PPP Client Howto

Robert Hart, hartr@hedland.edu.au

Version 1.0, 23 March 1996

This document shows how to connect your Linux PC to a PPP server.

Contents

Copyright

Distribution

This document will be posted to comp.os.linux.answers on a monthly basis or more frequently as new versions of the document are produced.

1 Introduction

PPP (Point to Point Protocol) is a mechanism for creating and running the Internet Protocol - IP (and other network protocols) over a serial link - be that a direct serial connection or a link made using modems and telephones. Using PPP, you can connect your Linux PC to a PPP server and access the resources of the network to which the server is connected (almost) as if you were directly connected to that network.

One major difference is of course speed - an Ethernet connection operates at 10 M bits per second (maximum theoretical throughput) whereas a modem operates at speed up to 33.6k bits per second. Depending on the type of PPP connection, there may be some limitations in usage of some applications and services.

1.1 Clients and Servers

When you dial into a site to establish a PPP connection, you are a *client*. The machine to which you connect is called the *server*.

This document describes the process of setting up a PPP *client*. When you are setting up a Linux box to receive and handle dial in PPP connections, you are setting up a PPP *server*, which is **NOT** covered in this document (but will be added later).

Any Linux PC can be both a PPP server and client - even simultaneously if you have more than one serial port (and modem if necessary). There is no real difference between clients and servers as far as PPP is concerned, once the connection is made. This means that PPP can (and is) used to link together two or more local areas networks - LANs, or to link one or more LANs into the Internet.

In the interests of clarity, this document refers to the machine that initiates the call (that "dials in") as the **CLIENT**, whilst the machine that answers the telephone, checks the authentication of the dial in request (using user ids, passwords and possibly other mechanisms) is referred to as the **SERVER**.

Probably the use of PPP as a client that most people are interested in is to link one or more machines at a location into the Internet. The procedure described in this document will allow you to establish and automate your Internet connection.

2 IP Numbers

Every device that connects to the Internet must have its own, unique IP number. These are assigned centrally by a designated authority for each country.

Please note that the IP numbers used throughout this document are **FICTITIOUS**! If you are connecting to the Internet, **YOU MUST** use an IP number from your own assigned range. You **MUST NOT** pick IP numbers "out of the air" and use these whilst connecting to another LAN (let alone the Internet) as this will cause total havoc.

There are IP numbers that are specifically dedicated to LANs that do not connect to the Internet. One of these sequences is 10.0.0.0 to 10.255.255.255. This is the sequence I have used as examples in this document.

For most users, who are connecting a single machine (or a small LAN) to an Internet service provider via PPP, obtaining an IP number (or more accurately, a network number) will not be necessary.

If you wish to connect a small LAN to the Internet, many Internet Service Providers (ISPs) can provide you with a dedicated subnet (a specific sequence of IP numbers) from their existing IP address space.

This will almost certainly cost you (significantly) more than a simple dial up connection using a *dynamic* IP number assigned to you automatically (and temporarily) when you connect to your ISP.

For users, who are connecting a single PC to the Internet via an Internet service provider, most providers use dynamic IP assignment.

That is, as part of the connection process, the PPP service you contact will tell your machine what its IP number is for the current session. **You are not given the same IP number each time you connect.**

This has implications for server type applications on your Linux machine such as sendmail, ftpd, httpd and so forth. The limitations of service due to dynamic IP number assignment (and ways to work around these, if possible) are discussed later in the document.

3 Aim of this Document

This document attempts to assist people who wish to use Linux and PPP to dial into a PPP server and set up an IP connection using PPP. It assumes that PPP has been compiled and installed on your Linux machine (but does cover reconfiguring/recompiling your kernel to include PPP support).

Whilst DIP (the standard way of creating a SLIP connection) can be used to forge a PPP connection, DIP scripts are generally quite complex. For this reason, this document does NOT cover using DIP to forge a PPP connection.

Instead, this document describes the standard Linux PPP software (chat/pppd).

This document at present does NOT cover :-

- Connecting and configuring a modem to Linux (in detail)
  See the Serial-HOWTO
- Setting up a PPP server on Linux
  A PPP server HOWTO is in production and will be added to this document when ready for release.
- Using DIP to make PPP connections
  Use chat instead...
- Using PAP and CHAP to authenticate PPP connections
  This will be added at a later date.
- Using socks or IP Masquerade
  There are perfectly good documents already covering these two packages.

4 Software versions covered

This HOWTO assumes that you are using a Linux 1.2.x kernel with the PPP 2.1.2 software. At present, Linux 1.3.x is still under development and the author has yet to experiment with this.

PPP version 2.2 is also available. Whilst this can be used with Linux 1.2.x, it is designed for Linux 1.3.x and requires kernel patching to work with Linux 1.2.x.

As this document is specifically designed to assist new users, it is highly recommended that you use a version of Linux and the appropriate PPP version that are known to be stable together,

Once the author has the time to experiment with Linux 1.3.x and PPP 2.2, additional information will be added to this document. (Please don't email me for this information as I will release it when it is ready - such requests will be assigned to /dev/null).

In the meantime, if you *must* use Linux 1.3.x, read the appropriate documents! The information given here regarding kernel configuration is completely wrong for 1.3.x kernels. I understand that the operation of the PPP-2.2 version with 1.3 kernels is very similar to that described here for PPP2.1.2 and 1.2.x kernels, but there are some differences.

5 Other Useful/Important Documents

Users are advised to read :-

- the documentation that comes with the PPP package
- the pppd and chat man pages
- the Linux Network Administration Guide (NAG)
- the Net-2 HOWTO.

Whilst you can use this document to create your PPP link without reading any of these documents, you will have a far better understanding of what is going on if you do so! You will also be able to address problems yourself (or at least ask more intelligent questions on the comp.os.linux... newsgroups).

These documents (as well as various others, including the relevant RFCs) provide additional and more detailed explanation than is possible in this HOWTO.

If you are connecting a LAN to the Internet using PPP, you will need to know a reasonable amount about TCP/IP networking. In addition to the documents above, you will find the O'Reilly books "TCP/IP Network Administration" and "Building Internet Firewalls" of considerable benefit!

6 Configuring your Kernel

In order to use PPP, your Linux kernel must be compiled to include PPP support. Obtain the Linux source code for your kernel if you do not already have this - it belongs in `/usr/src/linux` on the Linux standard file system (FSSTND).

In this directory you will find at least one `README` file. This contains an excellent explanation of how to go about configuring and compiling a new kernel. Read this file (it's a good idea to print it out and have a copy handy whilst you are compiling until you have done this enough times to know your way around).

One small note - the current (Linux 1.2.13) `make config` dialogue does NOT allow you to go backwards - so if you make a mistake in answering one of the questions, exit by typing CTRL C and start again...

You MUST know what cards/devices you have inside your PC if you are going to recompile your kernel!!!

In order to use PPP, you must configure the kernel to include PPP support (PPP requires BOTH pppd AND kernel support for PPP).

The following is an excerpt from the Linux 1.2.13 make config :-

```
*
* Network device support
*
Network device support? (CONFIG_NETDEVICES) [n] y
Dummy net driver support (CONFIG_DUMMY) [n] y
SLIP (serial line) support (CONFIG_SLIP) [n] n
PPP (point-to-point) support (CONFIG_PPP) [n] y
```

If you are concerned about security of your computer and/or LAN whilst connected to the Internet, you may wish to add support for IP firewalls.

Answer the other make config questions according to the hardware in your PC and the features of the Linux operating system you want. Then continue to follow the README to compile and install your new kernel.

NOTE

If you want to set up you Linux machine as a PPP server or internet gateway, you should also include IP forwarding (you will need it!).

Also, the current kernel creates only 4 PPP devices. For multi- port serial cards, you will need to edit the kernel ppp sources to obtain more ports. (See the README.linux file that comes as part of the PPP-2.1.2 distribution)

Once you have installed and rebooted your new kernel, you can start configuring and testing your PPP link.

7 Getting the Information you need about the PPP service

Before you can establish a PPP connection with a server, you need to obtain the following information (from the sysadmin/user support people of the PPP server):-

- The telephone number(s) to dial for the service (if you are behind a PABX, you also need the PABX number that gives you an outside dial tone - this is frequently digit zero)
- Does the server use DYNAMIC or STATIC IP numbers? If the server uses STATIC IP numbers, then you need to know what IP number to use for your end of the PPP connection.
  Most Internet Service Providers use DYNAMIC IP numbers. As mentioned above, this has some implications in terms of the services you can use!
- If you are using static IP numbers, ask for the network mask as well.
- What are the IP numbers of the server's Domain Name Servers? (There should be at least two.)
- Does the server require the use of PAP/CHAP?
- Does the server automatically start PPP or do you need to issue any commands to start PPP on the server once you are logged in? If you must issue a command to start PPP, what is it?

Carefully note down this information - you are going to use it!

7.1 Testing your Modem Connection

You should make sure that your modem is correctly set up and that you know which serial port it is connected to.

Remember

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

Using you terminal communications package (such as minicom), dial into the PPP server you want to connect to with a PPP session. (Note: at this stage we are NOT trying to make a PPP connection - just establish that we have the right phone number and also to find out exactly what the server sends to us in order to get logged in and start PPP).

During this process, either capture (log to a file) the entire login process or carefully (very carefully) write down exactly what prompts the remote server gives to let you know it is time to enter your user name and password (and any other commands needed to establish the connection).

It is worth dialling in at least twice - some servers change their prompts (eg with the time!) every time you log in. The two critical prompts your Linux box needs to be able to identify every time you dial in are:-

- the prompt that requests you to enter your user name;
- the prompt that requests you to enter your password;

If your server automatically starts PPP, once you have logged in, you will start to see garbage on your screen - this is the PPP server sending your machine information to start up the PPP connection.

This should look something like this :-

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

(and it just keeps on coming!)

At this point, you can hang up your modem (usually, type +++ quickly and then issue the ATHO command once your modem responds with OK).

On some systems PPP must be explicitly started on the server. This is usually because the server has been set up to allow PPP logins and shell logins using the same username/password pair. If this is the case, issue this command once you have logged in. Again, you will see the garbage as the server end of the PPP connection starts up - so you can now hang up.

If you do NOT see the garbage on your screen when the server starts up PPP, it is quite likely (though not certain) that you have done something wrong.

Notwithstanding this, some PPP servers are set up to be passive - they send nothing until the client (your computer) starts the PPP process.

However, the majority of servers are active and you should see the garbage.

If you can't get your modem to work, read your modem manual, the man pages for your communications software and the Serial-HOWTO. Once you have this sorted out, carry on as above.

8 Logging in without entering a user name/password - PAP and CHAP

It is possible for a PPP server to be set up in such a way that the user authentication is carried out by PPP itself, using PAP/CHAP. The authentication information is exchanged by the two ppp daemons themselves.

This section has yet to be added.

9 A note about Serial Ports and speed capabilities

If you are using a high speed modem (14,400 Baud or above), your serial ports needs to be capable of handling the throughput that such a modem is capable of producing. This requires your serial port to use a modern UART (Universal Asynchronous Receiver Transmitter) such as a 16550(A). If you are using an old machine (or old serial card), it is quite possible that your serial port has only an 8250 UART, which will cause you considerable problems when used with a high speed modem.

Use the command `setserial -a /dev/ttySx` to get Linux to report to you the type of UART you have. If you do not have a 16550 type UART, invest in a new serial card (available for under $50).

10 Configuring your modem

You will need to configure your modem correctly for PPP - to do this **READ YOUR MODEM MANUAL**! Most modems come with a *factory default setting* that selects the options required for PPP. The minimum configuration specifies:-

- Hardware flow control (RTS/CTS) (&K3 on many Hayes modems)

Other settings (in standard Hayes commands) you should investigate are:-

- E1 Command Echo ON (required for chat to operate)
- Q0 Report result codes (required for chat to operate)
- S0=0 Auto Answer OFF
- &C1 Carrier Detect ON only after connect
- &S0 Data Set Ready (DSR) always ON
- (depends) Data Terminal Ready

Use your communications software (eg minicom) to find out about your modem configuration and set it to what is required for PPP. Many modems report their current settings in response to AT&V, but you should consult your modem manual. If you completely mess up the settings, you can return to sanity (usually) by issuing an AT&F - return to factory settings. (For most modem modems I have encountered, the factory settings include all you need for PPP - but you should check).

Save your modem configuration in non-volatile RAM (usually the modem command AT&W will do this - but check in your modem manual).

With the correct modem configuration already in the modem, resetting the modem will activate this. Arringing things this way considerably simplifies the chat script necessary for the PPP connection.

10.1 Note on Serial Flow Control

When data is travelling on serial communication lines, it can happen that data arrives faster than a computer can handle it (the computer may be busy doing something else - remember, Linux is a multi-user, multi-tasking operating system). In order to ensure that data is not lost (data does not over run in the input buffer and hence get lost), some method of controlling the flow of data is necessary.

There are two ways of doing this on serial lines:-

- Using hardware signals (Clear To Send/Request to Send - CTS/RTS)
- Using software signals (control S and control Q).

Whilst the latter may be fine for a terminal (text) link, data on a PPP link is 8 bits - and it is quite probable that somewhere in the data there will be data bytes that translate as control S and control Q. So, if a modem is set up to use software flow control, things can rapidly go beserk!

For PPP (which uses 8 bits of data) hardware flow control is vital.

11 Setting up the PPP options file

The options to use when PPP is started up can be controlled using command line options to the pppd command or stored in a file that the PPP daemon (this is what pppd is) reads when it is invoked.

11.1 Using PPP and root privileges

Because PPP needs to set up networking devices, change the kernel routing table and so forth, it requires root privileges to do this.

The pppd program should be setuid root on installation :-

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

If `/usr/sbin/pppd` is not set up this way, then *as root* issue the command `chmod u+s /usr/sbin/pppd`

Depending on how you want your system to operate - specifically if you want ANY user on your system to be able to initiate a PPP, you should make your ppp-on/off scripts world read/execute. (This is probably fine if your PC is used ONLY by you).

However, if you do NOT want just anyone to be able to start up a PPP connection (for example, your children have accounts on your Linux PC and you do not want them hooking into the Internet without your supervision), you will need to establish a PPP group (edit `/etc/group`) and :-

- Make the ppp-on/off scripts owned by group PPP
- Make the ppp-on/off scripts read/executable by group PPP
- Make the other access rights for ppp-on/off nill.

- add the users who will be firing up PPP to the PPP group in /etc/group

You should end up with the following ownerships and permissions in response to `ls -l /usr/sbin/ppp-*`

```
-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
```

11.2 Setting up the files

You now need to be logged in as root to create the directories and edit the files needed to set up PPP, even if you want PPP to be accessible to all users.

In your `/etc` directory there should be a directory:-

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

If it does not exist - create it, `cd /etc; mkdir ppp`

If the directory already existed, it should contain a template options file called `options.tpl`. This file is at the end of this document. Print it out as it contains an explanation of all the PPP options (these are useful to read in conjunction with the pppd man pages). Whilst you can use this file as the basis of your `/etc/ppp/options` file, it is probably better to create your own options file that does not include all the comments in the template - it will be much shorter and easier to read/maintain.

If you have multiple serial lines/modems (typically the case for PPP servers), create an EMPTY `/etc/ppp/options` file and set up individual option files for each serial line on which you will be establishing a PPP connection. These are named options.ttyS1, options.ttyS2 and so forth. If there are ppp options that are common for all lines, then you can put these in the options file and the port specific options in the relevant options.ttySx file.

However, for a single PPP connection, you can happily use the `/etc/ppp/options` file. Alternatively, you can put all the options as arguments in the pppd command itself.

It is easier to maintain a setup that uses `/etc/ppp/options.ttySx` files. If you use PPP to connect to a number of different sites, you can create option files for each site in `/etc/ppp/options.site` and then specify the option file as a parameter to the PPP command as you connect.

11.3 What options should I use?

Well, as in all things that depends (sigh)...the following is a basic options file that should work for most client connections that do NOT use PAP/CHAP.

However, if it does NOT work, read through `/etc/ppp/options.tpl` and the pppd man pages. Speaking to the sysadmin/user support people who run the server into which you are connecting may also be worthwhile.

In case this file is missing, here is a shortened version covering the basic options.

```
# /etc/ppp/options (NO PAP/CHAP)
#
# use the modem control lines
modem
# use hardware flow control
crtscts
# create a default route for this connection in the routing table
defaultroute
# do NOT set up any {\tt "}escaped{\tt "} control sequences
asyncmap 0
# use a maximum transmission packet size of 552 bytes
mtu 552
# use a maximum receive packet size of 552 bytes
mru 552
#
#-------END OF SAMPLE /etc/ppp/options
```

12 Setting up your /etc/resolv.conf file

Whilst we humans like give names to things, computers really like numbers. On a TCP/IP network (which is what the Internet is), we call machines by a particular name - and every machine lives in a particular "domain". For example, my Linux workstation is called *archenland* and it resides in the *hedland.edu.au* domain. Its human readable address is thus archenland.hedland.edu.au. In order for this machine to be "findable" by other computers on the Internet, it is actually known by its IP number.

Translating (resolving) machine (and domain) names into the numbers actually used on the Internet is the business of machines that offer the Domain Name Service.

When you forge a PPP connection, you need to tell your Linux machine where it can get host name to IP number (address translation) information so that you can use the machine names but your computer can translate these to the IP numbers it needs to work.

One way is to enter every host that you want to talk to into the /etc/hosts file (which is in reality totally impossible if you are connecting to the Internet); another is to use the machine IP numbers as opposed to the names (an impossible memory task for all but the smallest LANs).

The best way is to set up Linux so that it knows where to go to get this name to number information - automatically. This service is provided by the Domain Name Server system. All that is necessary is to enter the IP numbers in your `/etc/resolv.conf` file.

Your PPP server sysadmin/user support people should provide you with at least two DNS IP numbers (only one is necessary - but two gives some redundancy in the event of failure).

Your `/etc/resolv.conf` should look something like :-

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

Edit this file (creating it if necessary) to represent the information that your ISP has provided. It should have ownership and permissions as follows :-

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

If you have already set up a `/etc/resolv.conf` because you are on a LAN, simply add the IP numbers of the PPP DNS servers to your existing file.

13 Setting up the PPP connection manually

Now that you have created your `/etc/ppp/options` and `/etc/resolv.conf` files, you can test the settings by manually establishing a PPP connection. (Once we have the manual connection working, we will automate the process).

To do this, your communications software must be capable of quitting WITHOUT resetting the modem. Minicom can do this - ALT Q (or in older version of minicom CTRL A Q)

Make sure you are logged in as root.

Fire up you communications software (such as minicom), dial into the PPP server and log in as normal. If you need to issue a command to start up PPP on the server, do so. You will now see the "garbage" you saw before.

Now quit the communications software without resetting the modem (ALT Q in minicom) and at the Linux prompt (as root) type

```
pppd -d -detach /dev/cuaX &
```

(for dynamic IP numbers) or

```
pppd -d -detach <your IP number>: /dev/cuaX
```

(for static IP numbers - NOTE the colon after the IP number here).

In both cases, change "X" to the correct number for the modem port.

The -d option turns on debugging - the ppp connection start up "conversation" will be logged to your system log - which is useful if you are having trouble.

Your modem TX/RX lights should flash as the PPP connection is established. It will take a short while for the PPP connection to be made.

At this point you can look at the PPP interface, by issuing the command

```
ifconfig ppp0
```

You should see something like :-

```
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
```

Where

- inet addr:10.144.153.10 is YOUR IP number
- P-t-P:10.144.153.5 is the SERVER IP number

(Naturally, `ifconfig` will not report these IP numbers, but the ones used by your PPP server.)

`ifconfig` also tells you that the link is UP and RUNNING!

If you get something like

```
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:63287 errors:0 dropped:0 overruns:0
         TX packets:62187 errors:0 dropped:0 overruns:0
```

your PPP connection has not been made...see the later section on debugging!

Now test the link by `ping`ing the server at its IP number as reported by the `ifconfig` output, ie

```
ping 10.144.153.51
```

You should receive output like

```
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
```

This listing will go on for ever - to stop it press CTRL C), at which point you will receive some more information :-

```
--- 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
```

So far so good.

Now try pinging a host by name (not the name of the PPP server itself) but a host at another site that you KNOW is going to be up and running...). For example

```
ping hedunx.hedland.edu.au
```

This time there will be a bit of a pause as Linux obtains the IP number for the host you have pinged from the DNS you specified in /etc/resolv.conf - so don't worry (but you will see your modem lights flash). Shortly you will receive output like

```
PING hedunx.hedland.edu.au (10.144.153.3): 56 data bytes
64 bytes from 10.144.153.3: icmp_seq=0 ttl=254 time=190.1 ms
64 bytes from 10.144.153.3: icmp_seq=1 ttl=254 time=180.6 ms
64 bytes from 10.144.153.3: icmp_seq=2 ttl=254 time=169.8 ms
64 bytes from 10.144.153.3: icmp_seq=3 ttl=254 time=170.6 ms
64 bytes from 10.144.153.3: icmp_seq=4 ttl=254 time=170.6 ms
```

Again, stop the output by pressing CTRL C and get the statistics...

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

If you do not get any response, check in the debugging section of this document.

If everything works, shut down the connection by typing

```
ppp-off
```

If that does not work, either turn off your modem or fire up your communications software and interrupt the modem with +++ and then hang up with ATH0 when you receive the modem's OK prompt.

14 Automating your connections - Creating the chat script

A chat script automates the log in and PPP start up so all you have to do (as root or as a member of the ppp group) is issue a single command to fire up your connection.

A chat script basically consists of *expect send* pairs. For example, once the modem has dialled and connected to the remote modem, we *expect* to receive a login prompt (possibly preceded by a lot of text - such as the /etc/issue file on the server machine). Once we have received this, we *send* our user name. Next, we *expect* the password prompt and *send* our password.

Quite simple in principle really.

If the ppp package installed correctly, you should have two files in /usr/sbin called

```
/usr/sbin/ppp-on
/usr/sbin/ppp-off
```

These contain templates for establishing a PPP session using chat. The ppp-off file was the one we used in the previous section to shut down our manually established PPP session (so we know it worked - if it did!).

Just in case you **do not** have these templates, here are modified and commented ones that we will be using:-

---

```
/usr/sbin/ppp-on-----------------
#!/bin/sh
#
#       ppp-on
#
#       Set up a PPP link
#
LOCKDIR=/var/lock   # Change this to suit your set up
DEVICE=cua1         # Change this to reflect the com port you are  using
PHONE=XXXXXXXX      # Change this to your ISP's phone number
USER=XXXX           # Your user name on the ISP's machine
PASSWORD=password
OUR_IP_ADDR=x.x.x.x # Your IP address for STATIC IP numbers, else 0.0.0.0

if [ -f $LOCKDIR/LCK..\$DEVICE ]
then
   echo {\tt "}PPP device is locked{\tt "}
   exit 1
fi

fix-cua \$DEVICE
(
   stty 19200 -tostop
# Change this to 115200 if your modem serial port can handle it!

   if chat -v -l LCK..$DEVICE ABORT {\tt "}NO CARRIER{\tt "} ABORT BUSY {\tt "}{\tt "} ATZ OK\
      ATDT$PHONE CONNECT {\tt "}{\tt "} ogin: \$USER ssword: \\q\$PASSWORD
   then
       pppd -d -detach asyncmap 0 modem crtscts \$OUR_IP_ADDR: /dev/\$DEVICE
       rm -f LCK..\$DEVICE
       exit 0
   else
       echo {\tt "}PPP call failed{\tt "} 1{\(>\)}&2
       exit 1
   fi
) {\(<\)} /dev/\$DEVICE {\(>\)} /dev/\$DEVICE
#------END
```

/usr/sbin/ppp-off——————

```
#!/bin/sh
DEVICE=ppp0
#
# 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 unsuccessful, ensure that the pid file is removed.
#
       if [ ! "$?" = "0" ]; then
               echo "removing stale $DEVICE pid file."
               rm -f /var/run/$DEVICE.pid
               exit 1
       fi
#
# Success. Terminate with proper status.
#
       echo "$DEVICE link terminated"
       exit 0
fi
#
# The link is not active
#
echo "$DEVICE link is not active"
exit 1
#--------END
```

The file permissions and ownerships should be :-

```
-rwxr-xr-x   1 root     PPP          502 Sep  3 20:04 /usr/sbin/ppp-off
-rwxr-xr-x   1 root     PPP          612 Sep  3 20:04 /usr/sbin/ppp-on
```

14.1 What a Chat script means

The chat command bears some discussion. A chat script is a sequence of "expect string" "send string" pairs. In particular, note that we always *expect* something before we *send* something. If we are to send something WITHOUT receiving anything first, we must use an empty expect string (indicated by "") and similarly for expecting something without sending anything. Also, if a string consists of several words, (eg NO CARRIER), you must quote the string so that it is seen as a single entity (eg "NO CARRIER")

The chat line in our template is:-

```
if chat -v -l LCK..$DEVICE ABORT "NO CARRIER" ABORT BUSY "" ATZ OK\
   ATDT$PHONE CONNECT "" ogin: $USER ssword: \\q$PASSWORD
```

Basically, this says...

- chat -v -l LCK..$DEVICE
  invoke chat and set up a lock on the serial device to ensure exclusive access ($DEVICE is a shell variable set to the port name). The -v tells chat to copy ALL its I/O into the system log (usually `/var/log/messages`)
- ABORT "NO CARRIER" ABORT BUSY
  If the script receives either of these two responses, abort the connection attempt.
- "" ATZ
  expect nothing ("") then send ATZ (resets the modem).
- OK ATDT$PHONE
  expect OK (the modem response to ATZ) then dial $PHONE (another shell variable defining the phone number to dial).
- CONNECT ""
  expect CONNECT (the modem at the other end has answered and the modems have successfully negotiated baud rate etc) and send nothing ("").
- ogin: $USER
  expect "ogin:" - the login prompt (minus the leading 'l' - this needs to be edited to reflect exactly the propt that your ISP uses) and send $USER (a shell variable set to your user name).
- ssword:
  q$PASSWORD
  expect "assword:" the password prompt (minus the leading "p") and send $PASSWORD (another shell variable set to your password). The
  q tells chat NOT to place the password in syslog if chat is invoked with -v.

This is a very simple chat script. chat has considerable error recovery capability and a number of other option. For more information consult the chat manual page (man 8 chat).

15 Creating your ppp-on file

Copy the ppp-on file to a new name (I tend to use ppp-sitename for convenience and as an aid to memory as I have quite a few). This way, you can also go back to the original if you mess up the copy you have created.

Now, we are going to edit this, using the information we gathered from the sysadmin of the PPP server.

ppp-on is a Bourne shell script. It first sets up a number of shell variables for later use.

15.1 Lock file directory (LOCKDIR)

Have a look in `/var` and below for your lock file directory. For a Linux standard file system, this will be `/var/lock` (older systems may use `/var/spool/lock`).

Edit the LOCKDIR= line so it points at the lock directory your system uses..

15.2 Serial device (DEVICE)

This is the serial device to which you have connected you modem - without the `/dev/` prefix.

Edit the DEVICE= line so it points at the serial port on which you have your modem.

15.3 PPP server phone number (PHONE)

This is the phone number to dial to reach the PPP server (don't forget any leading numbers necessary to get past a PABX - you can insert the standard dial pause command (,) if necessary.

Edit the PHONE= line so it is the PPP server's phone number.

15.4 User name (USER)

This is the user name you use to log into your PPP server.

Edit the USER= line so it is your PPP server user name (do not forget, Unix and most PPP servers are case sensitive - so get the capitals/lower case right).

15.5 Password (PASSWORD)

This is the password you use to log into your PPP server.

Edit the PASSWORD line so it contains your password, as you would type it in at the password: prompt (again, remember that passwords can be both upper and lower case!).

Now, this is a bit of a problem. As the file permissions are currently set up, anyone can read the ppp-on (or whatever you called it) file - and can thus obtain your password! So, when you have finished editing this file change its permissions to be 700 (`chmod 750 /usr/sbin/ppp-on`) so that only the root user and members of the PPP group can read this file.

Remember, even if you are the only user of your machine, you are setting up a link that connects it to the Internet (or some other network). Linux is a multi-user operating system and whilst you are connected, anyone on the Internet may try to get into your system - if they get hold of your user name and password, they can log into your account - at the very least costing you money if not severe embarrassment or worse if they do "naughty things" using your account!

15.6 Your IP address (OUR_IP_ADDR)

If the server you are connecting to uses DYNAMIC IP numbers, set this to 0.0.0.0 - or better, delete this line altogether (in which case you must also remove $OUR_IPADDR: from the pppd invocation).

If your PPP server sysadmin has given you your own IP number, you are using STATIC IP numbers. So edit this line so it is set to the IP number you have been given.

15.7 Edit the chat line

The entire chat command must be on one line - no line breaks allowed (except as specified in the chat man page).

The chat line starts with options to chat itself.

```
-l LCK..$DEVICE
```

This sets up the lock file.

One other **very** useful option for debugging is `-v` which tells chat to log the entire chat "expect send" conversation to the system log. When you are first setting up the connection, I strongly suggest you use the `-v` option, so the start of the chat line would be

```
-v -l LCK..$DEVICE
```

The expect/send pairs start with

```
"" ATZ
```

This means "expect nothing and send ATZ (which resets the modem, loading its default configuration).

Now we have

```
OK ATDT$PHONE
```

Which "expects" an OK response (the result of the ATZ) and then sends the tone dial command and phone number. If you cannot configure your modem so that its reset state is the one used for PPP, you can include a modem initialisation string here (consult your modem manual). For example:-

```
OK ATDT&K3S0=0DT$PHONE
```

As we have set our modem to be correctly configured once it receives an ATZ, we do not have to include a modem initialisation string.

The next pair is

```
CONNECT ""
```

Which means expect CONNECT (our modem sends this when it connects) and send nothing.

Now we get to the login process itself.

```
ogin: $USER
```

Which looks for the login prompt and sends our user name. Edit this expect string to reflect the login process you carefully captured when you logged in before.

We now look for the password prompt

```
assword: \\q$PASSWORD
```

Which looks for the password prompt and then sends our password. The
q tells chat NOT to log the actual text of our password to the system log file (instead ?????? is recorded instead) if we are using the -v option to chat.

If you need to issue a command to start up PPP on the server, add an additional "expect send" pair - this one would look for your login prompt (beware of characters that have a special meaning in the Bourne shell - such as $ and *or* (open and close square brackets) - and issue the ppp start up command.

In my case, my PPP server uses the standard Linux Bash prompt

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

and requires that I type

```
ppp
```

to start up PPP on the server, so my expect/send pair is

```
hartr ppp
```

(NB - chat has significantly more capability than this - including recovering from errors and missed "expects" - read the chat man pages for more information).

Unfortunately, some servers produce a very variable set of prompts! You may need to log in several times using minicom to understand what is going on and pick the stable expect strings.

15.8 Edit the pppd line

Because you have set up a `/etc/ppp/options` file, you do not need to specify ANY ppp options other than $DEVICE (which tells PPP which serial port to bind to).

However, all the PPP options can be specified, and the only requirement is that the pppd command and its options is all on one line AND the device (followed by the baud rate if specified) must be the last options specified.

As we have already seen, you can turn on debug information logging with the -d option to pppd. As we are establishing a new connection with a new script, put in the -d as the first (and possibly only option apart from $DEVICE) to pppd. (Warning: if your disk space is tight, logging pppd exchanges can rapidly extend your syslog file and run you into trouble - but to do this you must fail to connect and keep on trying for quite a few minutes).

If you have called your ppp options file anything other than `/etc/ppp/options` or `/etc/ppp/options.ttySx`, specify the file name with the `-f` option to `pppd` - eg

```
-f options.myserver
```

15.9 Save your script

OK that's it - exit the editor and NOW CHANGE THE PERMISSIONS ON YOUR SCRIPT FILE to 750!!!

16 Testing your chat script

Open a new Xterm as root (if you are in X) or open a new virtual console and log in as root.

In this new session, issue the command

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

(or whatever your system log file is).

In the first window (or virtual console) issue the command

```
ppp-on &
```

(or whatever name you have called your edited version of `/usr/sbin/ppp- on`). If you do not put the script into the background by specifying & at the end of the command, you will not get your terminal prompt back until ppp exits (when the link terminates) as we have specified the `-detach`.

Now switch back to the window that is tracking your system log.

You will see something like the following (provided you specified `-v` to chat and `-d` to `pppd`)....this is the chat script and responses being logged to the system log file followed by the start up information for `pppd` :-

```
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 16:10:22 hwin chat[19868]:  57600^M
Oct 21 16:10:23 hwin chat[19868]: ^[[;H^[[2J^M^M
Oct 21 16:10:23 hwin chat[19868]: ^M
Oct 21 16:10:23 hwin chat[19868]: ^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I      This is node  kepler.hedland.edu.au^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I^I      at Hedland Campus^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I^I       Hedland College^M
Oct 21 16:10:23 hwin chat[19868]: ^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I  Authorised user ONLY are to use this system^M
Oct 21 16:10:23 hwin chat[19868]: ^M
Oct 21 16:10:23 hwin chat[19868]: ^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I For more information, contact ComputerSystems^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I^I     on +61 (0)91 72 0400^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I^I^I     or^M
Oct 21 16:10:23 hwin chat[19868]: ^I^I       email:  help@hedunx.hedland.edu.au^M
Oct 21 16:10:23 hwin chat[19868]: ^M
Oct 21 16:10:23 hwin last message repeated 3 times
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 16:10:23 hwin chat[19868]:  ^M^M
Oct 21 16:10:24 hwin chat[19868]: Last login: Sat Oct 21 14:55:53 on  ttyC0^M
Oct 21 16:10:24 hwin chat[19868]: ^M
Oct 21 16:10:24 hwin last message repeated 9 times
Oct 21 16:10:24 hwin chat[19868]: ^I^IYou have logged into node  kepler.hedland.edu.au^M
Oct 21 16:10:24 hwin chat[19868]: ^M
Oct 21 16:10:24 hwin chat[19868]:            This is a Compaq Prolinea 486DX2/50 running Linux 1.1.54^M
Oct 21 16:10:24 hwin chat[19868]: ^M
Oct 21 16:10:24 hwin chat[19868]:        This computer operates as the main Hedland Campus communicatio
Oct 21 16:10:24 hwin chat[19868]: ^I       node, providing dial-in terminal and SLIP access,^M
Oct 21 16:10:24 hwin chat[19868]: ^I^I      Kepler also runs the Hedland end of^M
Oct 21 16:10:24 hwin chat[19868]: ^I^I    the Hedland/Newman inter-Campus WAN link^M
Oct 21 16:10:24 hwin chat[19868]: ^M
Oct 21 16:10:24 hwin chat[19868]: ^M
```

```
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
```

Note - I am using STATIC IP numbers - hence my machine sent that to the PPP server - you won't see this if you are using DYNAMIC IP numbers.)

This looks OK - so test it out as before with pings to IP numbers and host names.

Fire up you web browser or whatever and go surfing - you are connected!

17 Shutting down the PPP link

When you have finished with the PPP link, use the standard ppp-off command to shut it down (remember - you need to be root or a member of the PPP group!).

```
/usr/sbin/ppp-off
```

In your system log you will see something like:-

```
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.
```

18 I Can't get a connection...

There are any number of reasons that this can happen - chat has failed to complete correctly, you have a dirty line, etc. So check your syslog for indications.

Look in the PPP FAQ (which is really a series of questions and answers) for relevant information. This is a very comprehensive document and the answers ARE there! From my own (recent and sad) experience, if the answer to your problems is not there, the problem is NOT ppp's fault! In my case I was using an ELF kernel that I had not upgraded when I had upgraded the kernel modules. I only wasted about 2 days (and most of one night) cursing what had been a perfect PPP server before the light dawned!

19 After the link comes up

19.1 Special routing, flushing the sendmail queue etc.

There is one further piece of automation possible. Once the PPP link is established, pppd looks for `/etc/ppp/ip-up`. If this script exists and is executable, the PPP daemon executes the script. This allows you to automate any special routing commands that may be necessary and any other actions that you want to occur every time the PPP link is activated.

For example, you can get sendmail to dispatch any waiting outbound messages in the mail queue.

Similarly, you can insert the commands into ip-up to collect (using pop) any email waiting for you at your ISP.

20 Shutting down the link

The existing `/usr/sbin/ppp-off` script should work just fine. The only changes you may wish to make are for the script to wait for any outgoing email currently being processed by sendmail.

In addition, you can create a script file that will be executed once the link has been terminated. This is stored in `/etc/ppp/ip-down`. It can be used to undo anything special that you did in the corresponding `/etc/ppp/ip-up script`.

21 Routing issues on a LAN

If you are connected to a LAN but still want to use PPP, you need to address some issues of the routes packets need to take from your machine to reach your LAN (through your Ethernet interface) and also to the remote PPP server and beyond.

This section does NOT attempt to teach you about routing - it deals only with a simple, special case of (static) routing!

I strongly urge you to read the Linux Network Administrator Guide (NAG) if you are NOT familiar with routing. Also the O'Reilly book "TCP/IP Network Administration" covers this topic in a very understandable form.

The basic rule of static routing is that the DEFAULT route should be the one that points to the MOST number of network addresses. For other networks, enter specific routes to the routing table.

The ONLY situation I am going to cover here is where your Linux box is on a LAN that is not connected to the Internet - and you want to dial out to the Internet whilst still connected to the LAN.

First of all, make sure that your Ethernet route is set up to the specific network addresses available across your LAN - NOT set to the default route.

Check this by issuing a `route -n` command, you should see something like the following:-

```
[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
```

If your Ethernet interface (eth0) is pointing at the default route, (the first column will show "default" in the eth0 line) you need to change your Ethernet initialisation scripts to make it point at the specific network numbers rather than the default route (consult the Net2 HOWTO and NAG).

This will allow pppd to set up your default route as shown below:-

```
[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
```

As you can see, we have a host route to the PPP server (10.144.153.51) via ppp0 and also a default network route that uses the PPP server as its gateway.

If your set up needs to be more complex than this - read the routing documents already mentioned and consult an expert at your site!

If your LAN already has routers on it, you will already have gateways established to the wider networks available at your site. You should STILL point your default route at the PPP interface - and make the other routes specific to the networks they serve.

21.1 Note on Security

When you set up a Linux box on an existing LAN to link into the Internet, you are potentially opening your entire LAN to the Internet - and the hackers that reside there. Before you do this, I strongly urge you to consult your network administrator and site security policy. If your PPP connection to the Internet is used to successfully attack your site, you will at the very least earn the intense anger of your fellow users, network and system administrators. You may also find yourself in very much more serious trouble!

Before you connect a LAN to the Internet, you should consider the security implications of even a DYNAMIC connection - hence the earlier reference to the O'Reilly "Building Internet Firewalls"!

22 Debugging

If you can not get you PPP link to work, go back through this document and check everything - in conjunction with the output created by `chat -v` and `pppd -d` in you system log.

Also consult the PPP documentation and FAQ plus the other documents mention herein!

If you are still stuck, the comp.os.linux.misc and comp.os.linux.networking newsgroups are reasonably regularly scanned by people that can help you with PPP (including myself) as is comp.protocols.ppp

You can try sending me personal email, but I do have a day job and I do not guarantee to respond quickly (if at all) as this depends on my current work load and the state of my private life!

In particular - **DO NOT POST REAMS OF DEBUGGING OUTPUT TO THE NEWS GROUPS NOR SEND IT TO ME BY EMAIL** - the former wastes huge amounts of network bandwidth and the latter will be consigned to /dev/null (unless I have specifically requested it!).

23 Common Problems once the link is working

One problem you will find is that many service providers will only support the connection software package that they distribute to new accounts. This is (typically) for Microsoft Windows - and many service provider help desks seem to know nothing about Unix (or Linux). So, be prepared for limited assistance from them - but a growing number of ISPs are using Linux too!

23.1 I can't see beyond the PPP server I connect to

OK - your PPP connection is up and running and you can ping the PPP server by IP number (the second or remote IP number shown by `ifcongig ppp0`), but you can't reach anything beyond this.

First of all, try a `ping` to the IP numbers you have specified in `/etc/resolv.conf` as name servers. If this works, you **can** see beyond your PPP server (unless this has the same IP number as the remote IP number of your connection). So now try pinging the full Internet name of your service provider - eg

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

If this does NOT work, you have a problem with the name resolution. This is probably because of a typo in your `/etc/resolv.conf` file. Check this carefully against the information you acquired by ringing your service provider. If all looks OK, ring your service provider and check that you wrote down the IP numbers correctly.

If it STILL doesn't work (and your service provider confirms that his name servers are up and running), you have a problem somewhere else - and I suggest you check carefully through your Linux installation (looking particularly for file permissions).

If you can't ping you service provider's IP name servers by IP number, either they are down (give them a voice call and check) or there is a routing problem at your service provider's end. Again, ring them and check this out.

One possibility is that the "remote end" is a Linux PPP server where the IP forwarding option has not been specified in the kernel!

A good general test is to try hooking in to your service provider using the software that most supply for Microsoft Windows. If everything works from another operating system to exactly the same account, then the problem is with your Linux system and NOT your service provider.

23.2 I can send email, but not receive it

If you are using dynamic IP numbers, this is perfectly normal. See ”Setting up Services” below.

23.3 Why can’t people finger, WWW, gopher, talk etc to my machine?

Again, if you are using dynamic IP numbers, this is perfectly normal. See ”Setting up Services” below.

24 Using Internet services with Dynamic IP numbers

If you are using dynamic IP number (and many service providers will only give you a dynamic IP number unless you pay significantly more for your connection), then you have to recognise the limitations this imposes.

First of all, outbound service requests will work just fine. That is you can send email using sendmail, ftp from remote sites, finger users on other machines, browse the web etc.

However, your machine is NOT connected to the Internet 24 hours a day, nor does it have the same IP number every time it is connected. So it is impossible for you to receive email directed to your machine, set up a web or ftp server! As far as the Internet is concerned your machine does not exist as a unique, permanently contactable machine as it does not have a unique IP number.

If you set up a WWW (or any other server), it is totally unknown by any user on the Internet UNLESS they know that your machine is connected AND its actual (current) IP number. There are a number of ways they can get this info, ranging from you ringing them, sending them email to tell them or cunning use of ”.plan” files on a shell account at your service provider (assuming that your provider allows shell access).

Now, for most users, this is not a problem - all that most people want is to send and receive email (using your account on your service provider) and make outbound connections to WWW, ftp and other servers on the Internet. If you MUST have inbound connections to your server, you should really get a static IP number. Alternatively you can explore the methods hinted at above...

24.1 Setting up email

Even for dynamic IP numbers, you can certainly configure sendmail on you machine to send out any email that you compose locally. Configuration of sendmail can be obscure and difficult - so this document does not attempt to tell you how to do this. However, you should probably configure sendmail so that your Internet service provider is designated as you ”smart relay” host (the sendmail.cf DS option). (For more sendmail configuation info, see the sendmail documents - and look at the m4 configurations that come with sendmail. There is almost certain to be one there that will meet your needs).

Once you have sendmail configured, you will probably want to have sendmail dispatch any messages that have been sitting in the outbound mail queue as soon as the PPP connection comes up. To do this, add the command

```
sendmail -q
```

to your ip-up script.

Inbound email is a problem for dynamic IP numbers. The way to handle this is to:-

- configure your mail user agent so that all mail is sent out with a "reply to" header giving your email address at your Internet Service provider.
- use the popclient program to retrieve your email from your service provider.

24.2 Setting Up a local Name server

Whilst you can quite happily use the domain name servers located at your ISP, you can also set up a local caching only name server that is brought up by the ip-up script. The advantage of running a local (caching only) name server is that it will save you time (and bandwidth) if you frequently contact the same sites during a session.

DNS configuration for a caching only nameserver (that uses a "forwarders' line in the named.boot file pointing at your ISPs DNS) is relatively simple. The O'Reilly book (DNS and Bind) explains all you want to know about this. I also hear a rumour that there is a DNS (mini) HOWTO in production...

One point of Nettiquette: ask permission of your ISP before you start using a secondary, caching only name server in your ISP's domain. Properly configured, your DNS will not cause any problems to your ISP at all, but if you get things wrong, it can cause problems....