Transparent Remote Network Connectivity: Internet Protocol Over Serial Lines

March 4, 1992

John Landwehr
Northwestern University
email: jland@nwu.edu

This document will cover the new technologies introduced in low-cost remote network connectivity using the TCP/IP protocol family. This protocol runs over a variety of network media, including IEEE 802.3 (ethernet) and 802.5 (token ring) LAN's, X.25 lines, satellite links, and serial lines. The low-cost serial line connections offer possibilities extending from home computing to local networks connected across the world using the Serial Line Interface Protocol (SLIP) or the Point to Point Protocol (PPP). A description and comparison of these protocols will be presented as an improvement over older technologies such as dumb terminals, kermit, and X windows. Examples will be used to illustrate conceivable applications across different computing environments.

1 Introduction

Networking is currently playing a vital role in all computing environments. Computer networking has allowed machines for many years to share information. Previously, a mainframe was used to store all of the information for the network and each client would access the mainframe host independently of the others. Obvious problems occurred when the central mainframe failed.

The current trend is to spread information and resources across the network. The user who creates the file should have it accessible to everyone, but stored on his machine for easy updates. If a machine should fail on the network, the whole network does not fail, but only a minimal amount of resources are inaccessible. Printers, file servers, and communication devices are spread across the network for convenience and network failure avoidance.

Networks are easily installed in buildings by using ordinary network cabling. Adding new machines involves tapping into the network at some point by running a minimal length of cable. However, it might not always be effortless to connect machines in other buildings across a campus, or across the world. Previously, this meant expensive buried cables, dedicated phone lines, or satellite links that were not only very difficult to install, but also very unreliable and cumbersome to maintain.

Serial line connectivity allows for low maintenance communication over both short and extremely long distances. Possible connection methods include straight serial line connections, modem connections, SCSI port connections, and ISDN connections. Since almost all computers sold today include serial ports, it is no longer always necessary to purchase expensive hardware. The software for this communication method is available in all major computing environments and also offers a way to communicate across mixed environments. The network users are not bound to using only a serial line protocol across the network. Local networks in the building running ethernet can be connected through a serial link to another building anywhere in the world running another ethernet network. Possibilities can be host-to-host, host-to-LAN, or LAN-to-LAN networks. All users will have transparent access to all the resources of a LAN at any remote location.

2 The two protocols

The Serial Line Interface Protocol (SLIP) has its origins in the 3COM UNET TCP/IP implementation from the early 1980's. Around 1984, it was implemented for 4.2 Berkeley Unix and Sun Microsystems released it to the world. It quickly caught on as an easy reliable way to connect TCP/IP hosts and routers with serial lines. SLIP is commonly used on dedicated serial links and dialups for speeds between 1200bps and 19.2Kbps.

The Point to Point Protocol (PPP) was developed in the late 1980's as a functional replacement for SLIP. SLIP was very useful for a time, but PPP has been adopted as the Internet standard correcting many of the deficiencies in SLIP.

2.1 SLIP

SLIP is an extremely simple framing scheme for putting IP packets on a serial line.[1] One flag byte at the beginning, another at the end. The data packet is contained in the middle. The maximum packet size for SLIP is usually 1006 bytes including the IP and transport protocol headers (not including the framing characters). SLIP's main advantage is simplicity and therefore ease of implementation.

One major drawback with SLIP is the lack of error detection/correction. Noisy phone lines will corrupt packets in transit. With the usual line speed of 2400 baud, retransmitting a packet is very expensive. To help this particular situation, some SLIP implementations incorporate "VJ" TCP header compression,[2] but both machines must have prior knowledge of this before connection.

Although allowing for faster initial connections, both computers need to know each other's IP addresses before connecting. SLIP also provides no type identification field. Therefore, only one protocol can be run over a SLIP connection. If a serial line connects two multi-protocol computers, multiple links must be used for multiple protocols.

The SLIP link is so simple, that it is either up and running successfully carrying IP packets, or it's closed. However, this creates problems with a malfunctioning SLIP link because it is often very difficult to determine the cause.

SLIP's initial availability was limited to Berkeley 4.3BSD distributions. It is now available on many different operating systems including DOS and many dialup terminal servers.

---

[1] J. Romkey, RFC-1055 *A Nonstandard For Transmission of IP datagrams Over Serial Lines: SLIP,* BBN, June 1988.

[2] Van Jacobson, RFC-144 *Compressing TCP/IP Headers for Low-Speed Serial Links,* Lawrence Berkeley Laboratory, February 1990.

2.2 PPP

PPP is composed of three parts:[3] a method for encapsulating datagrams over serial links, an extensible link control protocol (LCP), and a family of network control protocols (NCP) for establishing and configuring different network-layer protocols.

In order to establish communication over a PPP link, the originating host would first send LCP packets to configure and test the data link.The originating host then sends NCP packets to choose and configure one or more network-layer protocols. Once configured, datagrams from each protocol can be sent over the link.

Each packet includes protocol, data, Frame Check Sequence, and Flag bytes. This protocol includes error detection in every frame. Bad frames never reach the network or intended applications. The protocol bytes allow other protocol families such as OSI, DECnet, Novell, and XNS to share the link equally with IP, without requiring that they be encapsulated within IP packets.

The network control protocol allows each host to negotiate IP addresses and whether they will use TCP header compression. This allows for dynamic address assignment which is important for hubs managing a large number of PPP connections.

The biggest complaint of PPP is often that it is over-featured. With the link management options, error checking, and multi protocol support, the link might as well be bullet-proof. Configuring PPP might be thought of as intimidating by some. However, most software packages with PPP are polished enough to establish basic links with minimal effort. There are always the more sophisticated options for those who desire the other features.

2.3 Older Technologies (kermit, terminals, X-windows)

The first protocol that is commonly compared to the serial line protocols is X windows. X is based on an asychronous network protocol rather than a procedure or system call. This allows both local and network connections making the network transparent from the user's point of view and from the application programmer's point of view. The speed is limited by the speed of actually drawing the graphics and the network speed.

The first point to stress is that X is a client-server protocol. To put this protocol in the simplest of terms, the server sends a picture of the screen to be drawn on the client, and the client sends keyboard and mouse movements back to the server. This system allows for many terminals to be connected to the server and each appear as if they are windows on the server. The end result is another monitor, keyboard, and mouse connected to the main CPU server. This advantage of this configuration is that you need only by an X terminal to use another machine's resources remotely and X terminals are moderately priced.

The major drawback is that you do not have the computing power on your desk. All system resources are at the server location. It is not possible to access a local disk, communication device, or other system resource from the local terminal location. The terminal is only a window showing

---

[3] Drew D. Perkins, RFC-1171 *The Point-to-Point Protocol for the Transmission Multi-Protocol Datagrams Over Point-toPoint Links,* CMU, July 1990.

what the server machine is actually doing. This setup offers minimum flexibility because you are limited to accessing machines that are capable of running X and do not have the actual computing machine on your desk in front of you. If your host server were to fail, your work would come to an immediate halt. Additionally, SLIP and PPP can run X, but not vice-versa.

Other remote users have been using dumb terminals for their needs. A dumb terminal is similar to an X terminal, except there are no graphics involved. Once again, there is no local processing being performed. You are completely bound to the remote machines resources. This setup, along with X windows could be compared to operating off of a mainframe. You are only as powerful as the mainframe. Files cannot be actually sent to the remote site. You can view them, but you are viewing them from the host server.

People with real processors on their desks might use kermit or some other file transfer method. This involves connecting the remote machine and only sending or receiving files. This does not allow a permanent connection, that is an integrated part of the operating system. The local machine can compute files received, and then send the answer back. There is no interactivity possible that would be useful for mounting file systems, or ongoing communication.

2.4 Comparison analysis

Both SLIP and PPP have major advantages over other remote networking options. It is easiest to describe their functions as literally stringing ethernet cable from the host machine to the remote machine. With these protocols, it is exactly as if you were hard-wired into the desired network. More specifically, it could be thought of taking the ethernet cable, connecting it to a modem, travelling a phone line to another modem, then back to ethernet. Packets are sent just as if they were on ethernet. All other machines on the network think that the remote machine is actually a local part of the network accessing all system resources.

One of the biggest advantages of these configurations is their transparent access to the network. Once configured, any networking requiring access to a remote network will automatically bring up the link. This could include dialing the modem and logging into the remote machine on demand. There are no commands that are necessary to be run each time network access is required. Then when finished with remote network traffic, the software will bring the line down automatically to save phone charges. The timeout value is user selectable, and can also be set to dedicated for hard-wired serial connections.

Both products also offer different packet size options depending on the information being sent. If interactivity is required, a small packet size would be used to avoid the jumpy-terminal effect. However, if large files are being transferred, the packet size would want to be a maximum.

The following are some important similarities and differences that users should be aware of when implementing remote IP access:

· SLIP will connect quicker to the remote network, and will create a quicker response. In fact, PPP will take about 41 seconds to send the first real packet, while SLIP takes about 33. This is due to the link management that PPP implements to determine the local machines address, and protocols that will be used. This appears as a major drawback for SLIP later when different protocols and dynamic addresses are required.

· "VJ" TCP header compression can be implemented over both. This is known as compressed SLIP (CSLIP) and is often not found in all SLIP implementations.

· SLIP uses a very simple framing scheme. Each SLIP packet contains a minimum amount of other data to be used when sending out packets. PPP packets contain a few more octets than a typical SLIP packet. This gives the appearance of SLIP actually outperforming PPP. However, when you factor in the error calculations used by PPP, it would be much wiser to have error calculations when using modems or other medium which might be prone to interference. Otherwise, packets will be constantly resent by SLIP, which in the end, will drastically reduce performance.

· The actual data size of a SLIP packet is much smaller than PPP. Most SLIP implementations range from 256 to a maximum of 1006, which are non-negotiable. PPP typically uses 1500, negotiable. For transferring large files, the data to packet overhead ratio can be greatly reduced by using PPP.

· Since SLIP is such a simple protocol, it is much easier to install. However, both the host and remote machines must be setup exactly the same. Addresses must be know before a connection is initiated. Here is where PPP offers much more flexibility. Host machines can be setup for remote access by machines with different addresses. Each phone line of the host system does not need to have an address assigned to it. All addresses can be assigned dynamically and once the initial installation is complete, the flexibility outweighs SLIP by far. Especially when it comes to multi-protocol access. PPP will allow different protocols to travel the link, whereas SLIP limits you to IP with assigned addresses.

· The biggest concern of many is the standardization of these protocols. SLIP would seem to be the standard, but the SLIP RFC actually describes itself as a non-standard and lists several deficiencies. PPP is the product of the Internet Engineering Task Force's Point-to-Point Protocol Working Group and addresses many of SLIP's deficiencies. SLIP has remained stagnant for a number of years, while PPP is undergoing active development.

3 Applications of SLIP and PPP

By far, the most frequently used applications for these connections are *telnet, rlogin,* and *FTP.* Plenty of bandwidth is available for these services. Sending SMTP mail, or accessing NNTP news is accomplished very easily with these protocols. Many users find that remote USENET news and mail reading are the most important features required in remote connections.

Other applications are possible with some care. Dial-up modem speeds can be very limiting in the applications being run. Interactive applications such as *xterm* would require a high-speed modem and might produce some sluggishness in large bitmap images. The biggest concern of bandwidth is when using Sun's Network File System (NFS). NFS will easily overwhelm these links. In these situations, very high speed links are required to avoid timeout errors when executing programs remotely mounted. It is obviously going to be somewhat slower, but by changing the mount request's *timeo* timeout parameter, NFS will run fairly smoothly.

3.1 System tested for both SLIP and PPP

The major reason for tackling this project was to have access to the campus resources from the home machine. Most universities will laugh fairly hard when an individual asks for an ethernet network connection at their house. Determination was a factor in finding a way to connect a UNIX workstation to receive the benefits of not only the campus network, but also the nationwide Internet.

3.2 Implementing SLIP

SLIP was the first protocol used to connect to the campus network. The system used was a NeXT Cube with an Intel 9600EX modem running V32 and V42bis at a rate of 19.2Kbps with no flow control from the computer to the modem. Using SLIP software, the NeXT dialed into an Annex terminal server at the computing facility on campus. The terminal server is available only to the Northwestern community and allows SLIP access with each phone line assigned to a certain IP address.

The software was configured with the phone number, the IP address assigned to that phone number, and the IP address of the terminal server. Login scripts were written to log into the terminal server, set communication options, and enter SLIP mode. From this point, the remote machine acted as if it were on the campus network. The NeXT machine was capable of doing any IP activity including telnet, NNTP news, and mail with any of the other machines on campus. In addition, other machines on campus could log into the remote machine when the link was up. Routing was automatically enabled, and the remote software directed all routing through the terminal server host.

Here is an example of the PING statistics obtained via the SLIP link:

```
51 packets transmitted, 50 packets received, 1% packet loss
round-trip (ms)  min/avg/max = 220/244/415
```

3.3 Implementing PPP

Implementing PPP on campus involved setting up a host machine for the connection because the terminal server is not configured to run PPP. Another NeXT machine in a public workstation lab was chosen to run PPP. However, this machine did not have a modem attached to it. Therefore, a somewhat unusual configuration was used.

PPP was installed on the remote NeXT machine, named *snoopy*. PPP was also installed on the lab machine, named *cutie*. *Snoopy* was assigned a IP address on a separate subnet than *cutie*. To log into *cutie*, *snoopy* first called the terminal server, named *elvex*. The script was then programmed to telnet from *elvex* into *cutie*, at which point a login prompt from *cutie* was shown. This has the same effect as putting a modem on *cutie*, and dialing in to access the login prompt.

In this situation, the *elvex* terminal server is fairly transparent to the whole operation. But it does add somewhat of a delay due to network times between *elvex* and *cutie*. For routing, the remote machine was instructed to run default routing through *cutie*. *Cutie* then instructed the rest of the campus routers to access the separate PPP subnet through *cutie*.

One benefit noticed from this situation is the possibility of using a modem pool to run PPP connections. The terminal server on campus is capable of accessing any host on campus. It is then possible to have PPP running on an office machine without a modem. The remote user can call the modem pool and access all of the resources of the office machine, including access to the rest of the campus network.

Here is an example of the PING statistics obtained via the PPP link:

```
51 packets transmitted, 51 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 238/271/692
```

3.4 Introducing a PC to the home network

Not only is there a NeXT workstation at home, but also an IBM PS/2 50Z. Rather than purchasing an ethernet card for this machine to be connected via ethernet to the NeXT, PPP was used. The IBM was hard-wired to the NeXT via serial ports at 9600 baud. PPP software was installed on the IBM to log into the NeXT and bringup the PPP link.

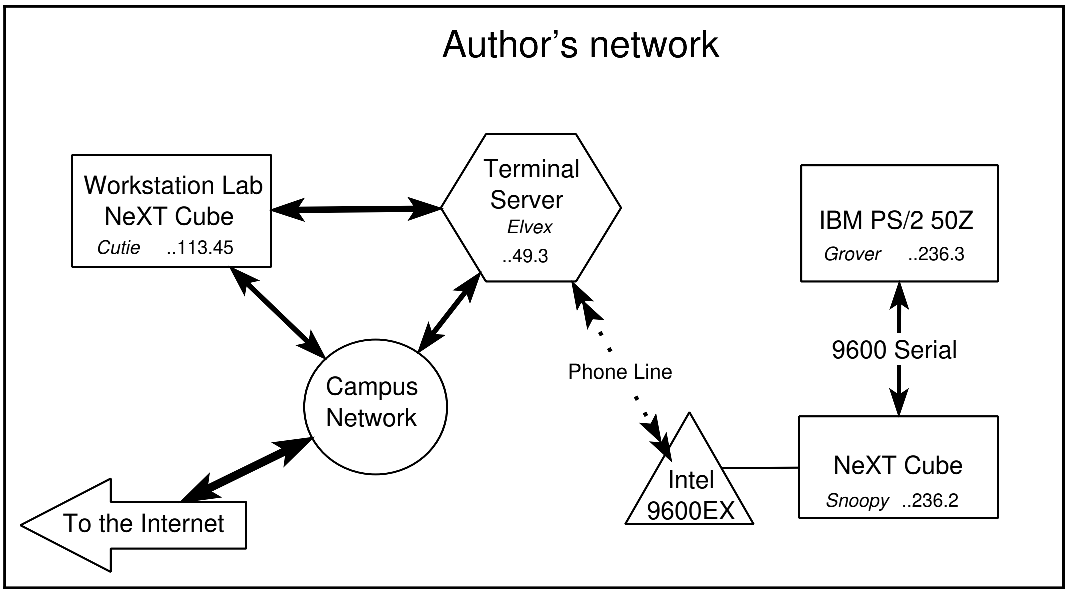
The machine name for the IBM is *grover*. *Grover* directs all of it's routing to the NeXT, *snoopy*. *Snoopy* is then also connected via modem to *cutie* via the link previously mentioned. Routing is then installed on *cutie* to direct all traffic to *grover*, through *snoopy*. Both *snoopy* and *grover* are on the same subnet.

This configuration allowed for FTP and telnet capabilities between the IBM and the NeXT. In addition, if the IBM were connected to another network, the IBM could act as a gateway out of the IBM network into the campus network.

Here is an example of the PING statistics obtained via the IBM PPP link:

```
51 packets transmitted, 50 packets received, 1% packet loss
round-trip (ms)  min/avg/max = 208/214/238
```

3.5 Illustrated network



3.6 Problems with mail

One of the first issues to deal with was sending and receiving electronic mail. Actually sending the electronic mail out was fairly easy. Once a letter was composed and sent, the networking daemon would automatically bringup the link and *sendmail* did the rest. With routing installed properly, mail could be sent from the remote workstation to any other machine on the Internet with absolutely no problems.

However, problems occur when someone tries to reply to the message. There are two possibilities. The first is that if the link is dedicated or happens to be up at the time the message is sent out, the remote machine will receive the message normally and the mailer will load it into the system. But these protocols offer on demand connections initiated generally by the remote machines. If a site sends a message and the link is not up, the message will bounce and never reach the remote machine.

One option is to configure the host machine to have outcalling. When a message is received, it will dialup the remote computer and send the message out. In this case, one might as well get a dedicated telephone line, because either computer could initiate a connection at any time, and humans would not want to pick up the phone for an annoying screech each time the phone rings.

The other option is to have all mail saved on the host machine. The remote machine can have it's return address configured to be, not its real address, but the address of the host machine. Since the host machine is always on the network, mail will always reach it. Now there needs to be a way to transfer the mail to the remote machine.

To accomplish this task, an FTP script was written for the remote machine to run under *cron*, the UNIX timer facility. At given intervals during the day, the remote machine would automatically initiate a connection to the host machine. The remote logs into the host machine and FTPs the mail file to the remote machine, then deletes the mail on the host machine. In UNIX systems, this involves writing a *.netrc* script in the users home directory that specifies the host, username, password, and commands to be executed. In the test cases, the remote machine would log into the host and copy the /usr/spool/mail/*username* files to the remote /usr/spool/mail/*username* file. To simplify this even more, a simple application was written to have a button on the screen at all times. When pressed, this program would automatically get the mail and then hang up the phone line.

3.7 Other applications tested

Another fascinating feature of the nationwide Internet is the ability to access USENET News. By using SLIP and PPP, the remote machines are able to run NNTP and use such applications as NewsGrazer on the NeXT. Here the advantage is using a graphical interface for reading the news, instead of logging into a machine and using a text only news reader.

Another key aspect of these protocols is that multiple sessions can be running simultaneously. In fact, it is possible to have many different terminal windows open at a time. Each window can then be connected to a different site, anywhere on the Internet. Obviously, the more connections running, the worse the network performance will be, especially at modem speeds.

Keeping this idea in mind, it is possible to be logged into a machine elsewhere in the world, be reading the USENET news, and also FTPing a file from an archive site all simultaneously. Previously if files were required for the home machine, it would mean bringing home the disk and reloading the software. Now it is possible to access machines all over the world and use their resources as if they were local.

4 Possible Configurations

Both SLIP and PPP offer many different options for connecting machines and networks together. In fact, this is not limited to connecting single machines together, but it is possible to connect entire networks together across the globe.

4.1 Two Workstations

Two distant workstations can use a private dial-up link for exchanging files. An example would be an isolated machine in an office with a modem, and a remote machine at home with a modem. Rather than using traditional file transfer software, the remote user could actually NFS mount the office file system for remote access at home. It would no longer be necessary to always bring files home.

4.2 Workstation to LAN

In a similar situation, the office machine might be connected to a network of other machines which share a large file system. In this case, the remote machine would have access to the entire networks's resources. The resources available to the remote machine are exactly the same as the host machine. If the host is on the Internet, the remote machine has access to the Internet as well.

In a business situation, this could be the employee working late on a project at home with full access to everything he would need at work. Or it could be an employee out in the field with a cellular modem on a laptop, that can also call in from any location and act as if there were ethernet running to his actual location.

In the educational area, many students will find this useful in connecting their machines to the campus network. Most campuses have yet to actually install wiring in the buildings for computers, and the students are left out from most of the computing resources. With these protocols, students in their rooms either on or off campus are able to use the entire campus network from home.

4.3 LAN to LAN

Many companies have offices separated by hundreds of miles. Rather than using microwave, satellite links, or leased lines, these protocols can be used to make the separate LANs appear as one. Users at both sites would view the filesystem the same way. The networks could be setup for on demand dialing, if the file transfers were infrequent. Or if constant network traffic was desired, a dedicated phone link could be up only during office hours.

These protocols can also act as a backup if other networking methods should fail. If the expensive leased line should go down, many offices would come to a complete halt. By using these protocols, the system can be brought back up immediately and act as a temporary route until regular communication can be resumed.

4.4 LAN to Internet

One of the biggest advantages of these protocols is the ability to access the worldwide Internet. It is now possible for companies and universities that are not physically connected, to access all of the aspects of the Internet from their own network. It is usually very expensive to run dedicated cables for connections to the Internet backbone. There are now many companies that offer access to the Internet for low monthly rates.

The application most often used with these services is electronic mail. The Internet provides gateways into many other mail services and is gradually becoming the preferred choice for electronic mail in the business world since it has already made its mark in the educational setting.

4.5 Related products and services

KA9Q for IBM PCs by Phil Karn: Available by anonymous FTP at ucsd.edu

Marble Teleconnect, 408.436.7299: SLIP for the NeXT Computer

Merit PPP collection: Available by anonymous FTP at merit.edu

Morning Star PPP, 800.558.7827: PPP for NeXT, Sun, and DEC

Ohio PPP collection: Available by anonymous FTP at archive.cis.ohio-state.edu

PSINet, 800.82PSI82: Public Dialup SLIP and PPP support to the Internet

TransSys Dial-Up IP, info@TransSys.com: SLIP for the NeXT Computer

4.6 Obtaining RFCs

Most of the written information about TCP/IP and the connected Internet, including its architecture, protocols, and history can be found in a series of reports known as *Request For Comments* or RFCs. The papers range from frivolous documents discussing early network humor to the most up-to-date information on new network technologies. These documents are available electronically from the Internet Network Information Center.

Anonymous FTP access is available from *nic.ddn.mil*.

Electronic mail access is available my mailing *service@nic.ddn.mil.*

Paper copies are available by calling 1-800-235-3155.

5 Final Analysis

The modem was certainly a big breakthrough in computer technology. Computer users were able to use their regular phone lines for data communication. Lines of text would go scrolling across the screen at a snails pace. Everyone thought that it was great to be able to send files across the phone line to users all around the world. The biggest hurdle in describing serial line interface protocols is distinguishing this technology from just using an ordinary modem and terminal software.

It is clear that the biggest advantage to using these protocols is actually being a part of the network that is being accessed. One computer no longer holds all of the information. The trend is for workstations on every user's desk. File servers, print servers, communication servers, and mail servers are spread throughout the network. One machine will no longer perform all the tasks for a company. Everyone machine ends up talking with every other machine.

When a remote user wants to access these resources, they are no longer limited to simply transferring the files, or using the computing power of the distant machine. Local computing power can be utilized as well as the distant resources. Just because a machine is hundreds of miles away, it doesn't have to be completely isolated. In fact, with these protocols, it will act just as if it were in the same room as all of the other machines.

Speed is the critical issue with these protocols. It wasn't too long ago that 300 bps seemed fast. Now most modems are available at 9600 bps and are pushing 14.400 bps. The new ISDN phone network will open many more possibilities for network access. Both SLIP and PPP will run over ISDN at a great speed and most networking problems will be solved with a low cost, widely available, and high-speed solution.

Many companies are offering connections for dialup PPP and SLIP. With low monthly fees, any business can call up and send and receive their electronic mail to users across the world. The Internet has gone well beyond just the educational and government uses that it was originally intended for. Even the educational users will branch their networks out further to allow access for every student from their dorm rooms.

This technology is only a few years old. Within that time, many revisions have improved the original plans for simple serial line networking. These protocols are now polished enough to allow low-cost efficient networking to absolutely everyone. Every computer user in the world can now be connected to a network wherever there is a phone, and receive the collective computing power of the world.

Additional copies of this paper are available in postscript format via anonymous FTP from: ftp.acns.nwu.edu