問題

一個問題是你將會發現許多服務提供者只支援他們給新帳號的套裝連線軟體· 這（典型）是給微軟視窗環境的 :-(— 而且許多服務提供者的支援處看來一點都不了解 Unix (or Linux)· 所以要有準備，從他們那得不到太多協助!

你當然可以幫他們一個忙並且教育他們關於 Linux 的知識（任何 ISP 的協助人員在網際網路用語中都應該 "知道它" 而這意味著他們在家裡應該要有台 Linux 機器 — 當然是這樣）!

20.1 我無法看見我連線的 PPP 伺服器以外的世界

好的 — 你的 PPP 連線已建立並執行中而且你可以藉由該 PPP 伺服器的 IP 號碼（藉由 `ifconfig ppp0` 指令所顯示的第二個或 "遠端" 的 IP 號碼） ping 該伺服器，但是你不能觸及其後的任何地方·

首先，嘗試 ping 你在 `/etc/resolv.conf` 裡指定作爲名稱伺服器那台機器的 IP 號碼· 如果可以的話，那麼你就可以見到在你 PPP 伺服器之外的東西（除非這個 IP 號碼跟你連線的 "遠端" IP 號碼相同）· 所以現在來試試 ping 你的服務提供者的完整網際網路名稱 — 例如

```
ping my.provider.net.au
```

如果這樣無法成功，那麼你有個名稱解析上的問題·這可能是因爲在你的 `/etc/resolv.conf` 檔案裡有打字錯誤· 打電話給你的服務提供者對照所得的資訊並且小心地檢查這個檔案·如果看起來全都沒問題，再打電話給服務提供者確定你寫下正確的 IP 號碼〉

如果它仍然無法運作（而且你的服務提供者確認他的名稱伺服器有開機並且正在執行），那麼你的問題在其它的地方 — 而且我建議你小心地從頭到尾檢查你安裝的 Linux（特別注意檔案權限）·

果你仍然不能藉由 IP 號碼 ping 你服務提供者的名稱伺服器，那麼不是他們關掉機器（打個電話給他們確定一下）就是有個遞送問題發生在你的服務提供者那邊· 再一次，打電話給他們查問看看·

有個可能是 "遠端" 是一台 Linux PPP 伺服器而在其核心裡沒有指定 IP forwarding 選項!

一個良好的測試方法是使用大多提供給（一整套） Microsoft Windows 的軟體嘗試進入你的服務提供者· 如果完全相同的帳號在另外一套作業系統下一切正常，那麼問題是在你的 Linux 系統上而不是在你的服務提供者那邊·

20.2 我可以寄送電子郵件，但不能接收

如果你使用動態的 IP 號碼，這是很正常的·參閱下面的 "21 (設立服務)" 一節·

20.3 爲什麼人們不能來我的機器 finger, WWW, gopher, talk 等等?

同樣地，如果你使用動態的 IP 號碼，這是非常正常的· 參閱下面的 "21 (設立服務)"該節·

21 以動態 IP 提供網際網路服務

如果你使用動態的 IP 號碼（許多服務提供者也只提供動態的 IP 號碼除非你爲你的連線付出更多代價），那麼你必須認識這隱含的限制·

首先，外送服務請求將不會有問題·這是說你可以使用 sendmail 送出電子郵件，從遠端節點傳輸(ftp)檔案， 調查(finger)在其它機器上的使用者，瀏覽全球資訊網等等·

特別是，雖然你不在線上但你可以回覆帶回你機器上的電子郵件·郵件將單純地丟在你的郵遞佇列中直到你撥回你的 ISP 再行送出·

然而，你的機器並非一天 24 小時都連接在網際網路上，它每次連線也不會擁有相同的 IP 號碼· 所以你不可能接得到直接寄到你機器的電子郵件，並且很難設立一台全球資訊伺服器或檔案傳輸伺服器! 因爲網際網路與你的機器不存在一種唯一的，永久可以聯繫的關係，因爲你的機器並沒有唯一的 IP 號碼(記住 - 其它的機器也會使用這些 IP 號碼，當它們撥上線時)·

如果你設立一台 WWW（或任何其它伺服器），在網際網路上的任何人都完全不會知道除非他們知道你的機器現在正在線上以及它確實（目前）的 IP 號碼· 他們有許多管道可以得到這項資訊，包括由你打電話給他們，寄發電子郵件告訴他們或者是在你服務提供者的帳號上巧妙地使用 ".plan" 檔案（假設你的提供者允許 shell 及 finger 存取）·

現在，對大部分的使用者而言，這不是個問題 — 大部分人們所想要做的全部只是寄送與接收電子郵件（使用在你服務提供者上的帳號）以及連線到全球資訊網，傳輸檔案以及其它網際網路上的服務·如果你必須擁有由外往內的連線服務，你真的應該取得靜態的 IP 號碼·另外一種方法是研究前面提示過的方法...

21.1 設定電子郵件

即使是動態的 IP 號碼，你的確可以配置你機器上的 sendmail 來送出你在本地端寫好的任何電子郵件· 有關 sendmail 的配置可能會是很含糊而且很困難的 — 所以這份文件不會試圖告訴你該怎麼做· 然

而，你可能應該配置你的 sendmail 以便你的網際網路服務提供者能授命你作爲 "smart relay" 主機（在 `sendmail.cf` 的 **DS** 選項）．（更爲詳細的配置資訊請參閱 sendmail 的文件 — 並詳閱隨附於 sendmail 的 m4 配置巨集．幾乎可以確定裡面一定有一個能符合你的需要）．

也有一本非常好的書在講 Sendmail(O'Reilly and Associates 稱它爲 '聖經')，但對大部份的使用者來說實在太過困難了!

一旦你配置好 sendmail 後，你可能想在 PPP 連線剛一起動的時候就要 sendmail 趕快送出任何待在郵件佇列裡的訊息． 要這麼做的話，加入這行指令

```
sendmail -q &
```

到你的 `/etc/ppp/ip-up` 指令稿中(參見下面說明)．

由外往內的電子郵件對動態的 IP 號碼而言是個問題．處理這個問題的方法是:

- 配置你的使用者郵遞程式讓所有送出去的郵件在 "reply to" 表頭中有個在你的網際網路服務提供者上的電子郵遞位址．
  如果可以的話，你也應該將你的 FROM 地址設爲你 ISP 提供的電子郵遞位址．
- 使用 popclient 程式從你的服務提供者那邊擷取你的電子郵件．或者，如果你的 ISP 使用 IMAP，使用支援 IMAP 的使用者郵遞程式(例如 pine)．

你可以藉由在 `/etc/ppp/ip-up` 指令稿中放入必要的指令以便在撥接時將這個過程自動化(參見下面說明)．

21.2 設定本地的名稱伺服器

雖然你大可使用在你 ISP 那裡的領域名稱伺服器，你也可以設立一台本地的暫存專用（次要）名稱伺服器，並藉由 `ip-up` 指令稿啟動． 跑本地（暫存專用）名稱伺服器的好處是如果你在連線期間常常聯繫相同的節點那麼它可以節省你的時間（以及頻寬）．

暫存專用名稱伺服器（在 `named.boot` 檔案裡使用一行 'forwarders' 指向你 ISPs 的 DNS）的名稱伺服器配置比較簡單． O'Reilly book (DNS and Bind) 解釋所有你需要知道的東西．

也有一份 *DNS-HOWTO* 可以取得．

如果你有一小型區域網路經由你的 Linux PC 取用網際網路(例如使用 IP 偽裝)， 在本地端執行名稱伺服器(使用 forwarders 導向)會是個好主意，因爲它會降低由名稱解析所佔用的網路頻寬及延遲時間．

網路作業的一個要點: 當你要在你 ISP 的領域裡開始啓用一台次要的，暫存專用的名稱伺服器之前先詢問你的 ISP 相關的權限． 正確配置它，你的 DNS 一點都不會造成你 ISP 的困擾，但是如果你做錯了些什麼，它可能會引起麻煩...

22 使用 PPP 連結兩個網路

基本上連結單一台 Linux PC 到 PPP 伺服器以及使用 PPP 連結兩個區域網路沒有什麼差別．記得，PPP 是一種點對點的協定．

然而，你的確得瞭解關於遞送路徑如何建立的問題．閱讀 NET-2 howto 以及 Linux Network Administrator Guide (NAG)． 你也將發現這本書 "TCP/IP Network Administration" (by O'Reilly and Assoc - ISBN 0-937175-82-X)會有極佳的幫助．

如果你想在連結的某一邊使用一個網路號碼的次網路分割，你會發現 Linux (draft) sub networking mini-howto 滿有用的． 這可在 *Linux Sub networking mini-HOWTO* <http://www.interweft.com.au/other/> 找到．

爲了連結兩個區域網路，你使用的必須是不同的 IP 網路編號(或是同樣網路編號的次網路)而且你將得使用靜態的 IP 號碼 — 或使用 IP 僞裝． 如果你想要使用 IP 僞裝的話，參閱 IP masquerade mini-howto 有關它的設定介紹．

22.1 設定 IP 號碼

與其它區域網路的網路管理者協商兩端的 PPP 界面使用的 IP 號碼． 如果你使用靜態的 IP 號碼，這可能也會要求你撥接特定的電話號碼．

現在編輯適當的 /etc/ppp/options[.ttyXX] 檔案 — 對於此連線在你這一端使用特定的數據機及串列埠是個好主意． 你可能也得要去變更你的 /etc/ppp/options 檔案 — 並且也要爲任何其它的連線建立適當的 options.ttyXX 檔案!

在適當的選項檔案裡指定 PPP 連接中你這端的 IP 號碼，這與前面所展示使用靜態 IP 號碼撥接的方式完全相同．

22.2 設定遞送路徑

你必須安排你區域網路上的封包經過 PPP 連結所建立的界面遞送出去．這是個兩階段的過程．

首先，你必須建立從執行 PPP 連結的機器到連結的遠方網路去的遞送路徑． 如果該連結通往網際網路，那麼這可以藉由 pppd 的選項 'defaultroute' 由它自己建立的預設遞送路徑來處理而你不必做任何事．

然而如果該連結只是連結兩個區域網路，那麼必須加入一個指定的網路遞送路徑． 這是在 /etc/ppp/ip-up 指令稿中使用 'route' 指令執行的（參閱連線起動之後... 該節裡有關如何執行的介紹）．

你得做的第二件事是告訴在你區域網路上的其它電腦這台 Linux 電腦實際上是該 ppp 連結中通往遠方網路的'閘道'．

當然，這些工作該連結另一端的網路管理者也全都得做! 然而，因爲她/他必須將封包遞送到你的網路，一個指定的網路遞送路徑是必要的，而不是預設遞送路徑(除非遠端的區域網路是連上你去取用網際網路的)．

22.3 網路安全

如果你使用 PPP 連結你的區域網路到網際網路上去 — 或甚至只是到一個"外面的"區域網路上去，你都必須考慮安全性的問題． 我強烈鼓勵你考慮設立一台防火牆!

你還應該在以此方式連到外面的區域網路或網際網路之前先告訴你的區域網路管理者． 沒有這樣做的話可能使你惹上嚴重的麻煩!

23 建立連結之後 - /etc/ppp/ip-up 指令稿

一旦 PPP 連結建立後， pppd 會找尋 /etc/ppp/ip-up 指令稿． 如果這個指令稿存在並且可以執行的話，那麼 PPP 伺服程式就會執行這個指令稿． 這允許你自動執行任何可能有必要的特殊遞送路徑指令及任何你想在每次 PPP 連結啓動時執行的動作．

這不過就是一般普通的 shell 指令稿，可以做任何指令稿能做的事(也就是，任何你想做的事)．

如，你可能想要 sendmail 趕快處理在郵件佇列中等待外送的訊息．

類似地，你可以在 ip-up 裡插入一些指令取得（使用 POP）在你的 ISP 那邊等著你的任何電子郵件．

不過在 /etc/ppp/ip-up 也有些限制:-

- 它以局限的環境變數以增加安全性．這也就是說你必須給出執行檔的全部路徑名稱等．
- 技術上來說，/etc/ppp/ip-up 是一個程式而非指令稿． 這是說它可以被直接執行 - 也因此它在第一行開始必須具有標準的 file magic (#!/bin/bash) 並且能被 root 讀取及執行．

23.1 特殊遞送路徑

如果你連結的是兩個區域網路，你將得設立一個到 '外面的' 區域網路的指定遞送路徑． 這可以很容易的使用 /etc/ppp/ip-up 指令稿達成．唯一的困難發生在你的機器有多個 PPP 連結時．

這是因爲 /etc/ppp/ip-up 這個指令稿是由每一個起動的 ppp 連線所執行的，所以你得要小心地爲每一個起動的連結執行正確的遞送指令!

23.2 處理電子郵件佇列

當兩個區域網路的連結建立之後，你可能想要確定放在佇列中的電子郵件被清出 - 送到它的目的地．這可以用適當的呼叫 sendmail 來達成．

在 pppd 傳遞給指令稿的特定參數上使用 bash 的 'case' 敘述來完成這個工作． 例如，這是我用來處理我們的廣域網路連結及通往我家乙太網路的（也是由相同的 PPP 伺服器處理）/etc/ppp/ip-up 指令稿．

23.3 一個 /etc/ppp/ip-up 指令稿的例子

下面的例子提供不同的使用範例．

```
#!/bin/bash
#
# Script which handles the routing issues as necessary for pppd
# Only the link to Newman requires this handling.
#
# When the ppp link comes up, this script is called with the following
# parameters
#       $1      the interface name used by pppd (e.g. ppp3)
#       $2      the tty device name
#       $3      the tty device speed
#       $4      the local IP address for the interface
#       $5      the remote IP address
#       $6      the parameter specified by the 'ipparam' option to pppd
#
case "$5" in
```

```
# Handle the routing to the Newman Campus server
        202.12.126.1)
                /sbin/route add -net 202.12.126.0 gw 202.12.126.1
# and flush the mail queue to get their email there asap!
                /usr/sbin/sendmail -q &
                ;;
        139.130.177.2)
# Our Internet link
# When the link comes up, start the time server and synchronise to the world
# provided it is not already running
                if [ ! -f /var/lock/subsys/xntpd ]; then
                        /etc/rc.d/init.d/xntpd.init start &
                fi
# Start the news server (if not already running)
                if [ ! -f /var/lock/subsys/news ]; then
                        /etc/rc.d/init.d/news start &
                fi
                ;;
        203.18.8.104)
# Get the email down to my home machine as soon as the link comes up
# No routing is required as my home Ethernet is handled by IP
# masquerade and proxyarp routing.
                /usr/sbin/sendmail -q &
                ;;
        *)
esac
exit 0
```

起動通往我們 Newman 校園的 ppp 連結以及這個指令稿的結果，我們最後得到下面這個遞送表格記錄（這台機器也是我們通常用的 PPP 伺服器並且處理我們的網際網路連結）· 我已經在這個輸出裡加入一些註解以解釋每個項目是什麼:

```
[root@kepler /root]# route -n
Kernel routing table
Destination     Gateway         Genmask         Flags MSS    Window Use Iface
# the HOST route to our remote internet gateway
139.130.177.2   *               255.255.255.255 UH    1500   0      134 ppp4
# the HOST route to our Newman campus server
202.12.126.1    *               255.255.255.255 UH    1500   0       82 ppp5
# the HOST route to my home ethernet
203.18.8.104    *               255.255.255.255 UH    1500   0       74 ppp3
# two of our general dial up PPP lines
203.18.8.64     *               255.255.255.255 UH    552    0        0 ppp2
203.18.8.62     *               255.255.255.255 UH    552    0        1 ppp1
# the specific network route to the Newman campus LAN
202.12.126.0    202.12.126.1    255.255.255.0   UG    1500   0        0 ppp5
# the route to our local Ethernet (super-netting two adjacent C classes)
```

```
203.18.8.0      *               255.255.254.0   U     1500   0     1683 eth0
# the route to the loop back device
127.0.0.0       *               255.0.0.0       U     3584   0      483 lo
# the default route to the Internet
default         139.130.177.2   *               UG    1500   0     3633 ppp4
```

23.4 處理電子郵件

上一節提及如何處理外送的郵件 — 一旦連線建立之後簡單地藉由清出郵件佇列達成．

如果你執行連往廣域網路的連結，你可以跟遠端區域網路的網路管理者協調請它們執行完全相同的動作． 例如，在我們的廣域網路連結中 Newman 校園那一端的 /etc/ppp/ip-up 指令稿看起來像:

```
#!/bin/bash
#
# Script which handles the routing issues as necessary for pppd
# Only the link to Hedland requires this handling.
#
# When the ppp link comes up, this script is called with the following
# parameters
#       $1      the interface name used by pppd (e.g. ppp3)
#       $2      the tty device name
#       $3      the tty device speed
#       $4      the local IP address for the interface
#       $5      the remote IP address
#       $6      the parameter specified by the 'ipparam' option to pppd
#
case "$5" in
        203.18.8.4)
                /usr/sbin/sendmail -q
                ;;
        *)
esac
exit 0
```

然而如果你只能使用動態 IP 號碼方式的 PPP 連線連往你的 ISP，你得從在你 ISP 機器上的帳號取得你的電子郵件． 這通常是使用 POP(Post Office Protocol)協定來達成的．可以使用 'popclient' 程式處理這個程序 — 而 ip-up 指令稿也可以爲你自動化這個程序．

簡單地建立一個 /etc/ppp/ip-up 指令稿，其中包含有起動 popclient 程式的適當指令． 在我執行 Red Hat Linux 的膝上型電腦上（任何旅行時我都帶著它）是這樣的

```
popclient -3 -c -u hartr -p <password> kepler.hedland.edu.au |formail -s procmail
```

你可以使用 slurp 或其它軟體以相同的方式取得網路新聞，以及諸如此類的服務． 記得，ip-up 這個指令稿只是個標準的 bash 指令稿因此可以用來自動化當每次適當的 PPP 連結建立時需要完成的任何功能．

24 使用 /etc/ppp/ip-down

你可以建立一指令稿在連接一旦結束之後執行·這存放在 /etc/ppp/ip-down 裡· 這可以用來還原任何對應你在 /etc/ppp/ip-up 指令稿中做的特殊動作·

25 區域網路上的遞送問題

如果你已經連接在某個區域網路上但是仍然想要使用在你個人的 Linux 機器上的 pppd 的話，你得要應付某些遞送封包的問題，從你的機器到你的區域網路（經過你的乙太網路界面）以及到遠端的 PPP 伺服器和更外面的世界·

這一節並不嘗試教你有關封包遞送的機制 — 它只處理一個簡單的，特定的遞送情況（靜態的）!

如果你不熟悉遞送機制的話，那麼我強烈地鼓勵你閱讀 Linux Network Administrator Guide (NAG)·同時 O'Reilly 的書 "TCP/IP Network Administration" 也以非常容易了解的方式涵蓋了這個主題·

靜態遞送路徑的基本規則是預設遞送路徑應該是指向網路位址號碼最多的那一個· 對於其它的網路則在遞送表格中輸入指定遞送路徑·

我唯一打算在這裡介紹的情況是你的 Linux 機器在一個沒有連上網際網路的區域網路上 — 而你想要在仍然連接著區域網路的情況下撥接到網際網路上供自己使用·

首先，確定你的乙太網路遞送路徑是設往能夠通過你的區域網路的指定網路位址 — 不是設往預設遞送路徑!

藉由發出 route 指令檢查之，你應該會看見如下的訊息:

```
[root@hwin /root]# route -n
Kernel routing table
Destination     Gateway         Genmask         Flags MSS    Window Use Iface
loopback        *               255.255.255.0   U     1936   0        50 lo
10.0.0.0        *               255.255.255.0   U     1436   0       565 eth0
```

如果你的乙太網路界面 (eth0) 指向預設遞送路徑，（在 eth0 該行的第一列上會顯示 "default" 字樣）那麼你得要變更你的乙太網路起始指令稿使它指向特定的網路號碼而非預設遞送路徑（參照 Net2 HOWTO 以及 NAG）·

這將允許 pppd 設立你的預設遞送路徑，如下所示:

```
[root@hwin /root]# route -n
Kernel routing table

Destination     Gateway         Genmask         Flags MSS    Window Use Iface
10.144.153.51   *               255.255.255.255 UH    488    0         0 ppp0
127.0.0.0       *               255.255.255.0   U     1936   0        50 lo
10.1.0.0        *               255.255.255.0   U     1436   0       569 eth0
default         10.144.153.51   *               UG    488    0         3 ppp0
```

如同你所見到的，我們擁有經由 ppp0 到 PPP 伺服器(10.144.153.51)的主機遞送設定而且也擁有使用 PPP 伺服器作爲閘道的預設網路遞送設定·

如果你的設定得要比此更複雜 — 閱讀已經提過有關遞送的文件並請教在你身旁的專家!

如果你的區域網路上已經有路由器，那麼在你那裡已經建立了通往寬廣網路世界的閘道． 你仍應把你的預設遞送路徑指往 PPP 界面 — 並且使其它的遞送路徑指定到由路由器服務的網路．

25.1 安全上的注意事項

當你在現有的區域網路上設定一台 Linux 機器連結到網際網路上，你在不知不覺中已經對網際網路 — 以及在那裡的駭客們 — 開放了你的整個區域網路． 在你這樣做之前，我強烈地鼓勵你請教你的網路管理者並參考該處的安全策略． 如果你的 PPP 連線被成功地用來攻擊你的站台的話，那麼你最少也會招致同夥的使用者，網路及系統的管理者強烈的憤怒． 你也可能會發現你自己置身於非常嚴重的麻煩!

在你連接區域網路到網際網路上去之前，甚至是動態的連線你都應該要考慮安全的隱憂 — 因此早點參考 O'Reilly 的 "Building Internet Firewalls"!

26 建立 PPP 伺服器

如同前面提過的，這件工作有許多方法可以可以達成． 我在這裡所介紹的是我怎麼做的方法（使用 Cyclades 多埠串列卡）以及一組自動轉接的電話線路．

如果你不喜歡我在這裡介紹的方法，請自由地使用你自己的方法． 不過，我很樂意在這份使用說明未來的版本中包含額外的方式．所以，請把你的評論跟方法寄給我!

請注意，這一節只跟如何設定 Linux 作爲 PPP 伺服器有關．我不想（不曾）包含有關設定特殊終端機伺服器及這一類的資訊．

而且，我尚未實驗過配合 shadow 密碼使用（但應該會做）．因此目前介紹的資訊並不包含任何 shadow 需要的設定．

26.1 編譯核心

所有早先提過關於編譯核心以及核心與 pppd 版本的說明都適用．這節假設你已經讀過這份文件前面的章節!

對於 PPP 伺服器，你必須在你的核心中包含 IP forwarding 的功能． 你也可能會希望包含其它的功能（像是 IP firewalls, accounting 等等）．

如果你使用多埠串列卡，那麼你也必須明確地在你的核心中包含必要的驅動程式!

26.2 伺服器系統的概觀

我們使用相同的使用者名稱密碼配對提供撥接 PPP (and SLIP) 帳號以及 shell 帳號． 這樣做的好處（對我們而言）是使用者只需要一個帳號就可以使用所有種類的連線．

因爲我們是教育組織，我們並不對職員以及學生的使用收費，所以不必擔心計算與收費的問題．

在我們的節點與網際網路之間我們設置有一台防火牆，而這會限制某些使用者的存取，因爲撥接線路是在我們的（網際網路）防火牆內（理由很明顯，其他網際網路防火牆的細節也不會在此出現而且在任何情況下都沒有關係）．

使用者建立連往我們節點之 PPP 連結的程序（當然是在他們擁有有效的帳號後）是:

- 撥入我們的自動轉接撥號器（這是連結一堆數據機的單一電話號碼 — 第一台空閒的數據機會接起撥入的電話）．
- 使用一對有效的使用者名稱以及密碼簽入．
- 在 shell 提示符號下，發出 ppp 指令以啓動伺服器上的 PPP 程序．
- 啓動他們機器上的 PPP（可以是執行 Windows, DOS, Linux, MAC OS 或任何作業系統的機器 — 這是他們的問題）．

這台伺服器爲每個撥入的埠使用個別的 `/etc/ppp/options.ttyXX` 檔案，而這爲動態的 IP 分配設定遠端的 IP 號碼． 伺服器爲遠地的客户端使用代理位址解析協定(proxyarp)來遞送封包（經由適當的 pppd 選項加以設定）． 這避免了使用 routed 或是 gated 的需求．

當使用者從他們那端掛斷時，pppd 會偵測出來並告訴數據機要它掛斷，同時停掉 PPP 連結．

26.3 所需的全部軟體

你將會需要下列的軟體:-

- Linux，適當地編譯以包含必要的選項．
- 適合於你核心的 pppd 版本．
- 一套能夠聰明地處理數據機通訊的 'getty' 程式．
  在這裡我們使用的是 getty_ps2.0.7h，但積極考慮 mgetty． 我了解 mgetty 可以偵測出使用 pap/chap 的呼叫（ pap 是 Windows 95 使用的標準）並自動地起動 pppd，但是我尚未研究過它．
- 你的撥接使用者能夠存取的一台領域名稱伺服器 (DNS)．
  如果有可能的話你眞的應該執行你自己的領域名稱伺服器...

26.4 設定標準的撥接(shell access)

在你可以設立你的 PPP 伺服器之前，你的 Linux 機器必須能夠處理標準的撥接存取．

這份使用說明並不涵蓋這項設定方法．請參閱你選擇之 **getty** 的文件以及 ***Serial HOWTO*** 裡有關這項設定的資訊．

26.5 設立 PPP 選項檔案

你將得爲所有撥接埠設立一個全體的 `/etc/ppp/options` 檔案，其中是通用的選項． 我們使用的選項是:

```
asyncmap 0
netmask 255.255.254.0
proxyarp
lock
crtscts
modem
```

注意 - 我們沒有使用任何(明顯的)遞送設定 - 特別是這裡沒有 defaultroute 選項． 這理由是因為所有你(做為 PPP 伺服器)所要做的就是將封包從 PPP 客户端遞送到你的區域網路或網際網路並且將封包由你的區域網路遞送到你的客户端．

部所需的只是到客户機器的主機遞送，而使用 pppd 的 'proxyarp' 選項達成．

這個 'proxyarp' 選項設立一個代理位址解析協定的項目於 PPP 伺服器的位址解析協定(ARP)表格中，那基本上就是說 '將所有要給 PPP 客户端的封包送給我'． 這是建立單一 PPP 客户端的遞送路徑最簡單的一個方式 - 但你不能用這種方式在兩個區域網路之間遞送封包 - 你必須加入適當的網路遞送而不能使用代理位址解析協定．

你幾乎一定會希望提供動態分配的 IP 號碼給你的撥接使用者．你可以藉由分配 IP 號碼給每個撥接埠來達成目的． 現在，為你的每個撥接埠建立一個 /etc/ppp/options.ttyXX 檔案．

在這裡面，只要簡單地放入本地（伺服器）的 IP 號碼及該埠所要使用的 IP 號碼．例如

```
kepler:slip01
```

特別注意，在這個檔案裡你可以使用合法的主機名稱（我發現我只記得在我網路上重要機器與設備的 IP 號碼 — 名稱會更有意義才對）！

26.6 設定 pppd 以便讓使用者（成功地）執行它

因為起動 ppp 連結隱含著配置核心設備（網路界面）及操控核心遞送表格的動作，所以需要特別的權限 — 事實上需要完整的 root 權限．

幸運的是， pppd 已經設計成可以安全地設定為以 root 的身分執行．所以你必須

```
chmod u+s /usr/sbin/pppd
```

當你列出這個檔案時，它應該會如同這樣

```
-rwsr-xr-x   1 root     root        74224 Apr 28 07:17 /usr/sbin/pppd
```

如果你沒有這樣做，使用者將不能設立他們的 PPP 連結．

26.7 為 pppd 設定一個全域的別名(alias)

為了簡化我們撥接 PPP 使用者的連線程序，我們建立一個全域的別名(放在 /etc/bashrc)如此一旦他們簽入之後只要一個簡單的指令就能起動伺服端的 ppp．

這看起來像

```
alias ppp="exec /usr/sbin/pppd -detach"
```

它所作的是

- exec : 意指以這個指令所執行的程式替換正在執行的程式（在這個例子中是 shell）．
- pppd -detach : 起動 pppd 並且不要把產生的程序放入背景執行．這確保當 pppd 結束時不會留下任何程序．

當一個使用者像這樣簽入時，他們在 'w' 的輸出中看起來會像這樣

```
  6:24pm  up 3 days,  7:00,  4 users,  load average: 0.05, 0.03, 0.00
User     tty       login@  idle   JCPU   PCPU  what
hartr    ttyC0     3:05am  9:14                -
```

就這樣... 我告訴過你這是一個很簡單的，基本的 PPP 伺服系統!

27 在 null modem (直接連線)上使用 PPP

這非常地簡單 - 因爲沒有數據機所以事情變得更容易．

首先，選擇其中一部機器做爲'伺服器'，在串列埠上設立 getty 以便讓你可以從'客户端'使用 minicom 去取用此串列埠以測試連接性．

一旦你成功了，你可以除去這個 getty，除非你想用使用者名稱/密碼來確認連線． 因爲你有兩部機器的'實體控制權'，我假設你並不想如此做．

現在，在伺服器端除去 getty 並確認你已在兩部機器上正確地使用 'setserial' 來設定串列埠．

所有你要做的就是在兩個系統上啓動 pppd． 我假設你在兩台機器上都使用 `/dev/ttyS3` 建立連線． 因此，在兩部機器上執行指令:-

```
pppd -detach crtscts lock <local IP>:<remote IP> /dev/ttyS3 38400 &
```

這將會建立起連結 - 但目前爲止你還未指定遞送路徑． 你可以在每部機器上用 ping 指令來測試連結．如果這樣可以的話，終止其中一個 pppd 程序以結束連結．

你需要的遞送路徑實際上看你究竟想做什麼而定． 一般來說，其中一部機器會連上乙太網路(其及後的網路)因此所需的遞送完全和 PPP 伺服器與客户端一樣．

因此在具有乙太網路的機器上，pppd 的指令應該是

```
pppd -detach crtscts lock proxyarp <local IP>:<remote IP> /dev/ttyS3 38400 &
```

而另一部機器上是

```
pppd -detach crtscts lock defaultroute <local IP>:<remote IP> /dev/ttyS3 38400 &
```

如果你連接兩個網路(使用串列連結!)或有更複雜的遞送需求，你可以完全按照這份文件前面所提的方法來使用 `/etc/ppp/ip-up`．

Robert Hart
Port Hedland, Western Australia
Melbourne, Victoria, Australia August/October 1996 January/March 1997