

# Dial-Up and The Point-to-Point Protocol (PPP)

## Contents

- **Dial-Up**
- **Point-To-Point Protocol**
  - **PPP Overview**
  - **PPP Inside**
  - **Example log of a PPP connection**
- **Glossary**

## Dial-Up

Dial-up pertains to a telephone connection in a system of many lines shared by many users. A dial-up connection is established and maintained for a limited time duration. The alternative is a dedicated connection, which is continuously in place. Dial-up lines are sometimes called switched lines and dedicated lines are called nonswitched lines. A dedicated line is often a leased line that is rented from a telephone company.

A dial-up connection can be initiated manually or automatically by your computer's modem, isdn-adaptor or other device.

## The Point-to-Point Protocol (PPP)

The Point-to-Point Protocol (PPP), documented in RFC 1661, is currently the best solution for dial-up Internet connections via phonenumber or ISDN.

## PPP Protocol Overview

PPP (Point-to-Point Protocol) is a protocol for communication between two computers using a serial interface, typically a personal computer connected by phone line to a server. For example, your Internet server provider may provide you with a PPP connection so that the provider's server can respond to your

requests, pass them on to the Internet, and forward your requested Internet responses back to you. PPP uses the Internet protocol (IP) (and is designed to handle others).

Relative to the Open Systems Interconnection (OSI) reference model, PPP provides layer 2 (data-link layer) service. Essentially, it packages your computer's TCP/IP packets and forwards them to the server where they can actually be put on the Internet.

PPP is a full-duplex protocol that can be used on various physical media, including twisted pair or fiber optic lines or satellite transmission. It uses a variation of High Speed Data Link Control (HDLC) for packet encapsulation.

PPP is usually preferred over the outdated de facto standard Serial Line Internet Protocol (SLIP) because it can handle synchronous as well as asynchronous communication. PPP can share a line with other users and it has error detection that SLIP lacks. Where a choice is possible, PPP is preferred.

## PPP Inside

PPP is a layered protocol, starting with a Link Control Protocol (LCP) for link establishment, configuration and testing. Once the LCP is initialized, one or many of several Network Control Protocols (NCPs) can be used to transport traffic for a particular protocol suite. The IP Control Protocol (IPCP), documented in RFC 1332, permits the transport of IP packets over a PPP link. Other NCPs exist for Appletalk (RFC 1378), OSI (RFC 1377), DECnet Phase IV (RFC 1762), Vines (RFC 1763), XNS (RFC 1764) and transparent Ethernet bridging (RFC 1638).

Here are some key PPP features, all of which are lacking in SLIP, the formerly used protocol for such issues:

- **Address Notification** allows a server to inform a dial-up client of its IP address for that link, but the mechanism is powerful enough for clients to request IP addresses and supports fallback configurations.
- **Authentication** is available as an option, either with the Password Authentication Protocol (PAP), or the Challenge-Handshake Authentication Protocol (CHAP). Both are documented in RFC 1334.
- **Multiple Protocols** can interoperate on the same link, simply by running additional NCPs. For example, both IP and IPX traffic can share a PPP link.
- **Link Monitoring** facilities include a link-level echo facility which can periodically check link operation.

## Example log of a PPP Connection

We want to start a connection to the internet via dial-up and our great protocol PPP.

Think of our modem (here called "/dev/modem") has already dialed the number of our provider and the provider's server picked up the phone. These two have already done the hardware-stuff like evolving connection speed, etc. A serial connection from our computer to the providers server does exist.

Now we have to start some software, which packs TCP/IP into serial data and sends it via the modem to our provider and the other way around. The PPP-Daemon or pppd (abbrev.). Let's see what it does to establish our IP connection. Let's take a look on the systems logfiles.

---

First of all, start the pppd.

```
Apr 14 15:34:28 linda pppd[2849]: pppd 2.4.1 started by thaas, uid 500
```

```
Apr 14 15:34:28 linda pppd[2849]: Perms of /dev/modem are ok, no 'mesg n'
neccesary.
```

```
Apr 14 15:34:28 linda pppd[2849]: using channel 9
```

The pppd has to create a network device which behaves like an ethernet-device to let our computer have something to send his IP-packets to.

```
Apr 14 15:34:28 linda pppd[2849]: Using interface ppp0
```

```
Apr 14 15:34:28 linda pppd[2849]: Connect: ppp0 <--> /dev/modem
```

Next step would be starting with Link Control Protocol (LCP) to authenticate ourselves.

```
Apr 14 15:34:28 linda pppd[2849]: sent [LCP ConfReq id=0x1 <asynctest 0x0> <magic
0x934fc628> <pcomp> <accomp>]
```

The server asks us to authenticate via PAP. PAP is a mechanism to authenticate via username and password.

```
Apr 14 15:34:28 linda pppd[2849]: rcvd [LCP ConfReq id=0x1 <mru 1514> <asynctest
0x0> <auth pap> <magic 0xd280aaca> <mrru 1514> <endpoint [null]>]
```

```
Apr 14 15:34:28 linda pppd[2849]: sent [LCP ConfRej id=0x1 <mrru 1514>]
```

```
Apr 14 15:34:28 linda pppd[2849]: rcvd [LCP ConfRej id=0x1 <pcomp>]
```

```
Apr 14 15:34:28 linda pppd[2849]: sent [LCP ConfReq id=0x2 <asynctest 0x0> <magic
0x934fc628> <accomp>]
```

```
Apr 14 15:34:29 linda pppd[2849]: rcvd [LCP ConfReq id=0x2 <mru 1514> <asynctest
0x0> <auth pap> <magic 0xd280aaca>]
```

```
Apr 14 15:34:29 linda pppd[2849]: sent [LCP ConfAck id=0x2 <mru 1514> <asynctest
0x0> <auth pap> <magic 0xd280aaca>]
```

```
Apr 14 15:34:29 linda pppd[2849]: rcvd [LCP ConfRej id=0x2 <accomp>]
```

```
Apr 14 15:34:29 linda pppd[2849]: sent [LCP ConfReq id=0x3 <asynctest 0x0> <magic
0x934fc628>]
```

```
Apr 14 15:34:29 linda pppd[2849]: rcvd [LCP ConfAck id=0x3 <asynctest 0x0> <magic
0x934fc628>]
```

```
Apr 14 15:34:29 linda pppd[2849]: sent [LCP EchoReq id=0x0 magic=0x934fc628]
```

```
Apr 14 15:34:29 linda pppd[2849]: cbcp_lowerup
```

```
Apr 14 15:34:29 linda pppd[2849]: want: 2
```

After some line sync we send our username and password.

```
Apr 14 15:34:29 linda pppd[2849]: sent [PAP AuthReq id=0x1 user="user@news.de"
password=<hidden>]
```

```
Apr 14 15:34:29 linda pppd[2849]: rcvd [LCP EchoRep id=0x0 magic=0xd280aaca]
```

```
Apr 14 15:34:32 linda pppd[2849]: sent [PAP AuthReq id=0x2 user="user@news.de" password=<hidden>]
```

```
Apr 14 15:34:35 linda pppd[2849]: sent [PAP AuthReq id=0x3 user="user@news.de" password=<hidden>]
```

```
Apr 14 15:34:38 linda pppd[2849]: sent [PAP AuthReq id=0x4 user="user@news.de" password=<hidden>]
```

```
Apr 14 15:34:38 linda pppd[2849]: rcvd [PAP AuthAck id=0x1 ""]
```

The server acknowledges our username/password combination. So we ask him for some IP for us, gateway and the DNS Servers.

```
Apr 14 15:34:38 linda pppd[2849]: sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <compress VJ 0f 01> <ms-dns1 0.0.0.0> <ms-dns3 0.0.0.0>]
```

```
Apr 14 15:34:38 linda pppd[2849]: sent [CCP ConfReq id=0x1 <deflate 15> <deflate(old#) 15> <bsd v1 15>]
```

The server tells us the gateway IP...

```
Apr 14 15:34:38 linda pppd[2849]: rcvd [IPCP ConfReq id=0x3 <compress VJ 0f 01> <addr 195.71.155.193>]
```

... we accept ...

```
Apr 14 15:34:38 linda pppd[2849]: sent [IPCP ConfAck id=0x3 <compress VJ 0f 01> <addr 195.71.155.193>]
```

...it send us the IP dedicated to our our network device and the IP to the DNS.

```
Apr 14 15:34:38 linda pppd[2849]: rcvd [IPCP ConfNak id=0x1 <addr 217.185.82.92> <ms-dns1 195.71.155.209> <ms-dns3 193.189.244.205>]
```

```
Apr 14 15:34:38 linda pppd[2849]: sent [IPCP ConfReq id=0x2 <addr 217.185.82.92> <compress VJ 0f 01> <ms-dns1 195.71.155.209> <ms-dns3 193.189.244.205>]
```

```
Apr 14 15:34:38 linda pppd[2849]: rcvd [LCP ProtRej id=0x4 80 fd 01 01 00 0f 1a 04 78 00 18 04 78 00 15 03 2f]
```

```
Apr 14 15:34:38 linda pppd[2849]: rcvd [IPCP ConfAck id=0x2 <addr 217.185.82.92> <compress VJ 0f 01> <ms-dns1 195.71.155.209> <ms-dns3 193.189.244.205>]
```

Here's our connection info:

```
Apr 14 15:34:39 linda pppd[2849]: local IP address 217.185.82.92
```

```
Apr 14 15:34:39 linda pppd[2849]: remote IP address 195.71.155.193
```

```
Apr 14 15:34:39 linda pppd[2849]: primary DNS address 195.71.155.209
```

```
Apr 14 15:34:39 linda pppd[2849]: secondary DNS address 193.189.244.205
```

Now we have to engage mechanisms for routing, domain lookup, etc. and after that we're ready to change data...

```
Apr 14 15:34:39 linda pppd[2849]: Script /etc/ppp/ip-up started (pid 2858)
```

```
Apr 14 15:34:39 linda modify_resolvconf: Service pppd modified /etc/resolv.conf. See info block in this file
```

```
Apr 14 15:34:40 linda pppd[2849]: Script /etc/ppp/ip-up finished (pid 2858),
status = 0x0
```

Throughout the duration of our connection the Line Control Protocol is used again. Our computer and the server play ping pong with data packets every n second to ensure the connection is still alive. If a packet is not answered within a certain timeout, the connection will be closed.

```
Apr 14 15:34:59 linda pppd[2849]: sent [LCP EchoReq id=0x1 magic=0x934fc628]
```

```
Apr 14 15:34:59 linda pppd[2849]: rcvd [LCP EchoRep id=0x1 magic=0xd280aaca]
```

---

## Glossary

### Full-duplex

Full-duplex data transmission means that data can be transmitted in both directions on a signal carrier at the same time. For example, on a local area network with a technology that has full-duplex transmission, one workstation can be sending data on the line while another workstation is receiving data. Full-duplex transmission necessarily implies a bidirectional line (one that can move data in both directions).

### OSI

OSI (Open Systems Interconnection) is a standard description or "reference model" for how messages should be transmitted between any two points in a telecommunication network. Its purpose is to guide product implementors so that their products will consistently work with other products. The reference model defines seven layers of functions that take place at each end of a communication. Although OSI is not always strictly adhered to in terms of keeping related functions together in a well-defined layer, many if not most products involved in telecommunication make an attempt to describe themselves in relation to the OSI model. It is also valuable as a single reference view of communication that furnishes everyone a common ground for education and discussion.

Developed by representatives of major computer and telecommunication companies beginning in 1983, OSI was originally intended to be a detailed specification of interfaces. Instead, the committee decided to establish a common reference model for which others could develop detailed interfaces, that in turn could become standards. OSI was officially adopted as an international standard by the International Organization of Standards (ISO). Currently, it is Recommendation X.200 of the ITU-TS.

The main idea in OSI is that the process of communication between two end points in a telecommunication network can be divided into layers, with each layer adding its own set of special, related functions. Each communicating user or program is at a computer equipped with these seven layers of function. So, in a given message between users, there will be a flow of data through each layer at one end down through the layers in that computer and, at the other end, when the message arrives, another flow of data up through the layers in the receiving computer and ultimately to the end user or program. The actual programming and hardware that furnishes these seven layers of function is usually a combination of the computer operating system, applications (such as your Web browser), TCP/IP or alternative transport and network protocols, and the software and hardware that enable you to put a signal on one of the lines attached to your computer.

OSI divides telecommunication into seven layers. The layers are in two groups. The upper four layers are used whenever a message passes from or to a user. The lower three layers (up to the network layer) are used when any message passes through the host computer. Messages intended for this computer pass to the upper layers. Messages destined for some other host are not passed up to the upper layers but are

forwarded to another host. The seven layers are:

- Layer 7: The application layer...This is the layer at which communication partners are identified, quality of service is identified, user authentication and privacy are considered, and any constraints on data syntax are identified. (This layer is not the application itself, although some applications may perform application layer functions.)
- Layer 6: The presentation layer...This is a layer, usually part of an operating system, that converts incoming and outgoing data from one presentation format to another (for example, from a text stream into a popup window with the newly arrived text). Sometimes called the syntax layer.
- Layer 5: The session layer...This layer sets up, coordinates, and terminates conversations, exchanges, and dialogs between the applications at each end. It deals with session and connection coordination.
- Layer 4: The transport layer...This layer manages the end-to-end control (for example, determining whether all packets have arrived) and error-checking. It ensures complete data transfer.
- Layer 3: The network layer...This layer handles the routing of the data (sending it in the right direction to the right destination on outgoing transmissions and receiving incoming transmissions at the packet level). The network layer does routing and forwarding.
- Layer 2: The data-link layer...This layer provides synchronization for the physical level and does bit-stuffing for strings of 1's in excess of 5. It furnishes transmission protocol knowledge and management.
- Layer 1: The physical layer...This layer conveys the bit stream through the network at the electrical and mechanical level. It provides the hardware means of sending and receiving data on a carrier.

## Protocol

In information technology, a protocol (pronounced PROH-tuh-cahl, from the Greek protocollon, which was a leaf of paper glued to a manuscript volume, describing its contents) is the special set of rules that end points in a telecommunication connection use when they communicate. Protocols exist at several levels in a telecommunication connection. There are hardware telephone protocols. There are protocols between each of several functional layers and the corresponding layers at the other end of a communication. Both end points must recognize and observe a protocol. Protocols are often described in an industry or international standard.

On the Internet, there are the TCP/IP protocols, consisting of:

- Transmission Control Protocol (TCP), which uses a set of rules to exchange messages with other Internet points at the information packet level.
- Internet Protocol (IP), which uses a set of rules to send and receive messages at the Internet address level.
- Additional protocols that are usually packaged with a TCP/IP suite, including the Hypertext Transfer Protocol (HTTP) and File Transfer Protocol (FTP), each with defined sets of rules to use with corresponding programs elsewhere on the Internet.

## Serial Links

Serial links take many forms, from 2400 bps dial-up modems (and their faster cousins), to dedicated T3

leased lines. All share two common traits – they interconnect two computers and transmit a single bit at a time in each direction. The stream of bits is assembled into bytes and then packets. The speed of a serial line is rated in bits per second (bps). They can be broken into two broad categories – synchronous and asynchronous.

- Synchronous Serial Links Synchronous links use some form of clocking, by which a clock signal is transmitted along with the data. This can take many forms. A separate signal may carry the clock. It is also possible to combine the clock and data signals together into one. Typically, one side of the link provides clocking (gives clock) for data traveling in both directions. The other side of the link takes clock. If a telephone company (telco) leased line is involved, the telco will give clock, since the data must be carefully synchronized as it moves through the telco's network.
- Asynchronous links lack any form of clock signal. Rather, a start bit is used to signal the beginning of a transmission. Once the receiver has seen the start bit, it begins counting bit times according to the pre-configured line speed. Without an explicit clock signal, the receiver risks gradually losing synchronization with the sender. For this reason, almost all asynchronous links transmit only a single byte at a time. The next byte requires a new start bit to re-synchronize the sender and receiver. Since this kind of links they lack clocking, require the sender and receiver to agree on the bit speed of the link. If they do not agree, for example if the sender transmits at 14.4 kbps but the receiver is configured for 9.6 kbps, gibberish will result, as almost anyone with an external modem has experienced from time to time.

## TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an intranet or an extranet). When you are set up with direct access to the Internet, your computer is provided with a copy of the TCP/IP program just as every other computer that you may send messages to or get information from also has a copy of TCP/IP.

TCP/IP is a two-layer program. The higher layer, Transmission Control Protocol, manages the assembling of a message or file into smaller packets that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, Internet Protocol, handles the address part of each packet so that it gets to the right destination. Each gateway computer on the network checks this address to see where to forward the message. Even though some packets from the same message are routed differently than others, they'll be reassembled at the destination.

TCP/IP uses the client/server model of communication in which a computer user (a client) requests and is provided a service (such as sending a Web page) by another computer (a server) in the network. TCP/IP communication is primarily point-to-point, meaning each communication is from one point (or host computer) in the network to another point or host computer. TCP/IP and the higher-level applications that use it are collectively said to be "stateless" because each client request is considered a new request unrelated to any previous one (unlike ordinary phone conversations that require a dedicated connection for the call duration). Being stateless frees network paths so that everyone can use them continuously. (Note that the TCP layer itself is not stateless as far as any one message is concerned. Its connection remains in place until all packets in a message have been received.)

Many Internet users are familiar with the even higher layer application protocols that use TCP/IP to get to the Internet. These include the World Wide Web's Hypertext Transfer Protocol (HTTP), the File Transfer Protocol (FTP), Telnet (Telnet) which lets you logon to remote computers, and the Simple Mail Transfer Protocol (SMTP). These and other protocols are often packaged together with TCP/IP as a "suite."

Personal computer users usually get to the Internet through the Serial Line Internet Protocol (SLIP) or the Point-to-Point Protocol (PPP). These protocols encapsulate the IP packets so that they can be sent over a dial-up phone connection to an access provider's modem.

Protocols related to TCP/IP include the User Datagram Protocol (UDP), which is used instead of TCP for special purposes. Other protocols are used by network host computers for exchanging router information.

These include the Internet Control Message Protocol (ICMP), the Interior Gateway Protocol (IGP), the Exterior Gateway Protocol (EGP), and the Border Gateway Protocol (BGP).